

---

# Realidad aumentada para el Museo García-Santosmases

---



Trabajo de Fin de Grado en Ingeniería del Software

Raúl Cobos Hernando  
María Picado Álvarez  
Álvar D. Soler Rus

Departamento de Ingeniería del Software e Inteligencia Artificial  
Facultad de Informática  
Universidad Complutense de Madrid

Junio 2016

Documento maquetado con TEXIS v.1.0+.

Este documento está preparado para ser imprimido a doble cara.

# Realidad aumentada para el Museo García-Santosmases

*Trabajo de Fin de Grado en Ingeniería del Software*

**Raúl Cobos Hernando**

**María Picado Álvarez**

**Álvar D. Soler Rus**

*Dirigida por el Doctor*

**Dr. Guillermo Jiménez Díaz**

**Departamento de Ingeniería del Software e Inteligencia  
Artificial**

**Facultad de Informática**

**Universidad Complutense de Madrid**

**Junio 2016**

Copyright ©

Contenido: Raúl Cobos Hernando, María Picado Álvarez y Álvar D. Soler Rus

Plantilla : Marco Antonio y Pedro Pablo Gómez Martín

ISBN 978-84-692-7109-4

*A nuestras familias y amigos que tanto nos han apoyado.*



# Autorización

Nosotros, Raúl Cobos Hernando, María Picado Álvarez y Álvar D. Soler Rus, alumnos matriculados en la asignatura Trabajo de Fin de Grado (TFG) en la Facultad de Informática de la Universidad complutense de Madrid durante el curso 2015/2016, dirigidos por Guillermo Jiménez Díaz, autorizamos la difusión y utilización con fines académicos, no comerciales, y mencionando expresamente a sus autores del contenido de esta memoria, el código, la documentación adicional y el prototipo desarrollado.

**Raúl Cobos Hernando, María Picado Álvarez y Álvar D. Soler Rus**



# Agradecimientos

En primer lugar, gracias a todas las personas que nos han apoyado durante la realización de este proyecto, a quienes debemos que este Trabajo de Fin de Grado haya llegado a buen puerto a pesar de los escollos que hemos tenido que pasar en este año.

Gracias también a quienes han evaluado la aplicación y nos han dado su opinión. A quienes nos han ayudado con la corrección del texto. Gracias también a las comunidades tanto de Vuforia, Unity3D y, como no, Stackoverflow, sin quienes la parte técnica de este trabajo habría sido un infierno.

En último lugar, quisiéramos dar las gracias a nuestro director de Trabajo de Fin de Grado, Guillermo Jiménez, que nos ha guiado y aguantado con cada duda que teníamos.



# Resumen

La Realidad Aumentada es una tecnología que combina imágenes reales con la superposición de imágenes virtuales. En esta memoria se detalla el trabajo hecho con esta tecnología en la creación de varios minijuegos para dotar al Museo García Santesmases de un atractivo añadido al de los objetos físicos ya expuestos. Veremos cómo ha sido el proceso de desarrollo de la aplicación, la toma de decisiones y los problemas que hemos encontrado. El nombre de la aplicación es Santesmases RA, y está disponible en Play Store para que cualquiera que visite el museo la pueda descargar y usar.

**Palabras clave:** Realidad Aumentada, Museos, Unity3D, Vuforia, Android, Videojuegos

## Abstract

Augmented Reality is a technology that combines real images with overlapping virtual images. In this report we detail the work done with this technology in the creation of various mini-games to provide to the Museo García Santesmases an attractive addition to the exposed physical objects. We will see how was the application's development process, decision-making and the problems we have encountered. The application's name is Santesmases RA, and it is available in Play Store so anyone who visits the museum can download and use.

**Keywords:** Augmented Reality, Museums, Unity3D, Vuforia, Android, Videogames



# Índice

<b>Autorización</b>	<b>VII</b>
<b>Agradecimientos</b>	<b>IX</b>
<b>Resumen</b>	<b>XI</b>
<b>1. Introducción a Santesmases RA</b>	<b>1</b>
1.1. Museo de informática García-Santesmases . . . . .	1
1.2. Objetivos y motivación . . . . .	2
1.3. Antecedentes . . . . .	3
<b>2. Estado del Arte</b>	<b>5</b>
2.1. La realidad aumentada . . . . .	5
2.2. Alcance . . . . .	6
2.2.1. Información interactiva . . . . .	6
2.2.2. Entretenimiento y educación . . . . .	7
2.2.3. Ciencia y desarrollo . . . . .	8
2.2.4. Otros . . . . .	8
2.3. Realidad aumentada en videojuegos . . . . .	9
2.3.1. Ejemplos de Realidad Aumentada en videojuegos . . .	10
2.4. Realidad aumentada en museos . . . . .	12
2.5. Herramientas de desarrollo . . . . .	13
<b>3. Diseño del videojuego</b>	<b>17</b>
3.1. Hilo argumental . . . . .	17
3.2. Mecánicas . . . . .	18
3.3. Conclusiones . . . . .	19
3.4. Conclusiones . . . . .	19
<b>4. Space Invaders</b>	<b>21</b>
4.1. Historia . . . . .	21
4.2. Nuestra versión . . . . .	21

4.3. Diseño . . . . .	22
4.4. Implementación . . . . .	24
4.5. Conclusiones . . . . .	26
<b>5. Arkanoid</b>	<b>29</b>
5.1. Historia . . . . .	29
5.2. Nuestra versión . . . . .	29
5.3. Diseño . . . . .	31
5.4. Implementación . . . . .	33
5.5. Conclusiones . . . . .	35
<b>6. Water Pipes</b>	<b>39</b>
6.1. Historia . . . . .	39
6.2. Nuestra versión . . . . .	39
6.3. Diseño . . . . .	41
6.4. Implementación . . . . .	42
6.5. Conclusiones . . . . .	46
<b>7. Detalles de implementación</b>	<b>47</b>
7.1. Realidad aumentada . . . . .	47
7.1.1. Vuforia . . . . .	47
7.1.2. Unity . . . . .	48
7.1.3. Unity + Vuforia . . . . .	49
7.2. Escenas intermedias . . . . .	50
7.2.1. SALSA with Random Eyes . . . . .	50
7.2.2. VozMe MP3 generator . . . . .	51
7.3. Persistencia de puntuaciones . . . . .	51
7.3.1. Symfony . . . . .	51
<b>8. Evaluación con usuarios</b>	<b>53</b>
8.1. Plan de evaluación . . . . .	53
8.1.1. Objetivos de la evaluación . . . . .	53
8.1.2. Métricas . . . . .	54
8.2. Descripción de la metodología del análisis de los datos . . . . .	55
8.2.1. Primera fase . . . . .	55
8.2.2. Segunda fase . . . . .	55
8.2.3. Tercera fase . . . . .	55
8.3. Primera evaluación con usuarios . . . . .	55
8.3.1. Informe de resultados . . . . .	56
8.3.2. Acciones . . . . .	58
8.3.3. Conclusiones . . . . .	59

<b>9. Conclusiones y trabajo futuro</b>	<b>61</b>
9.1. Conclusiones . . . . .	61
9.2. Líneas futuras . . . . .	62
<b>10. Aportaciones individuales</b>	<b>65</b>
10.1. Organización general del proyecto . . . . .	65
10.2. Raúl Cobos . . . . .	66
10.3. Álvar D. Soler . . . . .	68
10.4. María Picado . . . . .	69
<b>Bibliografía</b>	<b>71</b>
<b>Lista de acrónimos</b>	<b>73</b>



# Índice de figuras

1.1. Retrato de José García Santesmases por Eulogia Merle . . . . .	2
2.1. Cartel publicitario de la empresa CamOnApp . . . . .	7
2.2. Captura del entorno de desarrollo de la aplicación Aumentaty . . . . .	8
2.3. Realidad aumentada en medicina . . . . .	8
2.4. Realidad aumentada en medicina . . . . .	9
2.5. Aplicación de RA aplicada a la arquitectura . . . . .	9
2.6. AR Cards para Nintendo3DS . . . . .	10
2.7. Captura del juego Pokémon Go . . . . .	11
2.8. Captura del juego Ingress . . . . .	11
2.9. Captura del juego Paintball . . . . .	12
2.10. Captura del juego Father.io . . . . .	12
2.11. Captura del juego Night Terrors . . . . .	13
2.12. Aplicación del Museo de Mataró . . . . .	14
2.13. StreetMuseum . . . . .	15
2.14. Crononautas . . . . .	15
2.15. Captura de RACMA que muestra las zonas de influencia de las sociedades precolombinas. . . . .	15
4.1. Space Invaders original . . . . .	22
4.2. Captura durante el proceso de implementación . . . . .	23
4.3. Diagrama de los GameObjects . . . . .	26
5.1. Portada original Arkanoid . . . . .	30
5.2. Código QR de Super Mario . . . . .	31
5.3. BoxCollider Bola Arkanoid . . . . .	32
5.4. Distintos tipos de bloques del Arkanoid . . . . .	32
5.5. Diagrama del juego . . . . .	35
6.1. Simulador térmico del museo García-Santesmases . . . . .	40
6.2. Juego original de 1989 . . . . .	40
6.3. Uno de los 23 tiles que forman el tablero . . . . .	42

6.4.	Esquema de los GO y sus componentes, necesarios para el tile anterior . . . . .	43
6.5.	Sprites de las tuberías del juego Water Pipes . . . . .	44
6.6.	Captura del tablero generado de forma aleatoria . . . . .	45
6.7.	Diagrama del juego . . . . .	46
7.1.	Ejemplo de aplicación con Smart Terrain . . . . .	50
8.1.	Usuario realizando un prueba del Space Invaders . . . . .	56
9.1.	Fotografía del cuadro informativo con los disquetes . . . . .	63

# Índice de Tablas

8.1. Tabla de cuestionario SUS . . . . .	54
8.2. Promedio primera evaluación . . . . .	57



# Capítulo 1

## Introducción a Santesmases RA

El proyecto que hemos desarrollado tiene como finalidad atraer al público al museo García Santesmases con una característica nueva y atractiva. Para esto, hemos utilizado la Realidad Aumentada (en adelante RA), y con ella, diseñado tres pequeños minijuegos que requieren poco tiempo para ser jugados y dan una visión nueva de lo que la RA y los videojuegos pueden aportar a un museo. Todo esto dentro de una aplicación para móviles Android.

### 1.1. Museo de informática García-Santesmases

Inaugurado en noviembre del 2003, el museo debe su nombre al físico, profesor y precursor de la informática española José García Santesmases, el cual fue catedrático de la Universidad Complutense. En él, se exponen máquinas desarrolladas por la UCM entre 1970 y 1950. Además, se exponen las computadoras comerciales del Centro de Cálculo de la UCM, aportaciones de particulares, los propios departamentos de la Universidad, etcétera MIGS.

Además de computadoras, hay paneles explicativos que muestran información sobre éstas y sobre historia de la informática en general y cuenta también con gran cantidad de bibliografía presente en la biblioteca de la facultad.

El museo cuenta con dos plantas situadas en la 3<sup>a</sup> y 4<sup>a</sup> planta de la Facultad de Informática y su pieza más significativa es el “Analizador diferencial electrónico”, diseñado por García Santesmases y es el primer computador desarrollado en España.

Las visitas al museo son libres y cualquiera puede acercarse a la Facultad de Informática y recorrer sus pasillos, pero si se prefiere ir en grupo, se puede concertar una cita a través del email del museo. Además, a lo largo del curso se organizan visitas guiadas por el museo.

Toda la información necesaria sobre el museo se puede encontrar en la

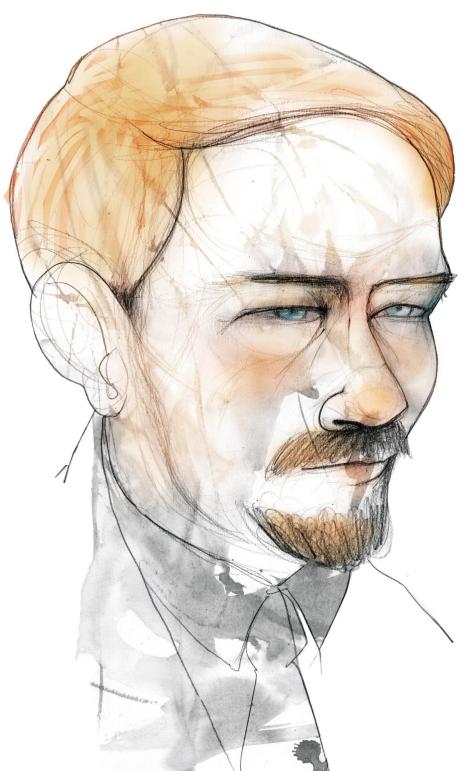


Figura 1.1: Retrato de José García Santesmases por Eulogia Merle

web<sup>1</sup>.

## 1.2. Objetivos y motivación

Nuestro objetivo es el desarrollo de una aplicación que mejorará la experiencia del usuario en un museo, en este caso, el museo de la Facultad de Informática “García-Santesmases”. Pero no solo esto, sino que nuestro objetivo es hacerlo a través de los videojuegos y utilizando la RA.

Esto lo logramos mediante una “yincana”, guiando al visitante a que recorra el museo en busca de misiones que tendrá que ir completando minijuegos para poder pasar a la siguiente misión. Así, al finalizar la visita, el jugador habrá recorrido el museo de una forma amena y divertida.

Nuestra motivación principal en la realización de este proyecto, fue profundizar en el desarrollo de videojuegos con una herramienta tan innovadora como lo es la RA. Para poder cumplir con los objetivos citados anteriormente, debemos antes centrarnos en investigar y recopilar información sobre

---

<sup>1</sup>,

distintos temas.

- Para la realización de este proyecto, es fundamental conocer los trabajos ya existentes en museos con RA y más específicamente los que añaden a esto los videojuegos en su proyecto. Esta diferencia es importante, debido a que los museos están apostando por la RA de muchas formas diferentes. Pero al estar nuestro trabajo enfocado a los videojuegos, creemos importante centrarnos en este punto más detenidamente.
- Existen diversas formas de incluir los videojuegos en un museo para dinamizar la visita al usuario. Como en nuestro proyecto decidimos hacer una yincana, otro objetivo que debemos cumplir es volver a investigar, pero esta vez, para recopilar la mayor información posible sobre cómo crear una yincana en un museo, cuál puede ser la mejor manera de ir guiando al visitante por el museo y cómo decidir qué partes del museo son las mejores para hacer captar la atención del visitante en ese lugar en concreto.
- Otra decisión importante que tenemos que tomar es que minijuegos van a componer nuestra yincana. No podemos solo pensar en qué minijuegos queremos implementar, sino también debemos tener en cuenta los puntos anteriores para pensar en esto. Visitar el museo para ver en qué partes de éste se pueden incluir los juegos nos puede ayudar mucho para saber cuales son los juegos que queremos adaptar a la RA e incluir en nuestra yincana.

### 1.3. Antecedentes

El proyecto que hemos realizado para este TFG es un proyecto nuevo y que ha sido diseñado e implementado desde el principio por nosotros. En años anteriores se realizaron trabajos de fin de grado dedicados a la RA en museos, como el realizado el año pasado (2014/2015) para el Museo de América (Caro Martínez y Hernando Hernández, 2014 - 2015). Nuestro proyecto guarda muchas similitudes con el citado anteriormente, como el uso de RA en museos para mejorar la experiencia del visitante. Por tanto, este trabajo puede que sea nuestro antecedente, aunque el concepto de proyecto sea distinto, ya que ellos utilizaban la RA como medio de información, mientras que nosotros añadimos los videojuegos en RA como medio de entretenimiento en la visita.



# Capítulo 2

## Estado del Arte

La Realidad Aumentada es el término que se usa para definir una visión a través de un dispositivo tecnológico, directa o indirecta, de un entorno físico del mundo real, cuyos elementos se combinan con elementos virtuales para la creación de una realidad mixta en tiempo real. Este tipo de tecnología se está utilizando cada vez más en distintos museos, para dinamizar su visita.

### 2.1. La realidad aumentada

Según Ronald Azuma Azuma (1997), desarrollador y líder de proyecto en New Media, Intel Corporation, y uno de los pioneros en el campo de la realidad aumentada, dice que la RA consta de las siguientes características:

- Combina elementos reales y virtuales.
- Es interactiva en tiempo real.
- Está registrada en 3D.

A su vez, consta de unos requisitos mínimos para poder ser emulada, los cuales son:

1. Una pantalla, donde mostrar la combinación de los elementos reales captados por algún dispositivo y los elementos virtuales generados por un software.
2. Un conjunto de dispositivos que capturen los elementos del entorno y nuestra situación como son una cámara, acelerómetro, giroscopio... de tal forma que permita al software tener referencias de cómo y dónde debe mostrar sus elementos virtuales.

3. Un hardware relativamente potente, para poder realizar los cálculos necesarios para mostrar el entorno que captura la cámara y ser capaz de hacer frente al software que genera los elementos virtuales y combinarlos en la pantalla con los reales.
4. Un software capaz de reconocer el entorno y calcular donde y como debe representar los elementos virtuales combinados con los reales para conseguir una visión de RA.

La evolución de los dispositivos móviles junto al aumento de su uso en la sociedad, han permitido que casi cualquier persona pueda tener acceso a la realidad aumentada.

## 2.2. Alcance

La RA es una tecnología que, aunque ya lleva muchos años, no se conocía a nivel usuario. Pero el incremento tan alto del uso de los Smartphones en los últimos años en la sociedad ha permitido crear más aplicaciones que todo el mundo pueda utilizar, ya que hoy en día casi cualquier persona de una edad comprendida entre los 16 y 55 años tiene un dispositivo móvil que le permite ejecutar una aplicación de RA en cualquier lugar y momento.

Ahora mismo, tiene diferentes aplicaciones en diversos campos como son:

### 2.2.1. Información interactiva

En este caso se utiliza para dar a conocer de forma interactiva información acerca de un elemento cercano al usuario, de tal forma que se puede mostrar un tipo de información u otra en función de la interacción entre el usuario y dicho elemento.

Ahora mismo este área está muy presente en museos como forma dinámica de conseguir una inmersión del usuario con lo que está viendo y hacer de su visita una experiencia más atractiva e incluso hasta más productiva.

Dentro de este campo también se hace uso de la RA en herramientas utilizadas para el intercambio de información en proyectos profesionales o con fines comerciales como el de mostrar catálogos de productos de una forma interactiva.

Un buen ejemplo de esto puede ser la empresa CamOnApp, que ha creado una aplicación con el mismo nombre, que crea contenidos interactivos para otras empresas. La aplicación se encarga de escanear los carteles, y te mostrará la información correspondiente CAMONAPP. Pero todas ellas tienen el mismo elemento en común, la RA como plataforma para llegar al público.



Figura 2.1: Cartel publicitario de la empresa CamOnApp

### 2.2.2. Entretenimiento y educación

En la actualidad, aunque no está todavía muy popularizado su uso, existen una gran cantidad de videojuegos que hacen uso de esta tecnología que, junto con diferentes mecánicas que se adaptan a ella, generan una novedosa experiencia para entretener el usuario.

Aunque en secciones posteriores, desarrollaremos más detenidamente este punto mostrando algunos ejemplos del uso de la RA en videojuegos, creemos interesante poner un ejemplo de este uso, pero más centrado en el entretenimiento y su aplicación en el ámbito educativo. Los profesionales de este sector, se están interesando en la RA, con el fin de hacer los contenidos académicos que deben de impartir a sus alumnos, más entretenidos y atractivos. Con esto consiguen que la motivación de sus alumnos aumente, y con ella la capacidad de aprender y así el resultado sea que se aprenda lo mismo pero de una manera más amena y divertida.

Existen diversas aplicaciones que permiten al profesor subir contenido y crear apps en realidad aumentada para llevar a cabo los aspectos anteriores, por parte de los alumnos y sobre todo de los profesores.

El proyecto Aumentaty es una idea del grupo de investigación LabHuman de la Universidad Politécnica de Valencia. Y lo que busca es potenciar el uso de los contenidos interactivos y con ello la RA, en las aulas de los colegios. Para esto, pone a disposición de los alumnos y los profesores, software gratuito que permite la creación de aplicaciones en RA de una forma sencilla AUMENTATY.



Figura 2.2: Captura del entorno de desarrollo de la aplicación Aumentaty



Figura 2.3: Realidad aumentada en medicina

### 2.2.3. Ciencia y desarrollo

Aunque todavía no está normalizado el uso de la RA en este campo, si están naciendo numerosos proyectos con el fin de desarrollar esta tecnología para utilizarse en áreas como la medicina.

Especialmente en el campo de la medicina, la inclusión de la RA puede suponer un gran avance. La unión del mundo real con elementos virtuales, puede convertirse y de hecho lo está haciendo, en una buena fórmula para facilitar la labor al especialista médico. Esto se consigue, debido a que la RA se convierte en una forma de ver la medicina nunca vista antes. Imágenes con esta tecnología, pueden ayudar al médico a tomar la mejor solución posible.

A continuación mostramos algunas de estas imágenes, que nos sirven de ejemplo de lo citado en las líneas anteriores (mocadele.net, 2015).

### 2.2.4. Otros

Además de los ya mencionados, existen todavía muchísimas áreas en las que tienen cabida en sus tecnologías la RA.

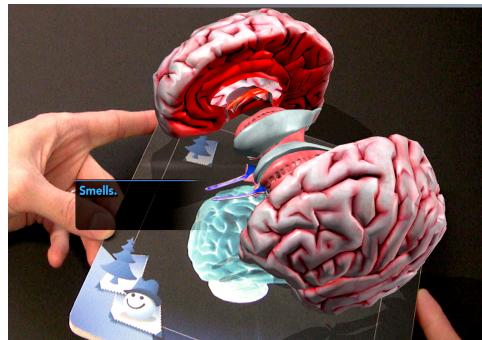


Figura 2.4: Realidad aumentada en medicina



Figura 2.5: Aplicación de RA aplicada a la arquitectura

El marketing y la publicidad están cada vez más incluyendo la RA en sus proyectos, esto se debe a que la RA es algo que suele llamar mucho la atención de las personas, lo que es en definitiva el objetivo principal de este sector.

Otros como la arquitectura, se benefician de esta tecnología al poder crear maquetas de edificios o realizar proyectos en espacios abiertos, todavía no construidos.

### 2.3. Realidad aumentada en videojuegos

La RA está siendo un gran descubrimiento para distintos ámbitos en el mundo de la informática y la tecnología. Pero, el sector que más interesado está en integrar la RA en sus desarrollos, para hacerla llegar a gran parte de la sociedad, es el de los videojuegos.

En los últimos años, han aparecido nuevas formas de jugar, y todas ellas intentan mejorar la experiencia del jugador, ya sea evadiéndolo de la realidad inventando un mundo y un espacio completamente nuevo, como hace La Realidad Virtual, o como es en nuestro caso, añadiendo información al mundo



Figura 2.6: AR Cards para Nintendo3DS

real, con la RA. Esta última consigue que el jugador maximice la interacción con el juego, este motivo es lo que la hace tan atractiva para la industria del videojuego.

### 2.3.1. Ejemplos de Realidad Aumentada en videojuegos

Hoy en día, la industria del videojuego está permanente creciendo, y la competencia es tan grande que siempre se están buscando nuevas formas de llamar la atención de los jugadores. Las nuevas tecnologías como la Realidad Virtual o la Realidad Aumentada son de las más utilizadas por las empresas para esto.

A continuación, se muestran algunos ejemplos de videojuegos con RA.

- Tarjetas RA Nintendo 3DS: Nintendo integra, desde su lanzamiento en 2011, los juegos en RA en su nueva consola. Puedes jugar a distintos minijuegos, utilizando las 6 tarjetas AR Cards. El jugador tiene también la posibilidad de desbloquear nuevos minijuegos. A parte de estas aplicaciones preinstaladas en la consola, Nintendo ha ido sacando distintos juegos que también utilizan las AR Cards. Nintendo (a)
- Pokémon Go: En esta versión, los jugadores podrán salir a explorar su ciudad e ir en busca de nuevos pokémon que se esconden por escenarios reales. Los encargados del desarrollo de este juego es la empresa Niantic Labs. Nintendo (b)
- Ingress: Este juego es otro ejemplo de desarrollo de Niantic Labs. Pero en esta ocasión, se trata de un juego de rol donde el jugador debe decidir en qué bando se encuentra, y capturar los portales. INGRESS
- Pintball: Este juego creado por Mambo Studios, utiliza la realidad aumentada para simular una partida del juego con el mismo nombre, pero



Figura 2.7: Captura del juego Pokémon Go



Figura 2.8: Captura del juego Ingress

con la gran ventaja de poder jugarlo en cualquier lugar, sin necesidad de ninguna equipación especial.

- Father.io: es un juego clasificado como shooter multijugador. Donde el objetivo del jugador es luchar en las calles, para conseguir capturar edificios e ir recolectando recursos para su equipo. Lo único necesario para poder jugar es un smartphone, pero se tiene la posibilidad de adquirir un Inceptor, un dispositivo que se adhiere al smartphone y que lo convierte en un arma láser.
- Nighth Terrors: Es un juego Survival Horror al que se le añade la RA, lo que da aumenta la sensación de miedo en el jugador. Para poder jugar, solo necesitas una casa a oscuras y un smartphone, con esto el juego se encarga de ir reconociendo los elementos de la casa o el lugar en el que te encuentres, para integrarlos al juego. El objetivo del jugador es salvar a una niña y salir con vida.

Como hemos podido observar, existen una gran variedad de juegos que utilizan RA. y aunque pueda parecer una tecnología muy joven, su crecimiento ha sido muy significativo.



Figura 2.9: Captura del juego Paintball



Figura 2.10: Captura del juego Father.io

## 2.4. Realidad aumentada en museos

La RA ha supuesto un avance muy importante en los museos ya que aporta nuevas experiencias para los visitantes. Se usan diferentes formas de usar la RA en el museo, desde aportar información adicional de un objeto expuesto (enlace a un vídeo, descarga de contenido extra, más información en texto, audios...) hasta búsquedas del tesoro, yincanas como la desarrollada en este trabajo... todo ello son experiencias añadidas y nuevas para la mayoría de museos.

A parte de los beneficios para el público, para el museo son formas de añadir contenido muy baratas en relación con lo que puede costar hacer obras para añadir salas, expositores... Además, se puede añadir contenido adicional a las aplicaciones, consiguiendo que el público vuelva. Sí se puede contemplar la posibilidad de tener smartphones y/o tabletas para el público visitante, una red WiFi abierta para facilitar la descarga de la aplicación.

Vamos a ver algunos de los museos que aplican la RA:

- Centro de artes Ca l'Árenas del Museo de Mataró: para la exposición Mar de Fons el museo dispone de una aplicación que, al enfocar a cualquiera de los cuadros, se muestra información extra. (Bosco y Caldana,



Figura 2.11: Captura del juego Night Terrors

2012)

- Street Museum del Museo de Londres: una de las más famosas. Añade contenidos en el exterior del museo. Básicamente, nos permite ver fotografías antiguas de los sitios donde estamos, dándonos una visión de cómo era mientras vemos cómo es ahora ayudándose del posicionamiento GPS del dispositivo. (of London, 2015)
- Crononautas del Museo Thyssen-Bornemisza de Madrid: esta aplicación pone al usuario como protagonista de una aventura con toques de ciencia ficción donde el usuario debe tomar decisiones que afectarán a la aventura. Además de servir de hilo conductor para la visita en el museo, aporta contenido adicional sobre las obras.
- RACMA del Museo de América de Madrid: aplicación desarrollada por compañeros de esta misma facultad el curso pasado. En ella se ofrece información a los usuarios sobre distintas sociedades precolombinas. Además, nos muestra sobre un gran mapa mudo que se encuentra en el museo el lugar de influencia de varias de esas tribus y se puede interactuar con los personajes.

## 2.5. Herramientas de desarrollo

En este apartado explicaremos las herramientas que existen para poder implementar RA.

- Vuforia: es el framework que hemos utilizado para hacer toda la parte de RA. Es gratuito y tiene una gran comunidad, así como buenos ejemplos y tutoriales. Facilita mucho el trabajo, y se puede utilizar con diferentes SDK (Android, iOS, Unity 3D, ahora con gafas de realidad virtual también...).

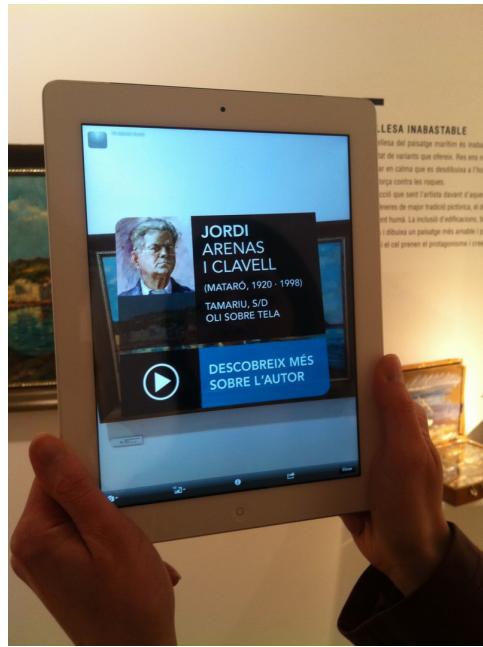


Figura 2.12: Aplicación del Museo de Mataró

- Layar: Proporciona un SDK con soporte para Android e IOS que permite integrar la RA en nuestras aplicaciones. Es de pago aunque tiene una versión de prueba de 30 días. Aunque no es una de las más potentes, si que es una de las más sencillas ya que contienen un montón de utilidades dentro del propio SDK. No está orientada a los videjuegos sino a la muestra de información interactiva basándose en la RA.
- ARTool Kit: Es de código abierto y es gratuita. Tiene soporte para múltiples SDK como son Mac OS X, Linux, Android e IOS, a través del plug-in ARToolkit para Unity3D. Tiene una gran comunidad detrás y la documentación está bastante completa donde incluyen tutoriales, APIs detalladas y multitud de ejemplos.<sup>1</sup>

<sup>1</sup><http://www.hitl.washington.edu/artoolkit/>



Figura 2.13: StreetMuseum



Figura 2.14: Crononautas



Figura 2.15: Captura de RACMA que muestra las zonas de influencia de las sociedades precolombinas.



## Capítulo 3

# Diseño del videojuego

El proyecto que se ha realizado es una yincana, con videojuegos en RA, por el museo García-Santesmases, de la Facultad de Informática de la Universidad Complutense. Esta se compone de varios minijuegos en RA y un hilo argumental. Las funciones de este último son las de ir dirigiendo al visitante por el museo y ayudarle a ir completando las diferentes misiones que se le van a ir presentando para poder finalizar con éxito la yincana.

La idea principal del proyecto es la diseñar e implementar estos minijuegos para que podamos adaptarlos a la RA, y hacer del museo no solo un espacio que aporte información interesante al usuario sobre la historia de la informática, sino que haga de éste un lugar más interactivo y entretenido para sus visitantes.

En este caso, hemos tenido la suerte de realizar un proyecto sobre videojuegos en RA en el museo García-Santesmases situado en la facultad de Informática. Esto nos brinda la oportunidad de poder entretenir al visitante mediante los videojuegos pero es importante destacar que a la vez estamos enseñando en primera persona historia de los videojuegos. Esto lo conseguimos al proponer al visitante que juegue a distintos videojuegos míticos como lo son el Space Invader, el Arkanoid o el Pipe Manía, juegos que hemos elegido para implementar en este trabajo.

Con el fin de hacer uso de la tecnología actual que se nos brinda para el desarrollo de la RA hemos implementado estos tres juegos citados anteriormente. Pero se tiene que tener en cuenta que los tres son juegos que sus primeras versiones cuentan con más de 20 años de antigüedad, por lo que hemos tenido que adaptarlos para que fuera viable jugarlos con RA, aunque siempre manteniendo su esencia original.

### 3.1. Hilo argumental

El videojuego nos traslada a una nave espacial que está siendo atacada por unos alienígenas. El piloto de la nave nos pondrá en situación y nos irá

pidiendo que realicemos diferentes tareas para que la nave pueda despegar. De esta forma conseguimos que el jugador se mueva por diferentes sitios del museo, ya que a cada minijuego solo se puede jugar en un punto del museo. Así añadimos dinamismo a la aplicación. Además, hay un sistema de puntuación diferente para cada minijuego y solo una oportunidad para cada minijuego.

Al principio nos pedirá que destruyamos a los alienígenas que nos están atacando. Para ello, debemos salir al exterior de la nave (la puerta de entrada de la facultad), donde comienza el primer minijuego al enfocar el cartel de la Facultad de Informática. Así comenzamos a jugar al Space Invaders. Conseguiremos puntos cada vez que destruyamos un invasor y perderemos puntos si no conseguimos destruir todos antes de que destruyan el escudo de la nave.

A continuación, el piloto nos pedirá que limpiemos los restos de las naves invasoras. Para esto debemos ir a los mandos del robot de reciclaje (la máquina arcade que hay en la tercera planta del museo) y así comenzará el segundo minijuego, el Arkanoid. En este conseguimos puntos cuando la bola choca contra la “chatarra espacial”.

Por último, libres de enemigos y con el camino despejado, solo nos queda un escollo para poder escapar. Los tubos del sistema de refrigeración de la nave no están bien alineados y tenemos que rehacer el camino desde el depósito hasta el motor. Para ésto debemos ir al puesto de mando donde está el panel informativo del sistema de combustión (cartel “Simulador Básico de Centrales térmicas” del otro pasillo de la tercera planta) y reajustarlos. Aquí comienza el tercer y último minijuego, WaterPipes. Este minijuego va por tiempo, y debemos encontrar un camino para el combustible antes de que se agote. La puntuación, con este minijuego, solo puede disminuir, pero en nuestra mano está que el camino sea lo más corto posible y así se nos reste el mínimo posible de puntos.

Al finalizar, se nos pedirá que introduzcamos nuestro nombre, y este se añadirá a un “Hall de la fama” de todos los usuarios del juego, ordenados por puntuación. Con ésto, hemos querido intentar que el juego sea rejutable por los usuarios, que intentarán conseguir más puntos afinando su puntería o su velocidad para quedar más alto del “Hall de la fama”.

### 3.2. Mecánicas

Uno de los principales problemas de la RA son las mecánicas. Es complicado adaptar un juego haciendo uso de la RA y elegir las mecánicas apropiadas para que este no se vea perjudicado por ella.

En este caso, se implementan juegos “clásicos”, por lo que un usuario que más o menos haya sido jugador, debiera conocer las mecánicas de estos, por lo que la comprensión inicial de estas no debiera ser el principal problema.

Ahora bien, hay que hacerle comprender al jugador como debe ejecutarlas y que estas le resulten cómodas, porque el hecho de utilizar la RA limita las mecánicas considerablemente. En primer lugar, el tener que estar enfocando a un “target” dificulta la acción; el usuario deberá estar apuntando a este y eso restringe aquellas mecánicas que puedan propiciar movimientos bruscos.

Otra dificultad reside en la ergonomía. El hecho de tener que apuntar con el smartphone, obliga a coger este de una determinada forma que deja apenas dos dedos libres, los cuales tendrán un área limitada donde poder pulsar. Además, el hecho de que sea un juego para móviles conlleva también limitaciones a la hora de recrear juegos que se jugasen con un mando o que requieran un amplio campo de visión.

El planteamiento teniendo en cuenta las limitaciones de la RA, sumadas a nuestra inexperiencia en el desarrollo de videojuegos ha sido el de implementar tres videojuegos con no mucha complejidad de desarrollo y con mecánicas muy simples que encajen con la RA.

En primer lugar, el jugador se encuentra un menú principal que le permite comenzar cuando este lo deseé y consultar las puntuaciones; algo importante como hemos mencionado anteriormente es que utiliza una interfaz muy simple con botones grandes. A continuación, empieza el hilo principal que sirve de conductor para el jugador y esta compuesto por escenas que le dicen a dónde se tiene que dirigir para comenzar el juego. Tras estas, en cada uno de los juegos se muestra la información que describe sus mecánicas y que le dice a el jugador cuál será el “target”al que deberán apuntar. Y en último lugar el juego en sí, que comenzará al reconocer el “target” y que contienen tres elementos fundamentales:

- El tiempo, el cual supone un reto para el jugador.
- Control. El jugador interactúa con el juego en el caso del Space Invaders apuntando con el propio dispositivo y pulsando sobre la pantalla; en el caso del Arkanoid deslizando el dedo sobre esta para desplazar la plataforma, y el Water Pipes, donde se pulsa sobre las tuberías para intercambiarlas.
- Puntuación, lo que motiva al jugador no solo a pasarse el juego sino a hacerlo de la mejor forma posible.

### 3.3. Conclusiones

En general hemos quedado bastante satisfechos con el resultado final. Sí nos hemos dado cuenta que podríamos haber implementado más minijuegos, pero los primeros meses fueron bastante lentos en cuanto a desarrollo.

Creemos que la yincana es una buena manera para que el usuario del museo se mueva por éste. Y que el interactuar con los elementos del museo

mejorarán su experiencia. Incluso es probable que pudiese atraer visitas, que motivadas por el hecho de probar esta tecnología relativamente “novedosa”, visiten el museo.

Por otra parte, se considera un acierto el sistema de puntuaciones, ya que anima al usuario a volver a jugar. Pero en esta ocasión ya conoce las mecánicas y es capaz de percibir más los detalles de las escenas, los cuales pueden contener referencias del museo lo que refuerza el objetivo del videojuego.

## Capítulo 4

# Space Invaders

Es el primero de los juegos de la yincana, el jugador comienza en la puerta principal de la Facultad. Inspirado en el famoso Space Invaders, el jugador, desde una perspectiva en primera persona, tendrá que disparar a un enjambre de Invaders que pretenden destruir las defensas de la nave donde se desarrolla la historia.

### 4.1. Historia

Space Invaders es uno de los juegos arcade clásicos más conocidos. La primera versión salió al mercado en 1978, hace casi cuarenta años (Wikipedia, Space Invaders). Uno de los precursores del género shoot ém up. El jugador controla una nave espacial que se mueve horizontalmente y debe hacer frente a hordas de alienígenas enemigos que atacan al jugador disparándole proyectiles. Además, a veces el jugador cuenta con pequeñas construcciones que hacen la labor de búnker donde ponerse a cubierto de los disparos, aunque éstos se van destruyendo.

Space Invaders está ampliamente extendido en la cultura popular ya que es uno de los grandes clásicos, por eso hemos considerado acertado incluirlo en nuestro proyecto.

### 4.2. Nuestra versión

Nosotros hemos decidido darle un cambio a la jugabilidad del juego. En nuestra versión utilizamos la RA para que la experiencia sea completamente diferente. Nuestro juego arrancará al detectar el cartel de FACULTAD DE INFORMÁTICA (Text Recognition), mostrándonos unos invasores alienígenas sobre el cartel, y unas defensas bajo éste.

Para destruir a los invasores, lo que tenemos que hacer es mover nuestro Smartphone para mover nuestra cámara y pulsar en la pantalla para disparar.

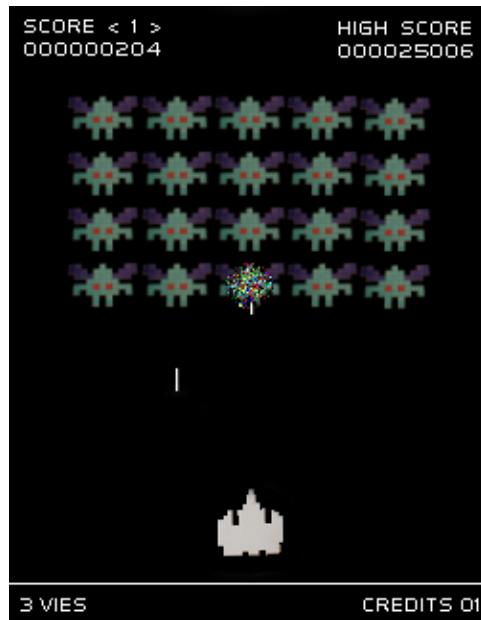


Figura 4.1: Space Invaders original

Hemos pasado de manejar la nave defensora en tercera persona, a hacerlo en primera persona, con un punto de mira en el centro de la pantalla que nos marca en qué dirección irán los láseres de nuestra torreta de defensa, convirtiendo el juego en un First Person Shooter. Tendremos que hacerlo antes de que los enemigos consigan destruir el escudo de nuestra nave espacial (cuando pasa del verde al rojo). Iremos obteniendo puntos según destruyamos naves enemigas y perderemos puntos al recibir impactos en el escudo, por lo que cuanto más rápidos seamos, más puntos obtendremos.

Con estos cambios hemos conseguido (a nuestro juicio) transformar un clásico de los arcade en un nuevo juego que utiliza la RA dando una experiencia diferente.

### 4.3. Diseño

El juego se compone de una única escena que contiene todo el juego. Básicamente se compone de:

1. La cámara de Vuforia, que a su vez tiene los siguientes hijos:
  - a) El canvas con la Interfaz de usuario (puntos y mensajes de inicio y fin del juego).
  - b) El punto de mira que utilizamos para apuntar al disparar, que también está hecho con un canvas.

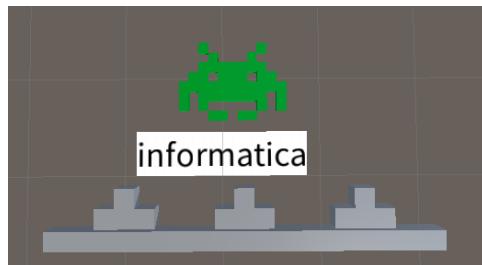


Figura 4.2: Captura durante el proceso de implementación

- c) El Cannon (Cañón de disparo) que representa nuestra arma. Básicamente dibuja una línea hacia el infinito para que de la sensación de un puntero láser para apuntar, además, desde su posición se lanza el raycast que calcula las colisiones con los posibles enemigos.
- 2. Un GameObject vacío llamado SpaceInvadersGame que contiene la clase singleton que gestiona el juego y la información mostrada por la interfaz.
- 3. El TextRecognition que sirve para cargar la detección de textos. A éste le hemos añadido un diccionario propio de palabras para poder leer texto en castellano. El diccionario contiene únicamente dos palabras, facultad e informática. Hemos configurado el TextRecognition de manera que sólo busque las palabras que están en su white list (lista blanca), que son las dos antes mencionadas, así las operaciones son más ligeras ya que no tiene que comprobar las miles de palabras. Vuforia (b)
- 4. Word representa a una palabra detectada por Vuforia. Se puede configurar para que represente cualquier palabra detectada o alguna en particular. Nosotros lo utilizamos para representar en particular la palabra INFORMÁTICA. Éste es el GameObject que sustituye al ImageTarget que utilizábamos en el pasado. Al detectar la palabra “INFORMÁTICA” en la cámara de RA, activa sus hijos y “avisa” al gestor del juego de que debe empezar a ejecutarse.
- 5. Un Enjambre, que contiene la lógica para crear varios Invasores y posicionarlos a cada uno en su sitio, así como para moverlos todos juntos.
- 6. Las copias de los invasores, las cuales disparan a veces a las defensas.
- 7. Las defensas, un objeto en tres dimensiones que representa a las defensas del jugador. Van cambiando de color, desde el verde al rojo

según van recibiendo impactos de los invasores (o del propio jugador que apunta mal, para ser algo más realista).

#### 4.4. Implementación

Pasamos a explicar qué clases componen el juego y para qué las utilizamos.

1. Defense.cs: Gestiona las defensas del usuario. Marca el color de inicio y el de final que debe tener la defensa para calcular los colores intermedios. Además gestiona las colisiones. Va añadido al GameObject de las defensas.
2. Projectile.cs: Muy simple. Va asociada a los proyectiles y los destruye al pasar unos segundos en escena. Es para que los proyectiles que no impacten con nada, no se queden siempre en la escena. En la versión final, los proyectiles con este Script solo son lanzados por los enemigos.
3. Enjambre.cs: Se encarga de gestionar la inicialización del Enjambre y de sus invasores (colocándolos en la posición que les corresponda en función de cuántos sean y cuántas filas queremos que haya) y el movimiento del Enjambre (del que “cuelgan” los invasores), así como la escala de los invasores. Además contiene la información para saber si se han eliminado a todos los invasores o no. Va añadido a un GameObject vacío llamado Enjambre, el cual cuelga del detector de texto y del cual cuelgan los invasores.
4. GameManager.cs: Es la clase que gestiona el juego en sí. Es un singleton y se le llama desde la mayoría de los otros scripts. Gestiona la interfaz de usuario, mostrando mensajes y los puntos cuando empieza el juego, además de cuando se puede disparar, etcétera. Es componente del GameManager, un GameObject vacío también.
5. Invader.cs: Lógica del invasor. Gestiona los disparos de los invasores, la muerte de éstos y el sonido que hacen al ser destruidos. Va añadido al GameObject Invader.
6. TextInformaticaTrackableEventHandler.cs: Implementación propia de la clase ITrackableEventHandler de Vuforia. Va asociada al Text de RA. Al “encontrarse” y si no está instanciado ya (es decir, que no se ha “encontrado” varias veces), le dice al GameManager que comience el juego, indicándole dónde están el Enjambre y las Defensas en relación al Text.
7. TextTimer.cs: De manera muy sencilla destruye el texto (de la interfaz gráfica) al que está asociado al pasar un tiempo dado una vez se ha

habilitado. Lo utilizamos para mostrar los mensajes de texto de información previa, avisando al jugador cuando debe comenzar a disparar. Va asociado a GameObjects de texto.

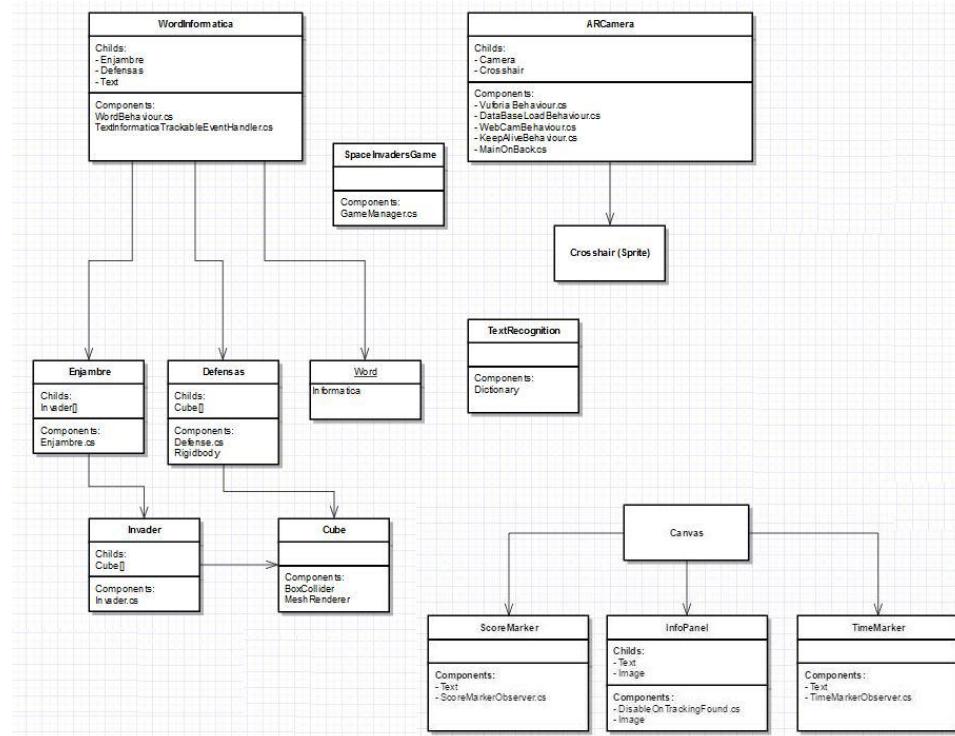


Figura 4.3: Diagrama de los GameObjects

## 4.5. Conclusiones

Al detectar el texto de FACULTAD DE INFORMÁTICA, el GameManager es avisado y comienza el juego. El GameManager instancia el Enjambre, que a su vez instancia todos los Invaders. Una vez hecho esto, se instancian las defensas desde el GameManager, se activa la información de la interfaz de usuario (puntuación, punto de mira ...). A partir de este momento, el jugador puede disparar el láser.

Ahora, en cada actualización, el Enjambre mueve su posición, moviendo a todos los Invaders con él. Por su parte, cada Invader tiene un tanto porciento de posibilidades de disparar. Si se da esa posibilidad, instanciará un proyectil en la dirección de las defensas.

Durante el desarrollo de este juego hemos encontrado unos cuantos escollos que superar, algunos que nos han llevado quebraderos de cabeza. Comenzamos desarrollando el videojuego utilizando un código QR como marcador de la RA, pensando que después pasar a utilizar un texto no tendría complicaciones. Una vez teníamos el juego desarrollado con un código QR (ImageTarget), probamos a detectar texto propio, ya que Vuforia nos da un diccionario con miles de palabras en inglés, pero nosotros no queríamos

detectar esas palabras, sino únicamente FACULTAD DE INFORMÁTICA.

Para esto, seguimos los tutoriales de Vuforia, y, tras resolver algunas dudas, implementamos una escena sencilla en la que se mostraba una esfera sobre el texto. Después, intentamos transferir lo desarrollado con el Image-Target, al texto.

Entonces surgieron los problemas. El primero que vimos, era que las proporciones de nuestros Invasores y las Defensas se quedaron muy pequeñas, haciendo imposible jugar cómodamente. La mejor manera que conseguimos para que se ajustaran los elementos del juego a un tamaño aceptable fue mediante un Script de C# que aumentaba la escala local de los Invasores y las Defensas (local scale).

El siguiente problema que encontramos fue al insertar nuestro Enjambre de Invasores. Por un lado, una vez aparecían los enemigos, si movíamos la cámara, éstos se quedaban en la misma posición respecto a la cámara, es decir, si cuando salían por primera vez estaban en la parte superior izquierda (por ejemplo) de la pantalla, y nos movíamos, seguían ahí, en vez de ajustar su posición con respecto al texto detectado. Tuvimos que cambiar la forma en la que los insertábamos en la escena, antes de manera dinámica y creando una instancia con un Script, y ahora como hijos del GameObject que representa al texto detectado. Éste tipo de problema (de la posición respecto a los objetos de RA) nos volvería a salir más adelante, pero con los proyectiles que lanzábamos para acabar con los enemigos.

En las primeras versiones del juego, lanzábamos un prisma (nuestro Proyectil, similar al que lanzan los enemigos contra las defensas) contra los enemigos. Según lanzábamos, el proyectil se iba dirigiendo en la dirección que tenía la cámara respecto al ImageTarget al realizar el disparo. Al pasar a utilizar el texto, esto cambió. Vimos que nuestro disparo se mantenía siempre en el vector de dirección de la cámara, y cambiaba con éste. Es decir, nuestro proyectil siempre estaba en el centro de la pantalla, con lo cual se perdía toda la gracia al juego y su jugabilidad pasaba a ser bastante complicada. Éste problema nos dejó bastante confusos, ya que con cambiar el objeto de la RA (ImageTarget o TextRecognition) cambiaba el comportamiento del proyectil.

Para resolver esto, cambiamos nuestro proyectil por un láser. Ahora al tocar la pantalla no se lanza un proyectil, si no que se dispara un láser que destruirá los enemigos que estén en el vector de dirección de la cámara. Así hemos conseguido solventar este extraño comportamiento.

El juego en general ha quedado sencillo pero con todos los aspectos que debe tener un juego completo. Sonidos, animaciones, efectos visuales y jugabilidad aceptable. Creemos que sirve como una buena aproximación a otros juegos de este tipo, y que se podría ir escalando para desarrollar un juego más ambicioso.

La complejidad no es grande, aunque sí es recomendable que se tenga

cierta habilidad para apuntar con el móvil.

# Capítulo 5

## Arkanoid

Para el comienzo de este juego, el usuario proviene de defender la facultad (nave espacial tfg-III) de unos invasores y tras finalizar, recibirá una misión, que es la de despejar el campo de batalla para poder despegar la nave. Para ello debe abrir paso reciclando escombros haciendo uso del panel de reciclaje. Éste se sitúa en la tercera planta y se accede a él, apuntando al código QR de Super Mario situado junto a una de las consolas.

### 5.1. Historia

El Arkanoid es un juego de los 80, donde el jugador controla una plataforma que impide que una bola se salga de la superficie que limita el juego Wikipedia (Arkanoid). El objetivo principal de la bola es el de destruir todos los ladrillos o bloques de la pantalla sin salirse de esta. La misión del jugador es la de, no solo impedir que la bola se salga de la escena, sino la de situar la plataforma que maneja de tal forma que consiga que la bola rebote y destruya todos los bloques.

A lo largo del juego puede haber un gran número de variaciones que complican el juego o que facilitan las cosas. La plataforma del jugador puede modificar su tamaño, u obtener mejoras como disparos para romper los bloques. La bola, puede verse modificada mediante variaciones de tamaño o en la física del juego como romper varios bloques sin rebotar o incluso multiplicarse. Y los bloques pueden variar en su posición e incluso desplazarse con el fin de finalizar el juego cuando llegan abajo.

### 5.2. Nuestra versión

Una vez se reconozca el ImageTarget (el QR de Super Mario), aparecerá un viejo televisor, y en su pantalla se observa el juego y dos contadores. El objetivo es sencillo: hay que despejar los escombros. Hay un minuto para

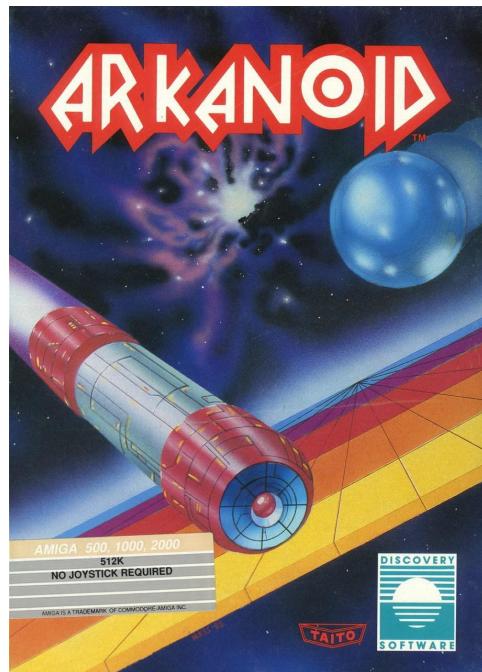


Figura 5.1: Portada original Arkanoid

realizar dicha tarea o hasta que la bola atraviese la deathzone. Cuantos más residuos recicla, mayor será la puntuación que arrastrará al siguiente juego.

Una vez en situación, mientras enfoca al target para no perder de vista la escena, el jugador deberá arrastrar su dedo por la pantalla de su smartphone para poder desplazar la plataforma y controlar donde rebota la bola. Mientras hace eso, la bola irá rebotando y el tiempo decrementando. A medida que destruya objetivos, la puntuación crecerá y el tiempo se irá agotando.

El juego está ambientado en un entorno espacial, haciendo referencia al hilo del juego pero solo de forma superficial. Lo que conforma el tema del espacio es lo que aparece en la pantalla del televisor que contiene la escena. Pero, en realidad, la escena tiene una connotación retro, donde un antiguo televisor al lado de una de las consolas que en un pasado ejecutaba el juego, quiere recrear la forma primitiva de jugar al Arkanoid; en un televisor de tubo y con un juego sencillo que era para lo que daba de sí la tecnología y los medios de la época.

Con esta recreación lo que se intenta es no tanto simular un juego entretenido para cumplir con el objetivo de la historia, sino el recrear una escena pasada basada en este juego. De esta forma se hace uso de la RA en este caso, no como medio de entretenimiento a través las mecánicas que proporciona, sino generando una escena para mejorar la experiencia del visitante del lugar, el museo de la facultad.



Figura 5.2: Código QR de Super Mario

### 5.3. Diseño

La escena se compone de los siguientes elementos:

- ARCamera e ImageTarget como elementos principales en una escena de RA. La propia librería de Vuforia nos provee de ellos, solo hay que definir el Image target al que reacciona la escena y añadirlos; del resto ya se encarga su lógica interna.
- Un modelo 3D de un televisor que hace de contenedor de la escena y que no tiene ninguna mecánica asociada.
- La bola, en la que se han implementado dos componentes; uno físico para hacer que esta rebote, y un script cuya finalidad es la de destruir los bloques, o mejor dicho, la de lanzar la animación de destrucción del bloque y finalizar el juego al acabar con todos los bloques, o, al salirse de la zona de juego. Para evitar problemas y por lógica del juego, aunque la escena sea 3D los componentes sobre este juego son en 2D.
  - Ball Material: Es un material de Unity que, junto al componente RigidBody2D, elimina la gravedad en el objeto, configura a este para poder rebotar, consigue que la fuerza aplicada sobre la bola sea infinita, de forma que no pare jamás de desplazarse; y, define las coordenadas en las que puede desplazarse el objeto (x e y) y en las que puede rotar (ninguna).

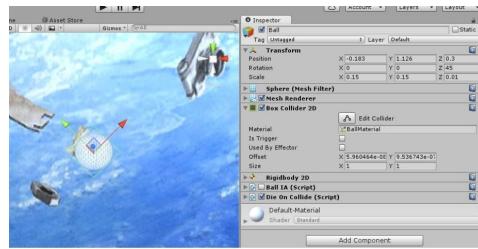


Figura 5.3: BoxCollider Bola Arkanoid



Figura 5.4: Distintos tipos de bloques del Arkanoid

- BoxCollider2D: Para controlar todo el tema de colisiones. En este caso, el Collider que envuelve a la bola es un cuadrado rotado 45 grados para evitar problemas en los rebotes.
- Los bloques, que se reparten por la escena esperando a que la bola colisione con ellos y recibir así, la orden de que se lance su animación de destrucción y se aumente el marcador.
  - Audio source: el componente que da sonido a la acción de destrucción.
  - Animation: la animación de destrucción que se lanza al colisionar con la bola. Esta se genera por interpolación, ya que es el sistema del que nos dota Unity, y lo que hace es reducir la escala del objeto a cero.
- La plataforma: Esta contiene un componente que reacciona a los eventos de la pantalla táctil y que le permite desplazarse, además también tiene un Rigidbody2D cuyas constraints impiden que pueda verse desplazado en el eje vertical o rotado.

- Canvas: Contiene la información del estado del juego. Este objeto y sus hijos, a diferencia del resto, en escena, se muestra en función a la pantalla del dispositivo en el que se ejecuta el juego y no en función de lo que enfoca la cámara. Sus hijos son, el marcador, el contador de tiempo y la pantalla de precarga que muestra la información de la ejecución del juego al usuario.

## 5.4. Implementación

Para el Arkanoid, se han tenido que implementar las siguientes lógicas de juego:

### 1. ScreenDragListener

- Se encarga de capturar las pulsaciones en pantalla. En su método Update () se comprueba con cada llamada si la pantalla ha sido pulsada; si es así, se guarda el punto inicial donde se detectó la pulsación y se guarda, de forma que al volverse a llamar al método, esta vez, no solo se comprueba si ha habido contacto con la pantalla, sino que, además, se comprueba si ha habido variación en la posición del punto.
- El script, tiene una interfaz interna. En el método Start (), se comprueba todos los componentes que implementan dicha interfaz en la escena y se almacenan en un array. Cada vez que se detecta y calcula un movimiento de drag sobre la pantalla del dispositivo, se recorren los componentes del array. Estos reciben, a través del método, las variaciones en los ejes de la pulsación; y así pueden realizar las acciones correspondientes.

### 2. BallIA

- No tiene mucha explicación, aplica una fuerza en dos coordenadas que se le asignan como atributos en el momento en el que está activa y se toque por primera vez la pantalla.

### 3. DefaultTrackableEventHandler

- Es un script que viene por defecto asociado al ImageTarget de Vuforia. En este caso, hemos hecho una leve modificación sobre este script, donde hemos añadido un array de componentes que, cuando se detecta el QR en escena (OnTrackingFound ()) se recorre y estos se van activando.

### 4. DieOnCollide

Con dos parámetros que definen las etiquetas de los gameObjects que se comportarán como enemigo (el gameObject que delimita la death-zone) y los objetivos (los bloques que se destruyen al colisionar con ellos). Implementa el método `OnCollisionEnter2D ()`, que es aquel al que llama el collider 2D cuando entra en colisión el gameObject con otro objeto con collider. Cuando se detecta una colisión, se comprueba la etiqueta del objeto con el que se ha producido y si es de tipo enemigo, se destruye el propio objeto y se pasa a la siguiente escena. Si es de tipo objetivo, se llama a la animación de destrucción de este, se aumenta el marcador y se comprueba si quedan más, para que, en el caso de que no, pasar también a la siguiente escena.

Los mencionados son los más importantes, ya que son los que dotan de lógica al juego, aunque existen más de los que se hacen uso como el `globalGameManager`, que es el encargado de navegar entre escenas, `ScoreMarkerObserver` y `TimeMarkerObserver`, que pintan en el canvas el estado del juego o un script que se añade a la escena para controlar el back de nuestro dispositivo. Además, hay también otros scripts sencillos que controlan el resto de los objetos de la escena.

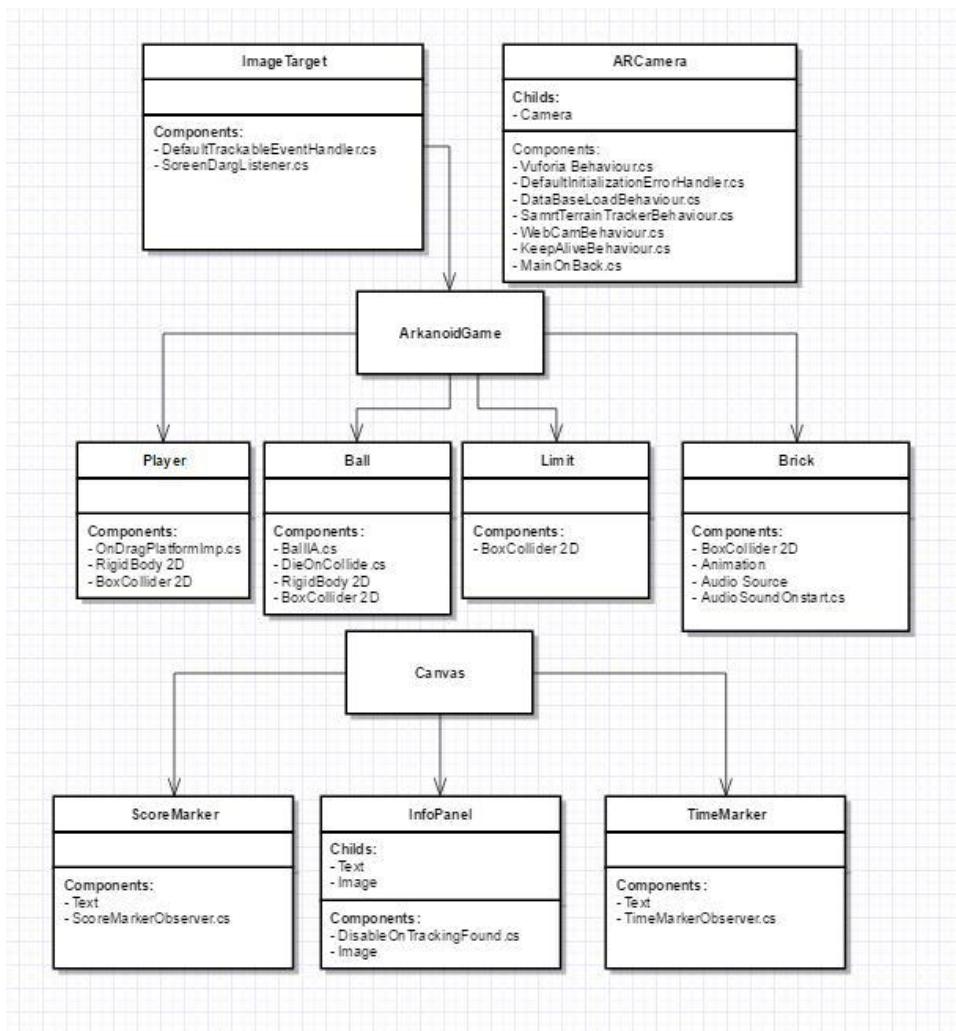


Figura 5.5: Diagrama del juego

## 5.5. Conclusiones

La RA tiene muchos usos como se menciona en el estado del arte. En este caso, el objetivo principal no es el de aprovecharnos de las dinámicas que nos provee para convertirlas en objeto de entretenimiento, sino el de dotar de dinamismo una parte del museo, ya que aportamos información de uno de los objetos expuestos en forma de videojuego (una versión propia que funcionaba en dicha máquina).

Una de las partes complejas en el desarrollo del juego, es la de configurar una forma de jugar en la que se le haga posible al usuario interaccionar con su dispositivo mientras apunta al target. Para esto se han hecho diferentes pruebas y se ha llegado a la configuración actual teniendo en cuenta los

siguientes factores:

- El usuario tiene una posición incómoda al utilizar su Smartphone. Él tiene que apuntar a la imagen mientras interacciona con el teléfono para evitar que la bola caiga. La mejor solución en este caso es la de implementar una mecánica basada en detectar el arrastre del dedo de la pantalla y que sea este movimiento el que desplace la plataforma en el juego. De esta forma, el usuario puede apuntar sin problema al QR mientras a su vez puede jugar. Esta forma de hacer las cosas consigue los dos objetivos, una mantener la cámara fijada en el target, y dos la de hacer cómodo junto con esto el poder manejar la plataforma.
- La complejidad del juego. En este caso se implementa un juego sencillo, muy intuitivo y corto, porque el jugador, por la posición que tiene, no puede pasar mucho tiempo jugando. El juego tiene como máximo un minuto de duración. Y es creemos que es la apropiada ya que en este período se puede jugar sin cansarse y acabar el juego si se tiene la habilidad suficiente sin que el cansancio de la posición evite dicho propósito.
- La escena. Es cierto que el modelo 3D del televisor no ayuda mucho a la jugabilidad. Pero en este caso en el que recrear una escena es casi más importante que la experiencia del jugador en el juego. Se ha preferido penalizar un poco la jugabilidad (tampoco en exceso) para añadir este elemento de forma que añadiendo este objeto a la escena, se haga una referencia a la forma de jugar de la época.

Con este juego respecto al desarrollo de RA con Unity y Vuforia, los conceptos sintetizados son:

- Aunque se trate de una escena 3D el uso de componentes 2D en los objetos de la escena simplifican mucho el trabajo, ya que al no tener que contar con un eje más; controlar el movimiento de la bola, los rebotes y las colisiones es más sencillo. Al fin y al cabo es como mantener el eje restante en un objeto 3D bloqueado pero esto lo hace por ti, por lo que te ahorras muchos quebraderos de cabeza a la hora de controlar casuísticas que se pueden escapar por ser detalles tan pequeños.
- El manejo de los componentes Collider y RigidBody que en esta escena hay que modificar el material por defecto para cambiar la lógica de la gravedad, el rozamiento, etc.; que en este caso, no es el que está por defecto, ya que en este juego no hay nada de eso. Además hay que hacer uso de las constraints del RigidBody para poder bloquear las rotaciones y desplazamientos de algunos de los objetos de la escena que no interesan que puedan moverse.

- La depuración del juego en modo escena en vez de hacer uso de la cámara. En Vuforia es normal el uso de una webcam para probar tu escena, pero en este caso, para probar el juego, era incómodo el tener que estar usando dicho elemento, por lo que, desactivando dicha opción de depurar la webcam en el ImageTarget colocando la ARCamera apuntando a nuestro modelo 3D podíamos depurar simplemente haciendo click en el play del IDE de Unity sin preocuparnos de apuntar al QR. Lo malo de este planteamiento es que traía un problema consigo, los eventos de pantalla; por defecto, Unity no detecta como eventos de pantalla en un dispositivo Android el uso del ratón. Para solventar esto se hizo uso de una aplicación llamada UnityRemote, que lo que hace es mostrar en la pantalla de nuestro dispositivo la escena que se está ejecutando en el IDE; y de esta forma podemos capturar los eventos que se realizan sobre esta pantalla.



# Capítulo 6

## Water Pipes

En este capítulo, vamos a centrarnos en la explicación del último minijuego al que se tendrá que enfrentar el jugador para completar con éxito la yincana. El recorrido finaliza en la 3º planta de la facultad, en un “simulador básico de centrales térmicas”. Este simulador y el minijuego combinan a la perfección, tanto por la similitud visual entre ambos como por el hilo argumental del juego.

### 6.1. Historia

Este juego fue creado a finales de los años 80 bajo el nombre de “Pipe Mania” por “The Assembly Line” Wikipedia (Pipe Mania), teniendo muchas versiones a lo largo de los años. Las primeras fueron realizadas por el estudio “LucasFilm Game” que lanzó el juego “Pipe Dream”.

Aunque posteriormente fueron lanzadas otras versiones del juego para PC, PS2, NintendoDS y PSP.

En la versión original, el juego consistía en ir colocando las tuberías que salían de forma aleatoria en un tablero de tal manera que el agua pudiera fluir por ellas. El objetivo era construir el mayor recorrido posible antes de que el agua lo inundara todo. Cada tubería llena de agua sumaba puntos y cada tubería que no hubiera sido inundada a la finalización del juego restaba puntos para obtener así la puntuación final. En cada nivel se iban complicando las cosas. Esto lo conseguían poniendo posiciones del tablero de juego en las cuales no se pudieran colocar tuberías, o comenzando antes a fluir el agua por las tuberías, dando al jugador menos tiempo de reacción.

### 6.2. Nuestra versión

En nuestra versión del juego existen algunas diferencias con respecto a la versión original. La principal es la mecánica del juego, ya que en esta ocasión

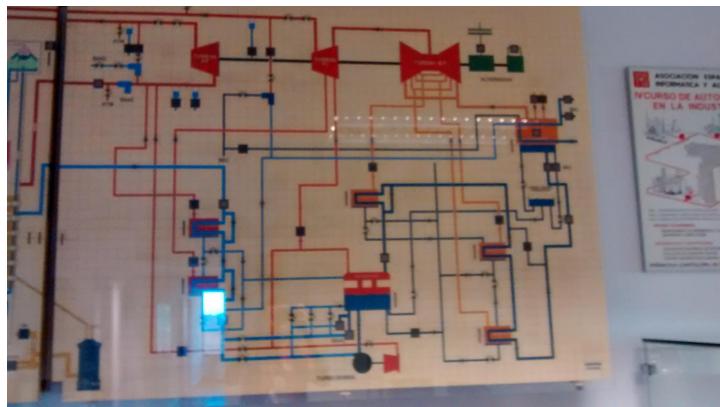


Figura 6.1: Simulador térmico del museo García-Santosmases

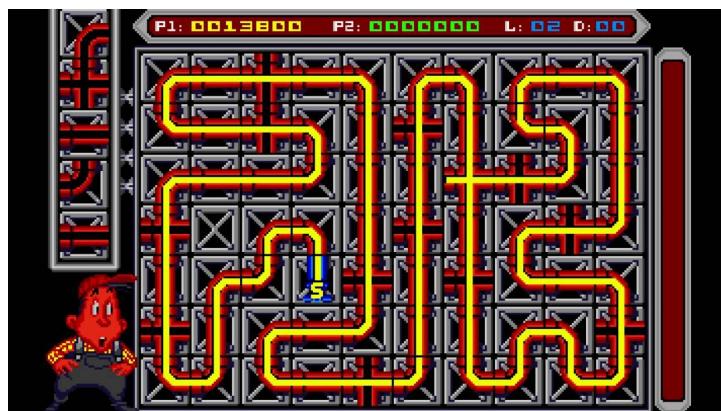


Figura 6.2: Juego original de 1989

el usuario acaba de exterminar al enemigo que estaba poniendo en peligro a la Facultad y ahora su misión cambia para poder despegar la nave y ponerse a salvo definitivamente. En esta última misión el jugador se encontrará frente a frente con los conductos de refrigeración de la nave. El problema es que, debido a la batalla, estos se han dispersado y si quiere despegar la nave tendrá que re ordenarlos para que el agua pueda fluir correctamente. Al comienzo, el jugador verá un tablero lleno aleatoriamente con estos conductos, además de una salida y una meta. Por lo tanto, el objetivo del jugador es ir cambiando de posición las tuberías que encuentra en la matriz, entre sí, para que el agua pueda fluir del punto de origen al de destino. En esta ocasión, el jugador tiene 6 segundos por cada tubería para ir colocando las demás y otros 10 segundos adicionales desde que el juego comienza hasta que la casilla de salida comienza a llenarse. Para poder despegar la nave, el agua ha debido fluir por todas las tuberías necesarias para llegar a la meta. En cambio si

el agua ya ha comenzado a fluir y el jugador no ha conseguido recolocar los conductos, en cuanto el agua no encuentre un camino factible la nave ya no se podrá despegar y el juego terminará.

Para lograr esto, el jugador deberá ir intercambiando una tubería por otra. La mecánica implementada para permitirle realizar esta acciones es que, en primer lugar, seleccione la tubería que desea colocar en otra posición pulsando sobre ella y, a continuación seleccione la otra tubería que desea intercambiar con que ya ha seleccionado anteriormente, y en el momento en el que se pulse esta última, automáticamente se cambiarán entre ellas. Las tuberías en la que ya ha comenzado a fluir el agua no podrán ser seleccionadas.

### 6.3. Diseño

En este apartado explicaremos todos los objetos y componentes necesarios para la implementación del minijuego.

Para este juego existe un solo tipo de objeto visible, los tiles que representan a los conductos. Todos estos objetos, se crean dinámicamente al comienzo de la escena, como hijos del Image Target.

- Tile: todo el juego se compone de 25 (matriz de 5x5) GameObjects de tipo Quad que representan los conductos de refrigeración. A cada uno se les han incluido los mismos componentes:
  - Box Collider: para detectar la colisión y así poder intercambiar dos objetos entre sí. En este caso, solo existe la colisión entre el usuario y los objetos (Touch).
  - RigidBody: Elimina la gravedad de los objetos y congela ciertos movimientos de desplazamiento y rotación de estos. En este caso hemos querido bloquear todas las rotaciones y la dimensión Z del desplazamiento. Esto es debido a que aunque es un juego en 3D, la experiencia con el jugador es totalmente en 2D y esos movimientos no nos serán necesarios.
- Tile animación: Aunque el único objeto “visible” en el juego sean los conductos (Quad), cada uno de ellos contiene otro GameObject como hijo. La funcionalidad de éste es la animación del agua cuando pasa por la tubería, por lo que cada uno de estos “hijos” contienen también los mismos componentes cada uno:
  - Sprite Renderer: Esto es necesario, ya que la animación se compone de un sprites.

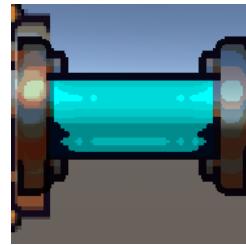


Figura 6.3: Uno de los 23 tiles que forman el tablero

- Animator: que contiene el Animator Controller de ese tipo de tubería. Cada controlador suele tener dos animaciones por tubería, que dependiendo de la salida de la anterior se reproduce una animación u otra.
- Canvas: Muestra la información del juego, y se encarga de mostrar el tiempo restante que le queda al jugador y cuando el juego finaliza muestra la pantalla de marcadores, para que el jugador pueda introducir su nombre y así entrar en la lista de clasificados.
- GameManager: Es un GameObject vacío en la escena que contiene los scripts para la funcionalidad del juego:
  - GameManagerWaterPipes.cs: es el encargado de generar toda la escena al inicio del juego.
  - WaterController.cs: desde este script, se controla todo el flujo del agua.
  - TimeController.cs: clase que decrementa el tiempo de juego y el tiempo que tarda el agua en pasar por cada conducto.

## 6.4. Implementación

Este juego se ha implementado en base al patrón command, que encapsula la decisión de por dónde va a ir fluyendo el agua. Esta funcionalidad es la principal de todo el juego, ya que aunque el jugador de lo que se encarga es de intercambiar las tuberías, el objetivo del juego es conseguir que el agua fluya desde la casilla de salida hasta la meta.

En primer lugar, vamos a describir las clases que se encargan de las otras funcionalidades del juego, las cuales son, el intercambio de tuberías que realiza el jugador, la inicialización y creación del entorno del juego y el tiempo del juego.

### 1. TouchObject

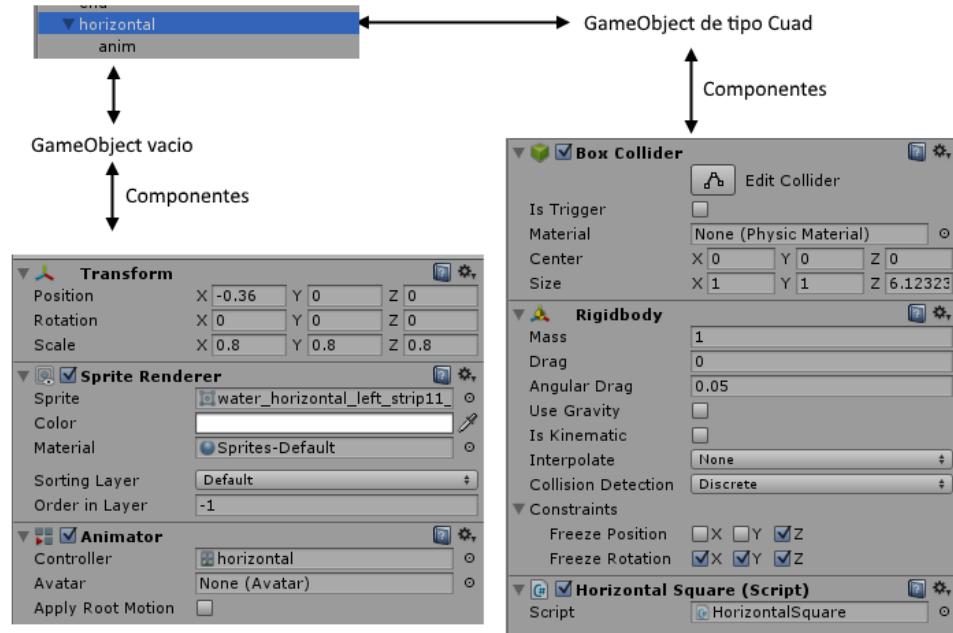


Figura 6.4: Esquema de los GO y sus componentes, necesarios para el tile anterior

Este script es el encargado de controlar el cambio de los conductos. Para ello, lo primero que hace es capturar en el método Update todos los toques en la pantalla y llamar a getObjectHit para obtener el objeto presionado. Este objeto lo obtenemos a través del método “GetNearestHitGameObject”, que genera un rayo con origen en la ARCamera y con la dirección generada por el rayo. De esos se queda con el que haya menor distancia.

Una vez tenemos el objeto presionado, tenemos que saber si ya existe otro objeto pulsado para poder intercambiarlos, o si es el primer elemento que queremos cambiar. Si es el primero, lo único que haremos es cambiarle a ese objeto el tag a “ButtonPush” y si es el segundo, obtenemos el que ya hemos pulsado anteriormente e intercambiamos su posición.

Este componente se encuentra en el objeto ARCamera que nos proporciona Vuforia.

## 2. GameManagerWarterPipes

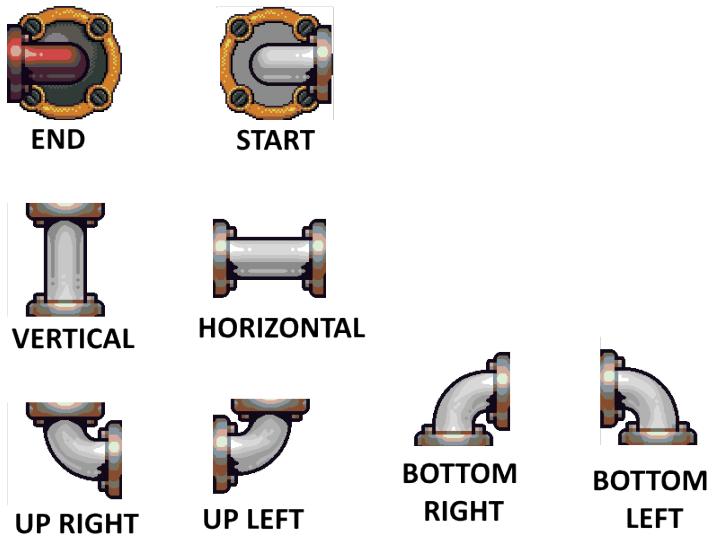


Figura 6.5: Sprites de las tuberías del juego Water Pipes

Es el encargado de crear dinámicamente todos los objetos de la escena al comienzo del juego, y de llevar el control del tiempo que tiene el jugador para completar la misión. Este componente, se añade al GameObject vacío de nombre GameManager de la escena. El script lo primero que hace es añadir los objetos prefabs de cada tipo de conducto (Horizontal, Vertical, Codo Derecho Arriba, Codo Derecho Abajo, Codo Izquierdo Arriba y Codo Izquierdo Abajo) en una lista de SquareReceiver para luego ir eligiendo de manera aleatoria los 23 conductos totales.

A continuación crea la salida y la meta, ya que estos siempre van a tener una posición fija en el tablero. Y cuando ya está todo listo, lo único que falta es ir creando el resto del tablero. Esto lo hace recorriendo todo el tablero desde la casilla 1 hasta la 23 y por cada una escoge aleatoriamente un tipo de conducto, de la lista anterior. Crea una instancia de ese tipo de objeto que será un hijo del Image Target. Dándole además las propiedades como la posición, el nombre, y la escala. Esta instancia la guardamos en un array del tipo SquareReceiver para luego poder acceder a él. El método Update controla el tiempo del juego.

### 3. WaterController

Este script es, junto con el GameManagerWaterPipes, el más importante ya que se encarga de controlar el flujo del agua. Esta es la funcionalidad básica de nuestro juego, y por lo tanto, está añadido al GameObject vacío de la escena, que como hemos dicho en el punto



Figura 6.6: Captura del tablero generado de forma aleatoria

anterior contiene también el script `gameManagerWaterPipes.cs`.

En este ocasión, se ha decidido diseñar el flujo de agua de la siguiente manera:

Por cada tipo de conducto el agua puede fluir en dos sentidos cada vez, por lo tanto a partir de la entrada del agua de la casilla anterior se calculará la casilla siguiente por la que el agua debería de fluir. Y una vez que ya sepamos cuál es la siguiente casilla por la que tenemos que llevar el agua, podremos comprobar si esa casilla es válida, es decir su entrada de agua coincide con la salida de agua anterior.

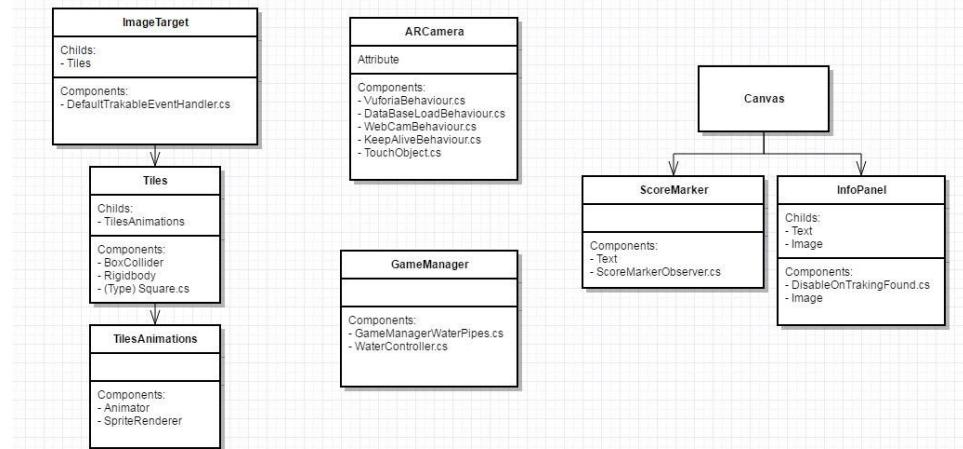


Figura 6.7: Diagrama del juego

## 6.5. Conclusiones

En este caso, el juego Water Pipes es un juego sencillo que la mayoría de la gente conoce, por lo menos en alguna de sus versiones, y que entretiene al usuario. Al tener varias versiones del juego fue interesante decidir cuál era la que mejor se podía adaptar a la RA. Quizá no es el juego que más llame la atención del usuario con respecto a este punto, sobretodo en las primeras fases de la implementación cuando no estaban introducidas las animaciones del agua. Pero una vez, terminado el juego la RA hace más llamativas estas, por lo que atrae más la atención del usuario, lo que era nuestro principal objetivo al empezar este proyecto.

Creemos que el resultado final cumple con los objetivos, ya que se ha implementado un juego que gracias a su sencillez y que la duración no es demasiado larga, entretiene y capta la atención del usuario.

## Capítulo 7

# Detalles de implementación

En esta sección se van a exponer detalles técnicos del desarrollo que sirva para completar todos aquellos puntos que no se tratan en los capítulos de los otros juegos, pero que han sido necesarios para el desarrollo del proyecto.

### 7.1. Realidad aumentada

Vamos a ver los diferentes aspectos y herramientas de la realidad aumentada que hemos utilizado.

#### 7.1.1. Vuforia

Vuforia es el framework que hemos utilizado para hacer toda la parte de RA. Este tiene una magnífica integración con Unity 3D y facilita muchísimo su desarrollo.

Para generar RA, Vuforia superpone a la imagen tomada por la cámara de, en este caso, nuestro Smartphone, cualquier modelo en tres dimensiones que queramos sobre la posición de un Image Target (u otro marcador) que le hayamos indicado. De esta manera, tenemos un “fondo” con la imagen tomada por la cámara, con modelos en tres dimensiones “por encima”. Además, nos mantiene siempre los objetos de tres dimensiones en el mismo punto del espacio, por lo que si movemos nuestra cámara, cambiará la perspectiva desde donde vemos el objeto, pudiendo girar alrededor de éste. El comportamiento puede ser diferente, dependiendo de cómo lo hayamos configurado (podemos hacer que el objeto persista aun que perdamos de vista el detector).

Vuforia proporciona paquetes para trabajar directamente con el SDK de Android o el de iOS, así como para Unity3D. Utilizando Unity3D po-

demos exportarlo después a una aplicación de Android o iOS también, aunque no quedaría de una manera tan “pulida” como desarrollándola directamente con el SDK del sistema operativo deseado. Nosotros hemos decidido utilizar el paquete para Unity3D, ya que es un motor de videojuegos y los tres teníamos unos conocimientos básicos en desarrollo con Unity, además de que nos permite exportar después el proyecto al sistema operativo que quisiéramos..

### 7.1.2. Unity

Unity Unity3D funciona con algo a lo que han llamado escenas, que son diferentes situaciones o niveles del juego. En toda escena hay una jerarquía de objetos que la componen, y de cada objeto pueden colgar otros objetos, además de que se pueden añadir (por medio de código programable) otros objetos a esa jerarquía de manera dinámica. Todos los objetos de Unity tienen una serie de componentes, el más básico sería el de su situación en las tres dimensiones (o dos), su escala y su rotación con respecto a los tres planos. Estos componentes permiten configurar los objetos de manera sencilla, encapsulando funcionalidades. Esta forma de “componer” los objetos no es casual: es la más utilizada en programación de videojuegos.

Además, Unity cuenta con una extensísima comunidad de desarrolladores, así como tutoriales, guías, dudas resueltas... solo con los tutoriales que proporciona la propia gente de Unity podemos hacer un sencillo juego casi de cada uno de los tipos más comunes de juegos.

Unity nos proporciona por defecto el cálculo de colisiones entre objetos, gravedad, eventos de teclado o ratón... en pocos minutos podemos hacer cosas sencillas pero que con otras herramientas, o programándolo directamente a mano con un lenguaje de programación cualquiera como podría ser Java o C++, nos llevarían bastante más tiempo.

Los scripts los podemos escribir en C, Boo o un lenguaje “parecido” a JavaScript. Nosotros hemos decidido utilizar C#, ya que era la opción que más nos convencía por varias razones:

- Es más eficiente. Los tres teníamos conocimientos previos de Java, y C# es muy similar a Java en cuanto a sintaxis.
- Es el más usado por la comunidad.

Todos los Scripts utilizados en Unity heredan de la clase MonoBehaviour, la cual permite a estos scripts integrarse con la ejecución interna de Unity. Toda clase que herede de MonoBehaviour tiene los métodos Start (), Awake (), Update (), FixedUpdate () , y OnGUI (), entre otros.

Éstos se ejecutan en diferentes momentos del juego.

- Awake (): el primer método al que se llama, antes incluso de que el objeto asociado esté habilitado en la escena. Se utiliza para inicializaciones o referencias entre scripts.
- Start (): se ejecuta después de Awake (), justo antes del primer Update () y después de que se active el objeto.
- Update (): se ejecuta en cada frame. Esto hace que dependa del procesador y del equipo donde se ejecuta. Se usa para actualizaciones comunes como mover objetos no físicos, recoger entrada del usuario...
- FixedUpdate (): el intervalo entre una ejecución y otra es consistente y siempre el mismo. Se utiliza para actualizaciones cómo ajustar objetos físicos.
- OnGUI (): se utiliza para gestionar y renderizar eventos de la Interfaz Gráfica de Usuario (Graphic User Interface, GUI). Sólo es llamada si el objeto está habilitado.

### 7.1.3. Unity + Vuforia

Vuforia nos proporciona un paquete de extensión de Unity 3D el cual debemos importar para trabajar. Éste paquete contiene diferentes prefabs (objetos ya construidos) que nos harán la tarea muy sencilla.

Lo que debe tener toda aplicación de RA hecha con Vuforia y Unity 3D es una ARCamera (cámara de RA). A ésta hay que indicarle el product key que nos da Vuforia desde su portal para desarrolladores, además de ésto, se le indicará el paquete de targets propios . Es la unidad mínima de desarrollo de RA con Vuforia.

Una vez hecho esto, tendremos diferentes opciones para lanzar los objetos de RA, que deben colgar en la jerarquía de Unity de cualquiera de los siguientes prefabs Vuforia (a):

- Frame Markers: Son marcadores muy sencillos que son proporcionados por la gente de Vuforia en su paquete. Se pueden utilizar para calibrar la cámara, pero no tienen una gran calidad a la hora de ser detectados. Son lo más sencillo para comenzar una aplicación de prueba.
- Image Targets: Imágenes propias del desarrollador. Funcionan como los Frame Markers, pero éstas deben ser importadas desde un paquete generado por el portal de desarrolladores de Vuforia, el cual nos indicará la calidad de esa imagen para ser detectada.
- Multi-Targets: Son varios ImageTargets que representan las diferentes caras de un prisma en tres dimensiones.



Figura 7.1: Ejemplo de aplicación con Smart Terrain

- Cylinder Targets: ImageTarget que envuelve un cilindro, para representar, por ejemplo, una botella u otro objeto similar.
- Text Recognition: Nos permite detectar textos, ya sean del diccionario proporcionado por Vuforia de palabras en inglés (más de 100.000 palabras diferentes) o de uno creado por nosotros mismos.
- Object Recognition: Sirve para configurar un objeto en tres dimensiones que no sea ninguno de los anteriores.
- Smart Terrain: Permite reconstruir el entorno del usuario de la aplicación en tres dimensiones.

Con cualquiera de estos objetos, la funcionalidad por defecto (que podemos modificar creando nuestras propias clases que hereden de las que nos da Vuforia) es que al detectarse (ya sea un ImageTarget, un Text Recognition, etcétera) se comenzarán a mostrar todos los objetos que cuelguen de él en la jerarquía de Unity.

## 7.2. Escenas intermedias

Entre cada escena; o juego del desarrollo, se ha establecido una intermedia donde un humanoide nos pone en situación antes de cada misión y nos sitúa acerca de cómo se ha llegado a dicha situación y a dónde debemos ir para resolverla.

### 7.2.1. SALSA with Random Eyes

Gracias al paquete de Unity de animación por voz, SALSA lip sync, el proceso de sincronizar el audio con la animación del discurso del modelo ha sido algo muy sencillo Crazy Minnow Studio. Dicha animación consta de tres componentes:

- El modelo, que en este caso era un prefab que venía configurado por defecto para soportar el componente de audio y de enfoque aleatoria.

- Componente de animación de los músculos faciales, al cual se le asigna un conjunto de audios de tal forma que la cara del modelo se articula de forma sincronizada con el audio proporcionado. El componente viene configurado para que el audio que le proporcionamos al modelo pueda ser interpretado por este con más o menos énfasis o con diferentes estados de ánimo.
- Random eyes es un componente que, asignado al modelo, articula sus ojos de tal forma que definiendo unos puntos objetivo, alterna y gesticula mirando a los diferentes objetivos a lo largo de la animación del objeto.

### 7.2.2. VozMe MP3 generator

Además del modelo utilizado, también hubo que crear los audios que narran el hilo argumental de nuestro juego. Para esta tarea, se generaron los audio con la herramienta vozme.com VozMe que convierte a mp3, con una voz un “robótica”, un texto dado.

## 7.3. Persistencia de puntuaciones

Al finalizar el juego, el usuario puede almacenar su puntuación, y ver el ranking de estas. Este sistema se hizo para enlazar el juego con lo que es un servicio web, ya que nos parecía una práctica interesante el hacer uso de la parte cliente que nos ofrece Unity para comunicarnos con servicios web. Se han implementado ambas partes en este caso, tanto el lado cliente que consume los servicios en Unity, como el lado del servidor, en el que se ha implementado un sistema de gestión de usuarios con una sencilla api REST en PHP, haciendo uso del framework Symfony 2.

### 7.3.1. Symfony

Para el desarrollo con symfony de una API REST se tuvieron en cuenta los siguientes puntos:

- Base de datos: Se genera una base de datos SQL con una sola tabla, la de usuarios y los campos que deseábamos guardar: id, nombre y puntuación.
- Operaciones: Para la parte REST; los servicios, solo pueden realizar las operaciones de leer y crear nuevos usuarios. En el panel de administración, se pueden realizar las CRUD, crear un nuevo

usuario, leer los usuarios y ver el usuario en detalle, actualizar y eliminar.

- REST: las operaciones de la API son, de tipo GET /api/users.[json | xml] para obtener un listado de los usuarios y sus puntuaciones en uno de los formatos especificados. Y para guardar la puntuación de un usuario, de tipo POST /api/users dónde se les manda, en el body de la petición, la puntuación del usuario y el nombre de este.
- Seguridad: De lado del servidor, Symfony nos provee de un sistema de seguridad basado en tres factores; la dirección a la que se accede, la autenticación para acceder a ella y una vez autenticado, el rol que tiene dicho usuario para poder acceder. En nuestro caso hay dos usuarios. “admin”, con el rol de administrador, que le permite entrar tanto en el panel de administración, como hacer uso de la api. Y “api\_user”, un usuario cuyo rol solo le permite usar los servicios web. El tipo de autenticación es a través de Basic auth. Consiste en añadir un campo en el header con la clave “Authorization” y como valor, la palabra “Basic” concatenada y separados por un espacio con la combinación de “username:password” codificada en Base64.

La implementación del sistema de persistencia se compone de dos partes, la parte del cliente, dónde se consumen los servicios web; y la parte del servidor, donde se configura el sistema.

El lenguaje utilizado para implementar la parte del lado del Servidor es PHP. Sobre este lenguaje se utiliza un framework de desarrollo llamado Symfony, que facilita mucho la tarea de construir este tipo de sistemas. Symfony hace uso de trabajar basada en el Modelo Vista Controlador y el flujo de trabajo sobre este framework consiste en simplificarte, bajo esta arquitectura, un montón de procedimientos y líneas de código que se repiten constantemente en este tipo de software.

## Capítulo 8

# Evaluación con usuarios

Una vez hemos tenido una versión terminada de todos los juegos y unidos ellos entre sí mediante una pequeña historia, damos por finalizada la primera versión estable de nuestro juego. Con esta versión, decidimos realizar pruebas con usuarios para mejorar los posibles errores o mejorar la usabilidad de los juegos de cara a una segunda iteración.

### 8.1. Plan de evaluación

#### 8.1.1. Objetivos de la evaluación

Nuestros objetivos a la hora de realizar la evaluación con distintos usuarios son: mejorar la usabilidad de cara al usuario, detectar posibles bugs, comprobar si las mecánicas son comprensibles para los usuarios del museo y ver si estas son entretenidas y coherentes de cara a la RA.

Cuando se desarrolla un juego, el programador realiza distintas pruebas durante toda la implementación. Y una vez que ha terminado el juego, intenta jugar de varias formas posibles con la única intención de encontrar el mayor número de errores posibles para luego solventarlos. Pero esto no suele ser suficiente debido a que si realiza las pruebas la misma persona que lo ha implementado, pasará por alto muchas cosas. Esto es debido a que él sabe el funcionamiento del juego e inconscientemente jugará bien y le resultará muy difícil darse cuenta de cierto tipo de errores. Un buen ejemplo de esto es, que si el juego es intuitivo, es decir, si aunque no sepas como funciona, está lo suficientemente bien explicado y diseñado para que el usuario no tenga problemas de ese tipo.

Otro motivo por el cual pensamos que la realización de pruebas a distintos usuarios es importante, es para tener la información suficiente,

N	Preguntas	Puntuación
1	Me gustaría volver a jugar	5
2	El juego tiene una complejidad innecesaria	1
3	Pienso que el juego es fácil de jugar	5
4	Necesitaría el apoyo de una persona que conozca el juego para poder jugarlo	1
5	Creo las funcionalidades están bien integradas en el juego.	5
6	El sistema tenía demasiados fallos.	1
7	La mayoría de las personas podrían aprender rápidamente a jugar.	5
8	El juego es incómodo de jugarlo.	1
9	Me sentí agusto jugando.	5
10	Necesitaría haber aprendido algunas cosas antes de jugarlo.	1
		100

Tabla 8.1: Tabla de cuestionario SUS

como para poder evaluar los errores en y clasificarlos por prioridad. Después de realizar las suficientes pruebas, se van a presentar varios errores, pero no tendrán la misma complejidad a la hora de solucionarlo y sobre todo no suponen lo mismo para la jugabilidad de la aplicación.

Creemos, que la evaluación con usuarios es una fase fundamental en cualquier aplicación. Cuantas más pruebas se realicen, la aplicación será mejor y la experiencia del usuario con el juego será más satisfactoria, lo que seguramente hará que ese usuario quiera repetir el juego y se descargue tu aplicación

### 8.1.2. Métricas

Al ser una aplicación, decidimos utilizar el cuestionario SUS para realizar estas pruebas. Pero este cuestionario está en un principio dirigido a evaluar una aplicación web, por lo que tuvimos que modificar un poco las preguntas para adaptarlo a un juego. Además, aparte de este, mientras el usuario juega, se toma nota de todo aquello que se puede observar importante, ya sean bugs o posibles mejoras y aquello que el usuario menciona mientras está jugando: Cosas que no comprende, observaciones que aprecia acerca de la dificultad de los juegos o de sus mecánicas.

La siguiente tabla muestra las preguntas finales, con las respuestas necesarias para sacar el 100 % de la puntuación.

Una de las ventajas de este cuestionario, es que las preguntas son sencillas y fáciles de entender. Pero también, es muy importante que sea muy sencillo de contestar, ya que el jugador, solo tendrá que evaluar cada pregunta en un intervalo del 1 al 5; siendo 1 la respuesta elegida si está muy en desacuerdo con lo que le preguntan, o 5 en el caso de que esté muy de acuerdo.

## 8.2. Descripción de la metodología del análisis de los datos

Dividimos en varias fases este proceso, con la intención de simplificar cada una de ellas y así tener muy claro lo que debíamos hacer en cada momento para que no se nos escapará ningún detalle, obteniendo así, no el mejor resultado en los test, sino lo más importante datos e información relevante para poder mejorar nuestra aplicación

### 8.2.1. Primera fase

Lo primero que hicimos, una vez adaptamos las preguntas del cuestionario a las de un juego, fue decidir sobre qué nos deberíamos fijar más detenidamente mientras realizamos los test y así poder tomar notas sobre ello para más tarde poder añadirlo a la evaluación final, ya que cuanta más información se obtenga de los test a los usuarios, más mejoras podremos realizar en nuestro juego.

Cuando tuvimos claro todos estos puntos, empezamos a realizar test a la mayor cantidad de usuarios posibles, sin importarnos su edad, su habilidad a la hora de utilizar un smartphone o si habían probado previamente los juegos que hemos implementado.

### 8.2.2. Segunda fase

El siguiente paso es la evaluación directa con los usuarios. Nuestro procedimiento a seguir en esta fase, es la de que el usuario juegue e intentando no explicarle nada del juego previamente analizar mientras juega y decirle que comente en voz alta sus observaciones. Además, mientras el usuario está jugando, nosotros tomamos notas que nos sirven de ayuda. Y una vez el usuario ha finalizado, se le pasa el cuestionario para que responda a las preguntas y anote su edad.

### 8.2.3. Tercera fase

Analizamos los resultados, tanto los obtenidos a través del cuestionario SUS, como de las notas recogidas durante la prueba del juego.

## 8.3. Primera evaluación con usuarios

En esta primera evaluación, efectuamos un total de 14 pruebas a los usuarios, de las cuales, 7 se realizaron en la facultad y el resto desde casa.



Figura 8.1: Usuario realizando un prueba del Space Invaders

Tras realizar la evaluación nos encontramos con que la mayoría de los usuarios tenían problemas similares:

- El space invaders, aunque era el más intuitivo, se acaba muy rápido y era demasiado fácil de jugar.
- En el Arkanoid, la bola iba demasiado rápida, el dedo que controlaba la barra tapaba la escena y no les permitía ver bien, el control era poco preciso y el fondo no permitía distinguir bien cuáles eran los objetivos que había que destruir.
- En el Water Pipes, la gente no comprendía la mecánica del juego al iniciar. Pulsaban sobre una pieza y no comprendían si tenían que intercambiar las piezas. Además, les costaba comprender cuál era el comienzo y cuál la meta.

### 8.3.1. Informe de resultados

#### Cuestionario SUS

El análisis de resultados de la primera evaluación con usuarios fue muy revelador con respecto a los controles del usuario con los distintos juegos. Tanto como los resultados del cuestionario SUS, como los comentarios de los usuarios unidos con nuestras notas tomadas mientras jugaban, nos revelaron que en los dos últimos minijuegos de la aplicación era necesario realizar alguna modificación para que el usuario pudiera manejar los controles del juego de manera más efectiva.

Es importante saber que el cuestionario SUS coloca el resultado obtenido en un intervalo de puntuación que determina la usabilidad de la aplicación y en nuestro caso de los mini juegos.

Este intervalo es el siguiente:

Space Invaders	Arkanoid	Water Pipes
78,2	52,5	55,5

Tabla 8.2: Promedio primera evaluación

- Puntuación  $> 80,3$  : Buena usabilidad
- $80,3 \geq$  Puntuación  $> 68$  : Bien, pero se puede mejorar
- Puntuación  $\leq 51$  : La usabilidad no es buena

A continuación mostramos el promedio de los resultados obtenidos de los cuestionarios realizados en este primer análisis.

Simplemente con estos datos, se puede ver a primera vista lo comentado anteriormente. El primer mini juego, según la escala SUS, no obtiene la puntuación necesaria para obtener una buena usabilidad, pero se acerca mucho, lo que nos indica que aunque existen ciertas cosas se pueden mejorar, el usuario piensa que el juego tiene lo necesario para ser jugado sin problemas y a demás que el juego le resulta entretenido.

En cambio, la puntuación de los otros dos juegos se encuentran en el caso opuesto. Aunque no llegan a obtener el 51, límite para que la usabilidad del juego no sea buena, se aproximan demasiado.

### Información recogida de los usuarios

Los resultados que nos ha proporcionado el cuestionario SUS, nos resultan muy útiles para obtener una visión clara de la usabilidad de cada juego, y aunque si analizamos las respuestas más detenidamente, podríamos hacernos una idea de los fallos y los deseos de los usuarios. Nos es imprescindible también tomar nota, tanto de los comentarios de los usuarios antes, durante y después de haber probado el juego, como de las observaciones que nosotros mismos realizamos.

En la primera evaluación, esta información es muy relevante, debido también a que se repite las mismas opiniones o muy parecidas, en la mayoría de los usuarios evaluados.

En las siguientes líneas mostramos las opiniones más frecuentes y relevantes de los usuarios, por cada uno de los juegos en esta primera evaluación.

- Space Invaders
  - a) El juego sale muy pequeño con respecto al Image Target.
  - b) La dificultad del juego es demasiado sencilla.
- Arkanoid

- a) El control del juego es demasiado complicado. La mayoría de los usuarios, no han sido capaces de controlar la barra para mantener la pelota a salvo.
- b) La velocidad de la pelota es demasiado rápida, lo que hace que el juego sea demasiado difícil de completar con éxito.
- c) La escena es demasiado grande, lo que obliga al usuario a separarse mucho del QR.

- Water Pipes

- a) La mayoría de los usuarios no saben cómo controlar el juego.
- b) El tiempo de comienzo antes de que el agua empiece a pasar por las tuberías es demasiado corto, por lo que los usuarios pierden antes de poder comprender el funcionamiento del juego.
- c) Las casillas de Start y End, se confunden por lo que el usuario no sabe dónde tiene que empezar a colocar las tuberías antes de que el tiempo acabe.

### 8.3.2. Acciones

Las acciones a emprender atendiendo al análisis de los cuestionarios son las siguientes:

- Space Invaders

- a) Aumentar el tamaño de los invaders.
- b) Aumentar el daño de los enemigos sobre la plataforma, aumentar la velocidad de desplazamientos de estos y aumentar la separación.

- Arkanoid

- a) Informar del control en la pantalla de información del principio e intentar mejorar la precisión del control de la plataforma.
- b) Si se es capaz de mejorar en el apartado uno quizás el reducir la velocidad de la pelota facilita mucho el juego, pero si no, se reducirá la velocidad o se ampliará el tamaño de la plataforma.
- c) Disminuir el tamaño del modelo del televisor.

- Water Pipes

- a) Especificar mejor el control y el objetivo del juego en el panel de información inicial.

- b) Si se mejora en el apartado uno quizás el aumentar el tiempo en este caso facilita en exceso el juego, pero sino, se aumentará el tiempo inicial.
- c) Resaltar de forma más significativa la diferencia entre ambos extremos.

### 8.3.3. Conclusiones

Después de analizar los resultados de los cuestionarios realizados a los usuarios, podemos decir que la conclusión más destacable es que a los usuarios les ha entretenido y gustado mucho más el primer minijuego (Space Invaders) que los dos restantes (Arkanoid y WaterPipes).

Una gran mayoría de los usuarios no han podido llegar a completar el minijuego del Arkanoid con éxito, y ni siquiera han podido comenzar a jugar el Water Pipes. En cambio, el 100 % de los usuarios no han tenido problemas de este tipo con el primer minijuego y todos han logrado completar con éxito el Space Invaders.

Los usuarios piensan que disminuyendo la velocidad de la bola en el caso del Arkanoid, su experiencia a la hora de jugar podría mejorar notablemente, aunque no es la única mecánica que creen que se debe mejorar, piensan que sería la primera que deberíamos solventar.

Los pocos usuarios que han llegado a comprender la mecánica del minijuego WaterPipes, nos han comentado que les ha gustado mucho el juego y que les ha resultado muy entretenido. Esto nos lleva a pensar que en este caso, añadiendo alguna mecánica que indicará más claramente el funcionamiento del juego y la forma de controlarlo, sería suficiente para que la experiencia del usuario mejore notablemente.

En cualquier caso, la experiencia del usuario en general ha sido buena debido a que sus comentarios al completar la yimcana eran positivos y la mayoría querían repetir la experiencia.



## Capítulo 9

# Conclusiones y trabajo futuro

En este capítulo haremos un análisis del trabajo realizado, así como de las decisiones que hemos tomado y los resultados de esas decisiones.

Además, haremos un análisis de posibles trabajos futuros a partir de éste proyecto.

### 9.1. Conclusiones

Como ya hemos comentado en el capítulo 2, la realidad aumentada tiene unas posibilidades enormes en multitud de campos. Desde los puramente lúdicos, como los videojuegos, hasta aplicaciones en ciencia o educativas (como pretende ser éste trabajo). En el campo que nos ocupa, que es la educación, encontramos infinidad de posibilidades. Desde atraer a gente a un museo para divulgar la información de éste, hasta formar a profesionales de la medicina con modelos en tres dimensiones de órganos o a ingenieros para ver cómo funcionan motores, engranajes o sistemas de transmisión en tiempo real.

Al final, lo que se ha realizado es una yimcana basada en un juego, donde el usuario deberá utilizar diferentes mecánicas relacionadas con la RA y los videojuegos para lograr pasarselo. El proyecto consta de tres escenas basadas en pantallas de juegos clásicos (Space Invaders, Arkanoid y Water Pipes) que se verán a través de la RA en diferentes lugares del museo y la facultad. Los tres juegos están unidos a través de un hilo argumental que se explica en las escenas intermedias que sirve para dar coherencia al juego, además de explicar al usuario dónde se va a desarrollar el siguiente nivel. Y para finalizar, un sistema de puntuaciones con persistencia, dejarán constancia de cómo lo ha hecho

cada jugador e incitaran a este a volver a jugarlo para competir por subir en el ranking.

Éste trabajo nos ha hecho darnos cuenta del hecho de que probablemente, si no lo es ya, la siguiente gran revolución en los museos sea la RA. Aporta una cantidad de posibilidades para atraer al público que todavía, creemos, no nos podemos hacer a la idea.

Aplicaciones como ésta en los museos, con una yincana, creemos es muy atractiva. Ya que se consigue que el usuario vaya a distintos puntos del museo y realice acciones determinadas, por lo que podemos acercar contenidos del museo que quizá no fueran en apariencia tan atractivos para los usuarios.

Teniendo esto presente, el objetivo de este proyecto era crear un atractivo para los usuarios del museo a través de la RA y videojuegos más que transmitir información al usuario sobre el contenido del museo. Consideramos que esto lo hemos conseguido. Hemos creado tres minijuegos de una dificultad muy aceptable, cada uno utilizando una manera de interactuar por parte del usuario. Además, se puede realizar el circuito en poco tiempo, y como se obtiene al final una puntuación que podemos comparar con las de otros usuarios, esto dinamiza la experiencia.

## 9.2. Líneas futuras

La verdad es que las posibles líneas futuras son muchísimas. Durante el desarrollo del proyecto hemos tenido muchas ideas que no hemos podido llevar a cabo por falta de tiempo.

Por un lado, con añadir más minijuegos o más niveles a los ya existentes, se podría hacer otra aplicación nueva. También hemos tenido otras ideas de minijuegos. Por ejemplo, un juego al estilo “aplasta topos” de las ferias se podría poner en un cuadro informativo que hay con imágenes de disquetes 9.1. Otro juego, más complejo, que se nos había ocurrido era, de alguna forma, hacer que el jugador tuviera que conectar diferentes salidas y entradas de cables en uno de los primeros ordenadores que hay en el museo.

Otra posibilidad es mostrar información acerca de los equipos expuestos y después hacer un test con preguntas al jugador.

Además, se podría añadir una especie de “Lore” interactivo donde al apuntar a los diferentes elementos del museo, se mostrase información de dicho objeto junto con escenas de personajes emblemáticos de las consolas o animaciones relacionadas con los otros objetos.

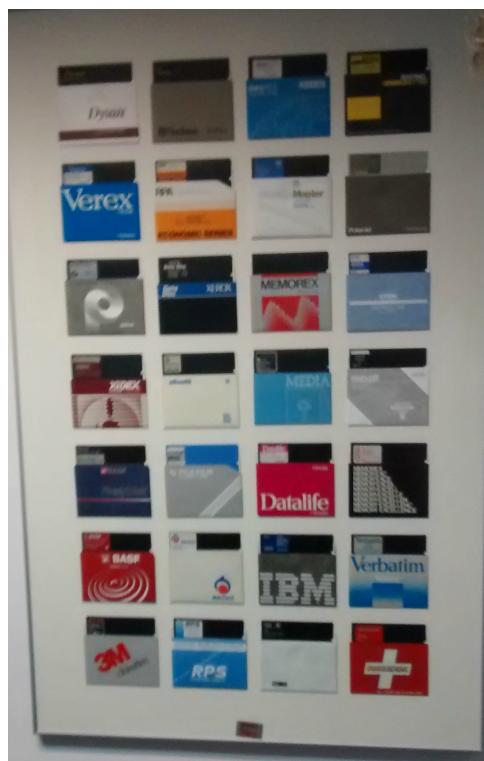


Figura 9.1: Fotografía del cuadro informativo con los disquetes



# Capítulo 10

## Aportaciones individuales

*Trabajar en equipo divide el trabajo y multiplica los resultados*

Anónimo

Lista de aportaciones individuales de cada uno de los miembros del equipo.

### 10.1. Organización general del proyecto

En general nos hemos organizado de manera independiente. Cada uno de los miembros ha realizado uno de los minijuegos, aunque luego hemos desarrollado algunas funcionalidades juntos y otras, a parte del minijuego de cada uno, también de manera individual. Aun siendo cada uno de los minijuegos responsabilidad de uno de los miembros del equipo, nos hemos apoyado cuando teníamos problemas en el desarrollo individual de cada uno.

Para mejorar el trabajo en equipo hemos hecho uso del Sistema de control de versiones Git a través de la plataforma de GitHub<sup>1</sup>. Se trabajaba en una rama de desarrollo sobre la que íbamos integrando aquellos cambios que generábamos sobre el proyecto y otra de producción, donde se registraban y etiquetaban, aquellas versiones finales del juego de cada fase del desarrollo.

---

<sup>1</sup>Repositorio utilizado

## 10.2. Raúl Cobos

El videojuego que he desarrollado es el Arkanoid, cuya implementación detallada se encuentra en el capítulo 5 de esta memoria. Además de la implementación de dicho juego, se realizan las siguientes aportaciones al proyecto.

- Realización de tutoriales en Unity3D para comprender en profundidad cómo funcionan sus escenas y sus mecánicas. Antes de empezar con Unity se realizaron tutoriales y se leyó bastante documentación para comenzar a comprender el funcionamiento de Unity, su arquitectura basada en componentes y la plataforma de desarrollo que este te proporciona. Además se estudiaron los componentes y clases predefinidas que Unity tiene por defecto y que sirven como base para cualquier juego.
- Autoaprendizaje e investigación de nuevas tecnologías para mi como era Vuforia. Ya que requiere un proceso de aprendizaje aparte del de la plataforma, el aprender a utilizar esta herramienta para implementar RA en Unity. Se estudió el funcionamiento del ARCamera y sus componentes que tuvieron que ser modificados en algunos casos como por ejemplo para depurar con la webcam o para sobreescibir algunos métodos como el de detección del ImageTarget para saber cuando comenzaba una escena.
- Realización de prototipos de Unity3D y Vuforia. En el proceso de aprendizaje se desarrollaron prototipos y modelos que al final no se utilizaron en el juego. Además de estos, los que sí se utilizaron fueron algunos como son el capitán de las escenas del argumento; el cual se sacó del paquete de assets que proporciona la extensión SALSA, o el modelo del televisor para el juego del Arkanoid.
- Testeo con Álvar de diferentes maneras de interactuar con la RA en los primeros momentos. Ya que en un principio no se sabía que juegos podrían implementarse, ya sea por la integración con la RA, nuestras limitaciones como principiantes o el tiempo. Se realizó una serie de pruebas, ya sea tratando de implementar funcionalidades para ver si eran viables como buscando posibles juegos que se pudiesen dar dentro del museo que estuvieran acordes con nuestras posibilidades. E investigando, en las distintas zonas del museo y facultad, como podía ser la interacción de ese juego con el usuario así como que juego podría encajar allí.
- Comunicación y reuniones con nuestro tutor Guillermo a través de reuniones presenciales y correos electrónicos. A pesar de nuestro limitado tiempo y problemas de horario, se realizaron bastantes reuniones presenciales con nuestro tutor de proyecto, Guillermo,

para definición de tareas, revisión y verificación. Además se realizaron entregas de contenido vía email para la verificación de algunas cosas del proyecto, además de para resolver problemas. Por otra parte, también hubo numerosas reuniones presenciales entre los compañeros del grupo para desarrollar algunas partes conjuntas y para generar tareas y asignarlas o para resolver entre nosotros mismos algunas cosas que iban surgiendo.

- Escritura de todos los apartados de la memoria (menos los de los minijuegos de mis compañeros) y revisión de los contenidos del a misma.
- Toma de decisiones con el resto de mis compañeros.
- Construcción del entorno de persistencia de puntuaciones. Con el objetivo de hacer la gymkana más atractiva y de motivar al usuario a volver a jugar, se implantó un sistema de persistencia de puntuaciones donde cada jugador puede almacenar su puntuación y consultarla. Esta parte está especificada en el apartado de los detalles técnicos de la memoria y su implementación tuvo parte de desarrollo y parte de aprendizaje del framework Symfony y algunas características especiales como configurar servicios REST y comunicar el juego con estos, además de implementar un apartado de seguridad basado en roles.
- Adaptación del cuestionario SUS. Ya que el cuestionario estaba basado en preguntas en torno a la usabilidad de páginas webs, hubo que traducir y adaptar estas preguntas para que tomarán coherencia con nuestro proyecto.
- Integración de la extensión SALSA With RandomEyes para la animación del personaje de las escenas intermedias. Para que los tres juegos tuvieran algo que los uniese se ha implementado unas escenas intermedias que en el paso entre un juego y otro se explique, a modo de historia, donde se encontraba el siguiente punto para poder jugar al nuevo juego. Para adornar esta escena, se añade un personaje que es quien nos dicta las instrucciones y se le dotó de animación tanto en los labios en sincronización con el audio como en el movimiento de los ojos haciendo uso de la extensión SALSA de Unity 3D. Esto conlleva un aprendizaje para ajustar las animaciones del personaje además de generar los audios para dotar de voz a este.
- Adaptación responsiva de las fuentes de los textos del juego. Ya que el proyecto está basado en una aplicación para móviles Android, y estos tienen muchas diferentes resoluciones con diferentes densidades de pantalla provoca que se tenga que ajustar los textos de forma especial para que puedan verse en diferentes dispositivos con una fuente aceptable, de tal forma que en unos se viese lo

suficientemente grande sin que en otros no se montara y pudiese verse al completo.

### 10.3. Álvar D. Soler

El minijuego que yo he desarrollado ha sido el Space Invader. Además de todo este minijuego, he llevado a cabo las siguientes tareas:

- Realización de tutoriales en Unity3D para comprender en profundidad cómo funcionan sus escenas y sus mecánicas. Antes de comenzar el desarrollo del proyecto tuve que aprender Unity3D en profundidad, ya que solo tenía conocimientos de un taller impartido durante la Semana de la informática en la facultad. Seguí los tutoriales que el equipo de Unity proporciona en su web.
- Autoaprendizaje e investigación de nuevas tecnologías para mí como era Vuforia. Una vez había aprendido a manejar en profundidad Unity3D, empecé con los tutoriales de Vuforia. Para ésto fue de mucha ayuda la comunidad de desarrolladores, ya que hay muchos tutoriales en Internet de ésta librería así como dudas resueltas en foros.
- Realización de prototipos de Unity3D y Vuforia. Con los conocimientos de Unity y de Vuforia ya adquiridos con tutoriales ajenos, comencé a desarrollar pequeñas pruebas de concepto utilizando ambas tecnologías. Ésto nos sirvió para saber qué aspectos de Vuforia nos podían ser útiles.
- Puesta en práctica de reconocimiento de texto propio en castellano. Tuve que aprender como añadir un diccionario propio al reconocimiento de texto, ya que el aportado por Vuforia sólo contiene palabras en inglés.
- Testeo con Raúl de diferentes maneras de interactuar con la RA en los primeros momentos.
- Pruebas con Virtual Buttons para ver la viabilidad de utilizarlos en los minijuegos. Tras ver que el tiempo de respuesta de los botones era muy lento descartamos completamente esta opción para un juego.
- Animaciones de los Invaders. Aprendí a animar un GameObject utilizando la funcionalidad “Animation” de Unity3D para animar los brazos de los Invaders.
- Búsqueda de sonidos. Tanto las explosiones como el sonido del rayo láser al ser disparado.

- Realización de fotografías en el museo para poder orientarnos cuando trabajamos en nuestras casas. Ésto nos fue de mucha utilidad cuando buscábamos como desarrollar la yimcana.
- Maquetación en LATEXutilizando la plantilla de . Para escribir la memoria lo hicimos utilizando Google Docs, pero la versión definitiva está maquetada con .
- Añadir las referencias bibliográficas a la memoria de .
- Mantenimiento del repositorio con la memoria en LATEX.
- Comunicación y reuniones con nuestro tutor Guillermo a través de reuniones presenciales y correos electrónicos. Durante todo el curso nos hemos reunido con el tutor los tres miembros del equipo para resolver dudas e informar de cómo llevábamos el proyecto. Ésto nos ha servido mucho para recibir orientación.
- Testeo para calibrar bien el tamaño de los Invaders y las Defensas con el cartel del exterior de la Facultad. Hubo que adaptar el tamaño ya que al desarrollar en mi casa utilizaba una fotografía del cartel impresa y el resultado no era el mismo.
- Búsqueda de imágenes que fueran fáciles de reconocer por la cámara de Vuforia. No todas las imágenes se reconocen igual de bien por Vuforia, así que con los códigos QR utilizados tuve que buscar algunos que fueran buenos en este sentido.
- Escritura de todos los apartados de la memoria (menos los de los minijuegos de mis compañeros) y revisión de los contenidos de la misma.
- Toma de decisiones con el resto de mis compañeros.
- Traducción al inglés del abstract.
- Testeo con smartphones Moto G 2013 y Orange Hi4G.

## 10.4. María Picado

En mi caso, mi trabajo ha estado dividido en varias etapas, en cada una de las cuales me he dedicado a diferentes tareas.

- La primera de ellas, fue junto a mis compañeros la realización de tutoriales en Unity3D para comprender y afianzar los conocimientos sobre el funcionamiento de esta herramienta.
- Como ya dijimos anteriormente, aunque si conocíamos Unity, Vuforia era completamente desconocida para nosotros, con lo que mi siguiente tarea fue la de investigación y autoaprendizaje para conocer el funcionamiento de Vuforia. Y más adelante comencé con la creación de pequeños prototipos en Unity3d y Vuforia.

- Durante todo el proceso de creación del proyecto, hemos mantenido los tres reuniones periódicas con nuestro director del TFG para ir mostrándole los avances y ponernos nuevos objetivos de cara a la siguiente reunión.
- Una vez adquirimos los suficientes conocimientos para poder desenvolverse tanto con Unity como con Vuforia, decidimos reunirnos los tres para diseñar el videojuego y decidir qué minijuegos implementaremos.
- En esa reunión, una de las decisiones que tomamos fue la de que juegos íbamos a implementar cada uno, y a partir de ese momento me centré en la realización del minijuego Water Pipes. Lo primero que hice fue documentarme del juego original para ver cómo sería la mejor manera de adaptarlo a la RA.
- Una de las cosas más importantes para que el juego se pudiera adaptar a la RA, era obtener la forma de que el usuario pudiera manipular las tuberías. Al ser una aplicación móvil, lo primero que intenté fue el ?DRAG AND DROP?, para que el jugador pulsara una tubería y la arrastrarse hasta donde quisiera cambiarla y al soltarla se cambiará automáticamente una tubería por la otra. Pero esta opción me dio bastantes problemas y entonces decidí que la forma en que se fueran colocando las tuberías fuera el intercambio entre ellas, pulsando sobre una e intercambiando por la siguiente en ser pulsada.
- Una vez implementada la funcionalidad para colocar las tuberías, comencé a implementar la parte más importante del juego; el flujo del agua. Como existen 6 opciones de tuberías distintas y cada una de ellas tiene otras dos direcciones posibles por las que puede circular el agua. Tenía que buscar una forma de encapsular esa información para que lo único que nos preocupase fuera la salida de la tubería actual y la entrada de la siguiente. Por eso decidí usar el patrón de diseño Command.
- Al acabar la implementación del juego, el siguiente paso fue realizar el mayor número de test posibles a los usuarios. Los test los realizamos del juego completo, pero pedimos a los usuarios que puntuaron los juegos de manera independiente. Por eso, una vez que obtuvimos los primeros resultados, me dispuse a modificar el juego para implementar las mejoras que me aconsejaron los usuarios.
- Durante todo el proyecto he colaborado en el desarrollo de esta memoria.

# Bibliografía

AUMENTATY. Web oficial. Disponible en <http://www.aumentaty.com/>.

AZUMA, R. T. A survey of augmented reality. *Introduction*, 1997.

BOSCO, R. y CALDANA, S. Construye tu propia exhibición. Disponible en <http://blogs.elpais.com/arte-en-la-edad-silicio/2012/05/construye-tu-propia-exhibicion.html>.

CAMONAPP. Web oficial. Disponible en <http://www.camonapp.com/>.

CARO MARTÍNEZ, M. y HERNANDO HERNÁNDEZ, D. *Realidad aumentada para el Museo de América*. Tfg, Universidad Complutense de Madrid, 2014 - 2015.

CRAZY MINNOW STUDIO, L. Salsa with randomeyes en asset store. Disponible en <https://www.assetstore.unity3d.com/en/#!content/16944>.

INGRESS. Web oficial del juego. Disponible en <https://www.ingress.com>.

OF LONDON, M. Street museum. Disponible en <http://www.museumoflondon.org.uk/Resources/app/you-are-here-app/home.html>.

MIGS. Web oficial del museo garcia-santesmases en la facultad de informática de la ucm. Disponible en <http://www.fdi.ucm.es/migs/>.

MOCADELE.NET. Realidad aumentada y realidad virtual. Disponible en <http://mocadele.net/arloon-app-educativas-de-ciencias-con-realidad-aumentada/>.

NINTENDO. Ar cards videogame. Disponible en <https://www.nintendo.es/Familia-Nintendo-3DS/Software-instantaneo/Juegos-RA-la-realidad-aumentada/Juegos-RA-la-realidad-aumentada-115169.html>.

NINTENDO. Pokemon go. versión del juego pokemon en realidad aumentada. Disponible en <http://www.pokemon.com/es/videojuegos-pokemon/pokemon-go>.

UNITY3D. Tutoriales de unity3d. Disponible en <https://unity3d.com/es/learn/tutorials>.

VOZME. Herramienta web para transformar texto en voz. Disponible en <http://vozme.com/>.

VUFORIA. Documentación oficial de vuforia. Disponible en <https://developer.vuforia.com/library/getting-started>.

VUFORIA. Vuforia library text recognition. Disponible en <https://developer.vuforia.com/library/articles/Training/Text-Recognition-Guide>.

WIKIPEDIA (Arkanoid). Disponible en <https://en.wikipedia.org/wiki/Arkanoid>.

WIKIPEDIA (Pipe Mania). Disponible en [https://en.wikipedia.org/wiki/Pipe\\_Mania](https://en.wikipedia.org/wiki/Pipe_Mania).

WIKIPEDIA (Space Invaders). Disponible en [https://es.wikipedia.org/wiki/Space\\_Invaders](https://es.wikipedia.org/wiki/Space_Invaders).

## **Lista de acrónimos**

