Trabajo Fin de Grado

GRADO INGENIERÍA INFORMÁTICA

Facultad de Matemáticas
Universidad de Barcelona

Creación de videojuego ‘Shooter on Rails’ utilizando periféricos

---

Joshua García Palacios

Director: Eloi Puertas Prats
Realizado: Departamento de Matemática Aplicada i Análisis. UB

Barcelona, 26 de junio de 2015

Índice general

Abstract

El proyecto documentado a continuación con nombre “Creación de videojuego Shooter on Rails utilizando periféricos” consiste en el desarrollo de un videojuego para PC utilizando los periféricos que se usan hoy en día.

Para poder realizar el siguiente proyecto se ha realizado un estudio de los periféricos actuales para ver cual se ajustaría mejor al tipo de juego que se va a desarrollar. También se han analizado videojuegos existentes de este estilo para poder observar cómo es la ambientación y la jugabilidad.

El proyecto será desarrollado para la plataforma de PC, utilizando alguno de los periféricos analizados, ya que para esta plataforma no existen juegos que utilicen otro tipo de periféricos que no sea teclado y ratón.

En el siguiente documento se detalla todo el proceso de desarrollo del videojuego junto a las herramientas e implementaciones que han sido necesarias para poder llevarlo a cabo.

Abstract (Català)

El projecte documentat a continuació amb nom "Creació de videojoc Shooter on Rails utilitzant perifèrics" consisteix en el desenvolupament d'un videojoc per a PC utilitzant els perifèrics que es fan servir avui dia.

Per poder realitzar el següent projecte s'ha fet un estudi dels perifèrics actuals per veure quin s'ajustaria millor al tipus de joc que es vol desenvolupar. També s'han analitzat videojocs existents d'aquest estil per poder observar com és l'ambientació i la jugabilitat.

El projecte serà desenvolupat per la plataforma de PC, utilitzant algun dels perifèrics analitzats, ja que per a aquesta plataforma no existeixen jocs que utilitzin un altre tipus de perifèrics que no sigui teclat i ratolí.

En el següent document es detalla tot el procés de desenvolupament del videojoc juntament amb les eines i implementacions que han estat necessàries per poder dur-lo a terme.

Abstract (English)

The project documented below with name "creation of video game Shooter on Rails using peripherals" consists in the development of a video game to PC using the peripherals that are used today.

To perform the following project has conducted a study of current peripherals to see which would adjust better to the type of game that will be developed. Also the existing games of this style were analysed to see as it is the atmosphere and gameplay.

The project will be developed for the PC platform, using one of the tested peripherals, since there are no games that use other types of peripherals that keyboard and mouse for this platform.
The following document details the process of development of the videogame and all implementations that have been needed to carry it out.

1. Introducción

A día de hoy, el mundo de los videojuegos es un sector que diariamente está consiguiendo más y más seguidores a medida que pasa el tiempo. La evolución continuada de hardware y de los nuevos dispositivos, dan lugar a las nuevas generaciones de consolas y dispositivos portables, como pueden ser los smartphones o tablets, que están consiguiendo que la industria del videojuego pueda llegar a todas las personas sin importar la edad.

Todo esto no hubiera sido posible de no ser por “Pong” [1] el primer videojuego de la historia que nació a principios de la década de los 70. A partir de este momento las empresas comenzaron a invertir en este tipo de sector junto a sectores como el del hardware para poder fabricar máquinas que pudieran soportar los videojuegos que vendrían de aquí en adelante.

En los últimos años, se asiste a una era de progreso tecnológico dominada por una industria que promueve un modelo de consumo rápido donde las nuevas superproducciones quedan obsoletas en pocos meses, pero donde a la vez un grupo de personas e instituciones han iniciado el estudio formal de la historia de los videojuegos.

Dentro de este proceso evolutivo del mundo del videojuego hay que destacar el desarrollo de los periféricos. En los inicios, los periféricos no eran más que simples palancas, que solamente podías presionar hacía delante o hacía detrás, hasta llegar a evolucionar en complejos mandos o incluso hasta en cámaras capaz de reconocer el cuerpo del jugador.

Este fenómeno dio lugar al desarrollo de videojuegos exclusivos para estos tipos de periféricos, y uno de estos estilos fue el “Shooter On-Rails”.

El “Shooter On-Rails” o juego sobre raíles, es un estilo de videojuego en el que el jugador usa un periférico similar a una pistola y tendrá que ir acabando con los enemigos, que van apareciendo, mientras que a la vez va avanzando, automáticamente, sobre un raíl el cual no es visible para el jugador.

1.1. Motivación

La creación del proyecto de un videojuego “Shooter On-Rails” utilizando periféricos surge por los principales motivos:

- Actualmente todas las marcas de consolas conocidas como son, Sony, Nintendo o Microsoft, trabajan en el desarrollo de nuevos periféricos, en el que cada vez el ser humano interactúa más con ellos, para poder ser utilizado después en los nuevos videojuegos que saldrán en el mercado produciendo en el consumidor una doble inversión tanto en la compra del videojuego como del periférico. Por eso una de las principales motivaciones de este proyecto es desarrollar un dispositivo que actúe como intermediario entre el periférico de uno de estas marcas y el PC.

- Los juegos del estilo “Shooter On-Rails” son un tipo de juego que crearon mucha expectación cuando salieron, ya que fueron los primeros videojuegos en que los jugadores utilizaban un tipo de periférico que no era un mando, sino en este caso una pistola.

Por eso otro de los motivos por el cual realizar el siguiente proyecto es la creación de un videojuego que siga los pasos de este estilo de juego.

- Otro de los motivos que personalmente es importante, y no sé si la gente es consciente, es que el uso de periféricos en el mundo del PC no están grande como el de las consolas de sobremesa. Parece ser que los juegos de PC se han encasillado en el uso del teclado y ratón, y la presencia de mandos u otro tipo de periféricos no son tan importantes. Por este motivo creo que es importante crear un videojuego de este estilo para poder hacer uso de periféricos como mandos, pistolas, cámaras etc.

- El principal motivo que hace que todo esto sea posible es la motivación máxima de poder crear mi propio videojuego y que sea funcional para que la gente lo pueda probar y lo puedan valorar y criticar.

1.2. Objetivos

Como he mencionado, en la sección de Motivación del proyecto, el principal objetivo de este proyecto es poder realizar un juego del estilo “Shooter On-Rails” que sea lo más completo y funcional posible para que los jugadores puedan vivir la misma experiencia como si estuvieran jugando a uno de los juegos realizados por profesionales de esta industria.

Para ello debe cumplir los siguientes objetivos:

- Debe de tener una interfaz atractiva para el jugador e intuitiva. Esta parte es muy importante, ya que el estilo de estos juegos debe ser de esta manera porque el jugador únicamente tendrá que estar concentrado en lo que le aparecerá en la pantalla.

- Deberá tener un modo “Arcade” o “Historia” con varios niveles de juego. Como en todos los videojuegos, estos tienen varios niveles en los que el jugador tendrá que ir superando los retos hasta llegar al final del juego y poder enfrentarse al jefe final.

- Deberá tener otras modalidades de juego. A parte del modo “Arcade” o “Historia” los videojuegos suelen tener otro tipo modalidades, más pequeñas que las principales, en las que el jugador realiza otro tipo de tarea parecida a la principal o no.

- Deberá tener la opción de poder jugar con un segundo jugador. Muchos de los videojuegos que existen hoy en día incorporan la opción de poder añadir un nuevo jugador en mitad de la partida. Esto hace que la experiencia del juego sea compartida y produzca una mayor satisfacción en la jugabilidad.

- Deberá poder permitirnos jugar en red con amigos. La opción del juego en red o multiplayer, es una de las características que hoy en día es indispensable y debe tener un videojuego.

- Deberá estar en varios idiomas. Los videojuegos de hoy en día son traducidos a varios idiomas para poder llegar a todo tipo de jugadores de diferentes países del planeta.

- Deberá tener “Logros” que el jugador podrá conseguir. La opción de “Logros” en los videojuegos es un tema que cada vez se está utilizando más y más en los nuevos juegos que salen al mercado, ya que los desarrolladores ponen a prueba las habilidades y “ansias” de competitividad de los jugadores para poder exprimir al 100% el videojuego haciendo que la jugabilidad se vea incrementada.

- Deberá ser exclusivo para PC. La exclusividad para PC es importante, ya que el juego a desarrollar queremos que utilice periféricos de otras consolas en este.

- Deberá hacer uso de periféricos. Para ello se probarán los periféricos de Sony y Nintendo como son el PS3 controller [2] y el WiiMote [3].

1.3. Estructura de la memoria

En la memoria se incluyen al detalle los siguientes puntos del proyecto:

- Capítulo 2: En este capítulo se hablará de la historia y evolución de los videojuegos y como ha ido evolucionando el estilo de juego “Shooter On-Rails”.

- Capítulo 3: En este capítulo hablaremos del desarrollo del videojuego, que cosas tendremos que tener en cuenta a la hora de crear un juego, y haremos un análisis de los periféricos que hay en el mercado analizando ventajas e inconvenientes de cada uno y cuales nos han sido útiles para el desarrollo del proyecto.

- Capítulo 4: En el siguiente capítulo se detallará la tecnología y herramientas que se han utilizado.

- Capítulo 5: En este capítulo entraremos al detalle de la implementación de los puntos que he creído más importantes del desarrollo de un videojuego.

- Capítulo 6: En este capítulo se hablará de la metodología de trabajo y planificación que se ha utilizado a lo largo del proyecto. También se analizará cuanto habría que haber invertido para poder realizar este proyecto y si se podría subvencionar utilizando las llamadas “kickstater” que están muy de moda hoy en día.

- Capítulo 7: En este capítulo se pondrán los resultados obtenidos del videojuego mostrando imágenes in-game, y se hará un estudio a través de una encuesta realizada a las personas las cuales podrán meterse en la piel de un “Tester” y podrán dar su propia opinión sobre el videojuego.

- Capítulo 8: En este capítulo se hablará sobre las reflexiones y conclusiones finales del trabajo y se evaluará la posibilidad de mejoras y extensiones futuras.

2. Antecedentes

En esta sección nos centraremos en períodos históricos y hechos que han marcado un proceso de cambio en el mundo de los videojuegos y la evolución que han sufrido los periféricos desde entonces hasta ahora.

2.1. Historia de los videojuegos

Mucho antes de que surgiera el primer sistema compatible con los primeros videojuegos, los científicos Alexander S.Douglas y William Higginbotham fueron los instigadores de lo que en un tiempo después se conocería como “videojuego”.

Alexander S. Douglas era un estudiante de la Universidad de Cambridge cuando en 1952 desarrollo el juego del **OXO** (3 en raya) utilizando como base una **EDSAC** (Ilustración1).

William Higginbotham a finales de los 50 creó lo que sería el primer juego para dos jugadores sirviéndose de un programa para el cálculo de trayectorias y un osciloscopio, dando como resultado “Tennis for two” (véase Figura 2.1).

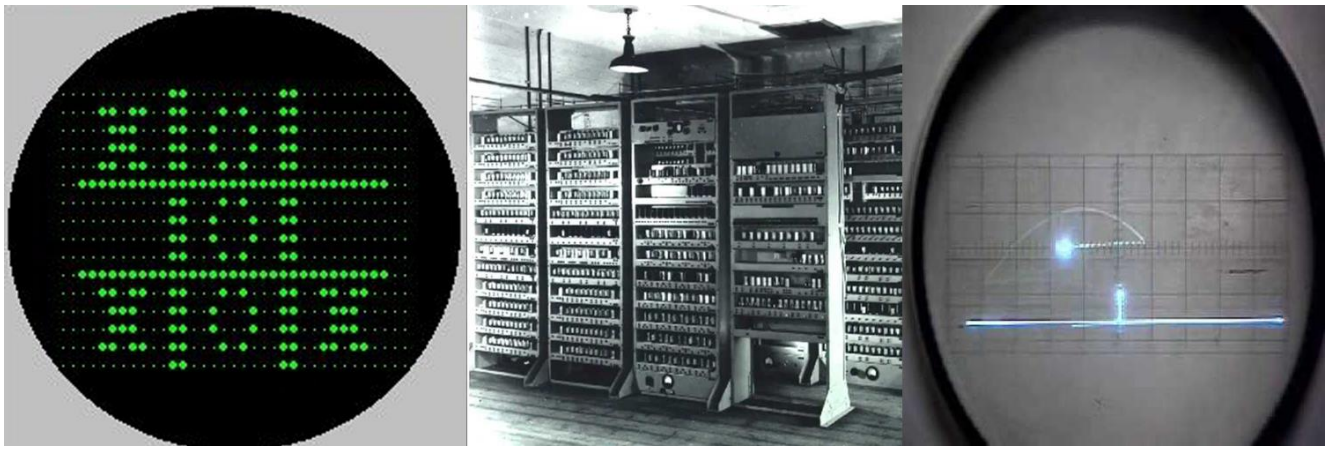


Figura 2.1: A mano izq. tenemos el juego del OXO en el centro la computadora ECDSA y a la derecha el juego “Tennis for two” [4].

A partir de este momento fueron muchos los que empezaron a investigar en el tema de los videojuegos hasta que finalmente en 1972 llegó la primera máquina recreativa, que implementaba de forma mejorada el juego de Higginbotham, PONG por Nolan Bushnell [5] fundador de **Atari** [6], la primera empresa de videojuegos del mundo en aquel entonces.

A partir de este momento la industria del videojuego sufrió numerosos avances técnicos, sobre todo a nivel de hardware, y comenzaron a aparecer nuevas máquinas recreativas como la mítica “Space Invaders” o la “Asteroids” (véase Figura 2.2).

Figura 2.2: A mano izq. tenemos “PONG” la primera recreativa, en el centro a “Space Invaders” y a la derecha “Asteroids” [4].

La década de los 80s fue un período de fuerte crecimiento en el sector del videojuego en donde comenzaron a llegar nuevas máquinas a los salones de diferentes compañías. Pero todo esto pronto se vería afectado por la llamada “crisis del videojuego” en 1983, ya que Japón entro en el sector del videojuego sacando su primera consola la Famicom [7].

En 1985, se produjo un punto de inflexión en el mundo de los videojuegos con el lanzamiento del primer videojuego que tenía una historia y varios niveles con un principio y un final. Este juego fue el actualmente famosísimo “Super Mario Bros” [8]. El lanzamiento de este juego supuso tal explosión de creatividad que las futuras generaciones lo copiarían a la hora de implementar nuevos juegos.

A finales de los 80s empezaron a salir las primeras consolas de 16bits, Sega Megadrive [9], y la primera consola portátil, GameBoy [10].

Con el surgimiento de las primeras consolas de 16bits comenzaron a desarrollarse juegos con técnicas cada vez más innovadoras y nuevas. Pero poco tardaron en salir una nueva remesa de consolas con arquitecturas de 32bits como la PlayStation de Sony [11] y la Nintendo 64 [12] de Nintendo que fueron pioneras en los primeros juegos que utilizarían modelados 3D.

La consola de SONY fue la primera que incorporó el CD-ROM como medio de reproducción de los videojuegos. Esto permitió crear juegos más largos y con más capacidad dentro de las restricciones que el hardware de la consola le permitía.

En esta década surgieron juegos muy populares que marcaron un antes y un después en el mundo del videojuego y que hoy en día muchos de estos títulos se siguen desarrollando con las consolas de última generación. Algunos de estos títulos fueron: Final Fantasy VII (Square), Resident Evil (Capcom), Winning Eleven 4 (Konami), Gran Turismo (Polyphony Digital) y Metal Gear Solid (konami) etc.

Con la entrada del nuevo milenio la compañía Japonesa, SONY, saco al mercado su nueva consola PlayStation 2 revolucionando nuevamente el mundo de los videojuegos. Pensó que el CD-ROM se había quedado obsoleto y aposto por una nueva tecnología del momento, el

DVD. El hardware había sido mejorado de tal forma que se empezaron a diseñar videojuegos con aspectos más realistas utilizando nuevas técnicas de iluminación y sombras. Los antiguos polígonos habían sido substituidos por formas más definidas en todos los elementos de la escena (véase Figura 2.3).

Figura 2.3: Se puede apreciar la diferencia gráfica que hubo en la consola de PSX con la nueva consola de PS2.

A partir de este momento el desarrollo de videojuegos empieza a estar limitado a las prestaciones que ofrecen las consolas. Los nuevos videojuegos que empezaron a salir en aquel momento intentarían explotar al máximo todos los recursos de esta para poder realizar todo tipo de acciones que hasta hace unos poco era impensable.

Actualmente, nos encontramos en la octava generación de consolas y por ello de videojuegos. En esta generación se centra especialmente en el gran nivel de detallado de todos los elementos de la escena sumergiendo al jugador dentro de una realidad cada vez más real al mundo al que se encuentra y por ello tenga una máxima experiencia del videojuego.

A día de hoy cada vez estamos más impactados con la realidad y calidad que ofrecen algunos videojuegos. Esto hace pensar cómo serán los juegos de dentro de unos años, ya que en un, breve, período de 20 años el mundo del videojuego ha sufrido un cambio exponencialmente, para mejor, que hace cuestionarse al jugador cual será los límites en un futuro, no muy lejano, del mundo del videojuego.

2.2. Estilo ‘Shooter on Rails’

Como mencionamos, un poco en la introducción del proyecto, los ‘Shooter on Rails’ es un tipo específico del género ‘shooter’. En este tipo de juego el jugador controla un puntero con forma de cruz utilizando un periférico, con forma de arma, para moverlo a través de la pantalla.

En estos juegos el jugador se moverá de forma predefinida a través de todo el escenario teniendo que superar enemigos y peligros que le aparezcan. En otras palabras, el jugador esta sobre un ‘rail’ y no puede decidir hacía donde quiere ir. En algunos ‘Shooter on-Rails’ dan la posibilidad, limitada, al jugador de poder elegir la dirección que quiere tomar.

Una de las características importantes de este tipo de juego es que el personaje debe estar en una perspectiva de primera persona, aunque no es del todo obligado.

2.3. Evolución de los ‘Shooter on Rails’

La primera vez, en la historia, que apareció el primer videojuego del estilo ‘Shooter on Rails’ fue casi a mediados de la década de los 80s. Las nuevas máquinas recreativas que estaban saliendo en aquel entonces empezaban a apostar por nuevos estilos de juegos, y uno de estos estilos fue el de “sobre raíles”.

Uno de los primeros juegos que salió, fue el “Star Wars” [13]. Era un juego totalmente vectorial que recreaba las formas de las naves y los enemigos que aparecían en el largometraje. El juego nos metía en la piel de “Luke” en su “Caza estelar T-65 Ala-X” [14] que avanza a través del espacio mientras nos aparecen naves de combates enemigas. En el centro de la pantalla había un puntero el cual era movido por una palanca que tenía la máquina haciendo que pudiéramos desplazar el puntero por toda la pantalla y de esta forma poder abatir a los enemigos (véase Figura 2.4).

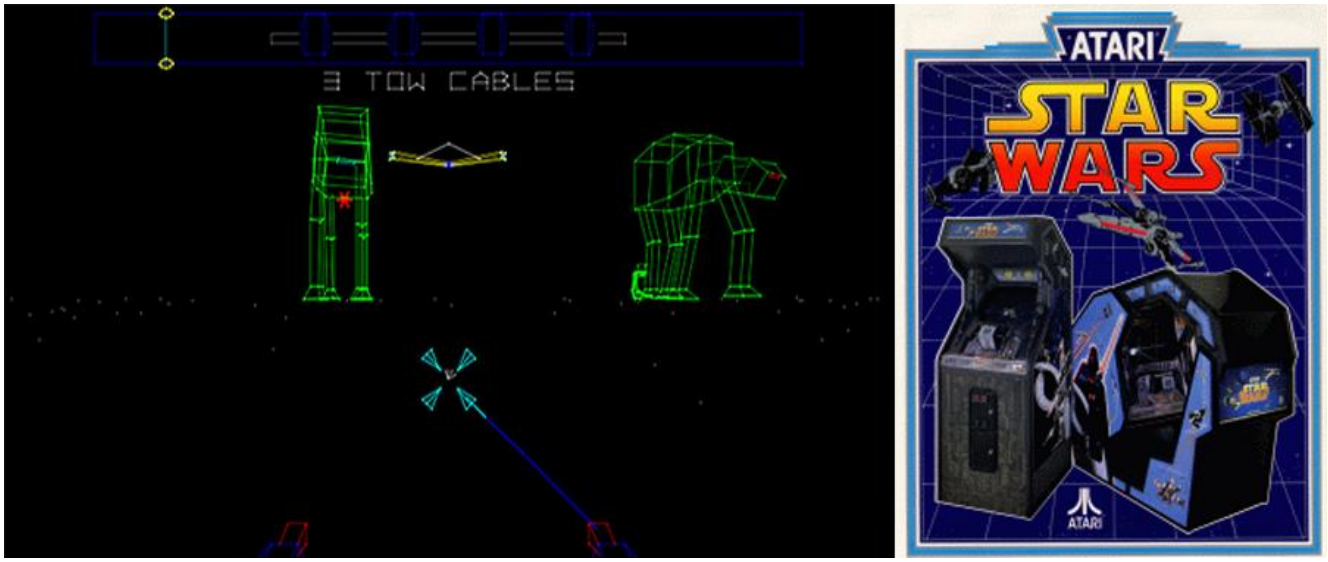


Figura 2.4: Imagen izq. juego “StarWars”(1983), imagen der. recreativa de Atari que contenía el juego. Podemos ver en el centro de la máquina el periférico con forma de palanca para poder jugar.

En 1984 Nintendo saco su primer periférico para su consola actual, el NES Zapper [15].
El NES Zapper era una “pistola de luz” con forma de pistola. Este utilizaba una técnica innovadora que al apretar el gatillo la pantalla se volvía negra por un segundo, como si de un flash se tratase. Entonces se dibujaban pequeños cuadros de color blanco desde el inicio de la pantalla hasta el final, y si se detectaba un cambio importante en este proceso quería decir que había un objetivo en la pantalla y por lo tanto acabaríamos con él.
Existe la posibilidad de engañar a esta técnica, algunos jugadores demostraron que, para algunos juegos mal diseñados, utilizando la luz de una bombilla se podía engañar al periférico haciendo que encontrara objetivos sin la necesidad de apuntar a la pantalla (véase Figura 2.5).

Figura 2.5: Imagen izq. NES Zapper e imagen der. primer videojuego de consola compatible con el periférico.

A finales de los 90s, con la incorporación al mercado de la nueva generación de consolas, empezaron a salir nuevos juegos ‘sobre raíles’ como fueron el “Time Crisis” y el “House of the Dead”, que tuvieron una gran popularidad en su momento. La jugabilidad de este tipo de juegos se había visto mejorada, con el uso de los nuevos gráficos de las nuevas consolas, y a su vez por la aparición de las nuevas pistolas que utilizaban una nueva técnica más precisa que calculaba la posición del puntero a partir de la excitación del fósforo de las pantallas de tubo de catódico en el que la pistola recibía la señal de dicha excitación y calculaba la posición del puntero en la pantalla (véase Figura 2.6).

Figura 2.6: GunCon negro hecha por Namco para la PlayStation.

Actualmente los juegos ‘sobre raíles’ que existen hoy en día utilizan todo tipo de periféricos, no solo pistolas sino mandos inalámbricos, cámaras de captación de movimiento o sensores IR(véase Figura 2.7), haciendo que la experiencia del juego sea mayor y que sea más frenético para el jugador de lo que lo eran sus predecesores (véase Figura 2.8).

Figura 2.7: Imagen izq. WiiMote (Wii), imagen central PS Move (Playstation), imagen der. Kinect (XBox).

Figura 2.8: Imagen izq. “House of the Dead 4” (Wii/PC), imagen central “Blue Estate” (PS4/ XBOXOne) imagen der. “Resident Evil Chronicles” (PS3).

Esto nos lleva a la conclusión que este tipo de género nunca pasará de moda ya que con la incorporación de nuevos tipos periféricos se seguirán desarrollando nuevos juegos que sean compatibles con estos y que cada vez sean más originales y mejor trabajados.

3. Desarrollo del videojuego

En este capítulo hablaremos de “Zombie Assault” (véase Figura 3.1), que es el nombre del videojuego del proyecto al más puro estilo Shooter on Rails, y de su historia. También se mencionarán las partes que se han tenido que tener en cuenta a la hora de desarrollar el videojuego. Se hará un análisis de los periféricos que actualmente están en el mercado para escoger cual/es de ellos poder utilizar en el proyecto y el porqué.

Figura 3.1: Logo del Proyecto “Zombie Assault”.

3.1 Historia

Zombie Assault es un juego que podemos situarlo en la actualidad. Por culpa de varios brotes, que han aparecido por todo el mundo, de una cepa mutada de ébola los científicos se ponen a investigar una posible vacuna que sea capaz de parar la expansión para que la gente pueda sobrevivir a la enfermedad. Pero esta cepa es mucho más agresiva que otras vistas anteriormente y parte de la población mundial va muriendo día tras día.

Una compañía farmacéutica Japonesa cree haber dado con el antiviral y están preparando todos los efectivos necesarios para poder subministrar la vacuna a la población.
Cuando comienzan a suministrar la vacuna a los civiles infectados empiezan a observar efectos secundarios a todos aquellos a los que se les ha sido suministrado apareciendo nuevas patologías y hemorragias hasta llevarlos al punto de la misma muerte.

Los sanitarios no dan credibilidad a lo que están viendo, todos aquellos que hasta hace unos momentos estaban muertos se estaban volviendo a levantar. Los muertos que se estaban levantando habían vuelto a la vida en forma de zombies. Los zombies empezaron a atacar a todos aquellos que estaban vivos arrancándoles sus extremidades para luego después comérselas.

El gobierno de los EEUU es alertado del caos que asola el país nipón y decide enviar un equipo de fuerzas especiales, especialistas en misiones de alto riesgo para acabar con la pandemia que se ha desatado y no pueda pasar a otros países.

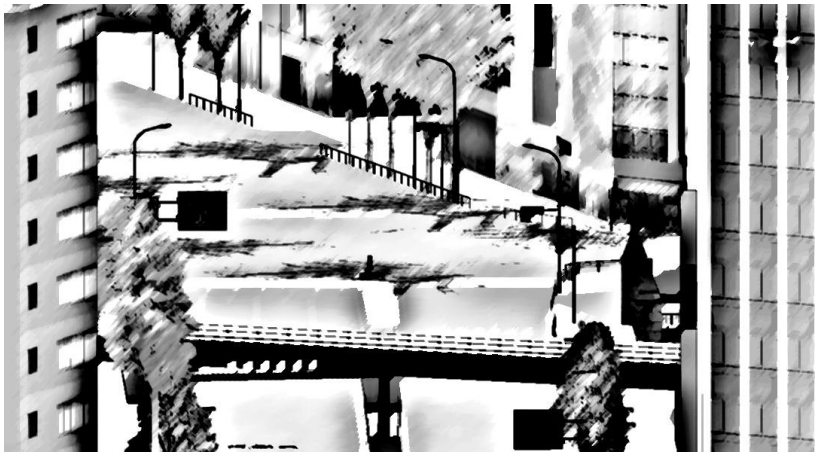
3.2 Storyboard

Para el desarrollo del juego se ha hecho unos storyboards, describiendo gráficamente el desarrollo del personaje en la escena en la que se encuentra.

3.2.1 Arcade

El modo “Arcade” o “Historia” es el modo principal del juego. En este modo nos situaremos en un mapeado de la ciudad de Akihabara [15], Tokyo, Japón que es uno de los lugares donde la epidemia se ha extendido. Nuestro personaje tendrá que superar varias hordas de zombies hasta llegar a su objetivo.

A continuación se detallará en formato de viñeta como se ha querido implementar este modo.



En esta primera viñeta mostraríamos el lugar donde se van a desarrollar los acontecimientos del juego. El jugador empezaría en la ciudad a la espera que aparezcan las hordas de zombies.

El jugador a medida que va avanzando tendrá que ir eliminando hordas de zombies para poder hacerse paso hasta su objetivo.

Llegados a ciertos puntos del juego al jugador se le dará la posibilidad de escoger variaciones del camino. El camino escogido le llevará a un punto final diferente del camino opuesto.

A medida que el jugador avance por el nivel aparecerán supervivientes perseguidos por zombies que tendrás la opción de poder salvarlos o simplemente dejarlos morir.

Después de que el jugador haya eliminado a todas las hordas que le han ido apareciendo el último obstáculo que deberá superar antes de terminar el nivel será el Boss.

Una vez eliminado el Boss del nivel aparecerá una pantalla con los resultados obtenidos a lo largo de nuestro recorrido mostrando puntuaciones máximas, muertes, combos etc.

3.2.2 Supervivencia

El modo “Supervivencia” es un modo complementario que tiene “Zombie Assault”. En este modo nos veremos las caras contra una infinidad de zombies que vendrán por todas las direcciones. El modo empezará con hordas bastante dóciles pero a medida que vayamos superándolas se volverán mucho más agresivas.

El modo Supervivencia finalizará una vez nos hayan eliminado de tal forma que podamos registrar nuestra máxima puntuación.

A continuación se detallará en formato de viñeta como se ha querido implementar este modo.

En esta primera viñeta mostraríamos el escenario en el que nos encontramos. El escenario es un pueblo y nosotros estaríamos situados en una calle de 4 direcciones que serían por donde vendrían los zombies a atacarnos.

El juego empezaría con la primera horda de zombies que vendrían a atacar al jugador. El jugador deberá eliminar a todos los zombies para poder ir aumentando de horda.

A medida que el jugador va eliminando a los zombies iremos girando a los otros carriles para eliminar a los zombies que nos irán apareciendo del lado contrario al que nos encontramos.

Finalmente, cuando el jugador haya sido eliminado saldrá una ventana de resultados mostrando las marcas obtenidas por el jugador.

3.3. Jugabilidad

La jugabilidad es un factor muy importante en la creación de videojuegos, ya que esta será decisiva a la hora de aceptar un videojuego o no por los jugadores. Por ese motivo se hace un estudio tan exhaustivo de las propiedades que querernos que tenga nuestro juego de cara al jugador futuro. Para garantizar la experiencia del jugador, citaré alguna de las propiedades que he tenido en cuenta a la hora de desarrollar ‘Zombie Assault’:

- **Satisfacción:** es el agrado del jugador ante el videojuego en algunos aspectos concretos de éste. La satisfacción es un atributo un tanto complejo ya que dependerá de los gustos del jugador. Para poder medir el grado de satisfacción tendríamos que partir de una base de juego determinado.

- **Aprendizaje:** es la facilidad para dominar la mecánica del juego, es decir objetivos reglas y formas de interactuar con el juego.

- **Inmersión:** es la capacidad de provocar al jugador verse envuelto en el videojuego volviéndose parte e interactuando con él.

- **Motivación:** es la característica que mueve al jugador a realizar determinadas acciones y persistir hasta conseguir aquello que te estás proponiendo.

- **Emoción:** son los impulsos originados por los estímulos que desencadenan conductos de reacciones automáticas.

- **Socialización:** es el factor de los juegos que fomentan la experiencia en grupo, lo que hace crear vínculos de compañerismo a las relaciones que se establecen con otros jugadores.

3.4. Dificultad

Mucho se habla hoy día sobre las facilidades que nos ofrecen los videojuegos, o la falta de obstáculos a la hora de completarlos. La dificultad es una de las características que siempre

ha existido en la lógica de desarrollo de los videojuegos que tratan de poner a prueba al jugador llevándolos a la más profunda frustración o por el contrario al más absoluto aburrimiento.
Esto se puede representar con la llamada “Curva de la Dificultad” (véase Figura 3.2). La Curva de la Dificultad es la relación que hay entre la dificultad del videojuego entre el tiempo jugado intentando pasarte el enemigo, nivel etc.

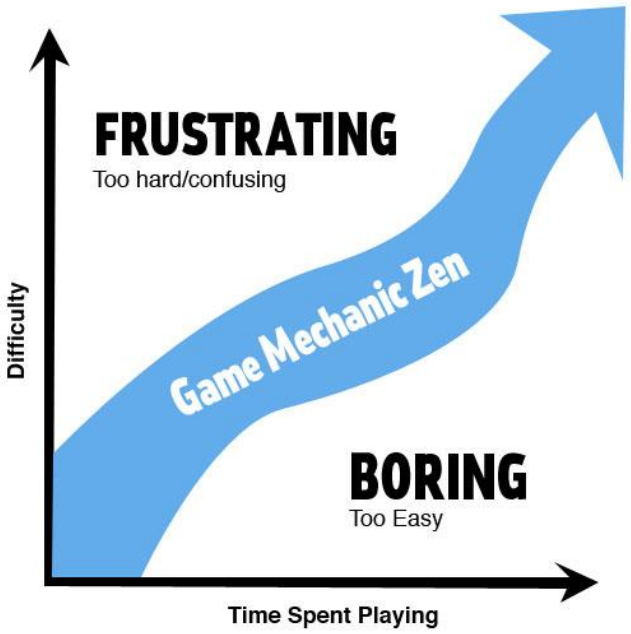


Figura 3.2: Curva que nos muestra el grado de dificultad vs el tiempo dedicado [16].

A la hora de desarrollar “Zombie Assault” este factor se ha tenido muy en cuenta. El juego por defecto empieza con una dificultad a la que hemos llamado “Normal”, es una dificultad balanceada para el jugador, novato, que juega por primera vez al juego. Este factor de dificultad es apreciable en el comportamiento de los enemigos, que los hará más agresivos o dóciles cuando nos ataquen o más rápido o lentos a la hora de ejecutar las acciones. También será apreciable en el número de enemigos que haya en nivel, apareciendo enemigos más poderosos desde el principio y en mayor cantidad.

Zombie Assault nos ofrece la posibilidad de cambiar el nivel de dificultad desde el menú de “Opciones” (véase Figura 3.3) pudiendo elegir dificultadas más altas para los jugadores más experimentados o más bajas para aquellos que únicamente quieren pasar un rato entretenido.

Figura 3.3: Opciones de “Zombie Assault” para cambiar los ajustes.

3.5. Idiomas

Otros de los requisitos a la hora de desarrollar un videojuego es el idioma. Actualmente los juegos llegan a todas las partes del planeta, pero no por ello tienen que venir traducidos o doblados a todos los idiomas existentes del planeta. Por eso existen clasificaciones llamadas “MultiX”. Los MultiX son el número de idiomas que se ha traducido o doblado un videojuego dentro de una región, siendo X el número de idiomas traducidos para esta.

A día de hoy sistemas “Multi” que se han utilizado son:

- Multi3: Inglés, Francés, Alemán.
- Multi4: Inglés, Francés, Alemán, Italiano.
- Multi5: Inglés, Francés, Alemán, Español, Italiano.

En Zombie Assault se ha tenido en cuenta esta característica y por eso lo hemos desarrollado para traducir 3 idiomas “Español”, “Inglés” y “Japonés”. Como en el apartado anterior, desde el menú de Opciones podemos cambiar el Idioma del juego visualizando al instante como cambia todo el texto del juego al idioma seleccionado (véase Figura 3.4 y 3.5).

Figura 3.4: Mostramos desde opciones el idioma que está actualmente (English).

Figura 3.5: Cambiamos el idioma al Japonés y observamos el cambio de todo el texto.

3.6. PEGI

Otro de los factores a tener en cuenta cuando se desarrolla un videojuego es hacía quien va dirigido, es decir, los juegos están clasificados por edades y no todo los jugadores son aptos para jugar a que juegos. PEGI (*Pan European Game Information)* es un sistema europeo para clasificar el contenido de los videojuegos y otro tipo de entretenimiento.

PEGI es clasificado por edades (véase Figura 3.6) y por contenido (véase Figura 3.7):

Figura 3.6: Estas son las franjas de edades mínimas en las clasifica PEGI los juegos.

Los descriptores de contenido se clasifican en:

- Lenguaje soez: Puede contener palabrotas, lenguaje soez, amenazas etc.
- Discriminación: Puede contener violencia o discriminación de razas.
- Drogas: Puede tener referencias a tema de drogas o derivados.
- Miedo: Puede contener escenas perturbadoras o aterradoras.
- Juego: Puede contener apuestas con dinero real o ficticio.
- Sexo: Puede hacer referencia a relaciones sexuales o desnudos.
- Violencia: Puede contener escenas de personas que sufran o mueran.
- Online: Puede un modo de juego online.

Figura 3.7: Descriptores de contenidos de PEGI.

Según PEGI Zombie Assault entraría dentro de la categoría de juego para mayores de 16 años y contendría descriptores de violencia, miedo y online (véase Figura 3.8).

Figura 3.8: Clasificación PEGI Zombie Assault.

3.7. Análisis de los periféricos a utilizar

Como hemos comentado en capítulos anteriores, los juegos de estilo “Shooter on-Rails” han triunfado por el tipo de dispositivo que se ha utilizado para poder jugar con ellos. Por eso, se ha realizado un análisis de los periféricos que hay actualmente en el mercado para poder desarrollar “Zombie Assault” con alguno de ellos.

Los periféricos que se han analizado han sido:

- PS3 Controller (Sony).
- PS Move (Sony).
- Kinect (Microsoft).
- WiiMote(Nintendo).

PS3 Controller

El mando PS3 Controller (véase Figura 3.9) no deja de ser un mando normal y corriente de la actual generación de consolas, la única característica que tiene que le hace diferente del resto es que utiliza una tecnología llamada SIXAXIS. SIXAXIS o “Seis Ejes” es un sensor

que recoge los movimientos del mando de arriba-debajo de izquierda-derecha y en las diagonales.

También ofrece una buena compatibilidad el controlador con el PC haciendo que podamos jugar a juegos de cualquier tipo de plataforma y compañía utilizando las propiedades de este.

En Zombie Assault he querido aprovechar la tecnología de este mando para poder jugar con el movimiento del SIXAXIS haciendo que el jugador solamente moviendo el mando pueda mover la mirilla de su arma por toda la pantalla.

El único inconveniente encontrado es que el manejo de este es algo complejo al principio hasta que le coges la práctica.

Figura 3.9: PS3 Controller.

PS MOVE

PlayStation Move (véase Figura 3.10) es un periférico de videojuegos de SONY, cuya tecnología está basada en el sensor de movimiento. Para ello usa un mando principal con sensores de movimiento y una esfera, que se ilumina en el extremo, junto a la cámara que se encarga de detectar la posición del mando principal.

Este controlador se analizó para una posible implementación en ZombieAssault, el problema que hubo fue que no había una buena compatibilidad con el motor gráfico usado, ya que no era capaz de reconocerlo.

Otro de los inconvenientes que tenía es que requería de un dongle Bluethoot especifico, no valía cualquier marca solo algunas eran compatible [17] junto con un driver privado.

Figura 3.10: PS Move.

Kinect

Kinect (véase Figura 3.11) permite a los usuarios controlar e interactuar con la consola sin necesidad de tener contacto físico con un controlador de videojuegos tradicional, mediante una interfaz natural de usuario que reconoce gestos, comandos de voz, objetos e imágenes.

Este controlador se analizó para poder ser implementado en “Zombie Assault”, el inconveniente que se encontró fue que al no tener mandos había acciones que no podían ser ejecutadas como por ejemplo, recargar el arma, ir a los menús etc.

También al ser un plugin de una extensión del SDK de Microsoft el código no era libre y no se podía acceder a los métodos para poder crear nuevos y adaptarlos a las necesidades del juego.

Figura 3.11: Kinect.

WiiMote

WiiMote(véase Figura 3.12) es el mando principal de la consola Wii de Nintendo, el cual también es compatible con la consola Wii U. Sus características más destacables son la capacidad de detección de movimiento en el espacio la habilidad de apuntar hacia objetos en la pantalla y señalización mediante el uso del acelerómetro y tecnología de sensores ópticos.

El diseño del Wii Remote no se basa en los tradicionales mandos para los videojuegos. A diferencia de ellos, es similar a un control remoto de televisión, creado para ser utilizado en una sola mano y de la manera más intuitiva posible.

Este periférico era el que mejor se adaptaba a las necesidades de “Zombie Assault” ya que al estar dotado de tecnología IR, giroscopios y acelerómetros era compatible con la mayoría del movimiento del juego.

El único inconveniente que tiene es que para poder utilizar todo su potencial es necesario tener una barra sensora emisora de IR ya que de esta manera es capaz de captar el movimiento preciso que hacemos con el mando.

Para solventar este problema he creado una barra sensora casera que satisface las necesidades que estaba buscando [A6].

Figura 3.12: WiiMote.

4. Herramientas de desarrollo

En este capítulo hablaremos de las herramientas que se han utilizado para llevar a cabo el desarrollo de Zombie Assaul y explicaremos para qué parte del proyecto se ha utilizado y cuál ha sido su importancia en la implementación.

4.1. Unity 3D

Para el desarrollo del proyecto se ha utilizado Unity 3D versión 4.6.1f1. Unity 3D es un motor de videojuegos multiplataforma tanto para Windows como OS X que permite crear videojuegos compatibles con los sistemas de Windows, Linux, OS X, iPhone, Android, Windows Phone como a su vez plataformas como PlayStation3, Xbox360, Wii y WiiU .

Unity 3D es un motor de desarrollo que el usuario puede obtener de manera gratuita con las funciones básicas que incluye el motor, pero si queremos sacar todo el su potencial tenemos la posibilidad de hacer un “upgrade” a la versión PRO que incluye funciones y herramientas especiales para poder desarrollar a otro nivel.

Unity 3D ha sido la plataforma en la que hemos creado e importado todos los elementos del proyecto como son escenarios, modelos de enemigos, personajes, objetos, efectos de sonidos, luces, máquinas de estados, programación etc.

Para el proyecto ha sido necesario, y casi obligatorio, el uso de la versión PRO de Unity porque era la única manera que había para poder comunicarnos con el controlador de Wii [A5].

4.2. Blender

Blender [19] es un programa gráfico dedicado, especialmente, al modelado de figuras 3D que incluye funciones de texturización, iluminación, renderización y animación. Blender es compatible con un gran número de ficheros que trabaja con entornos 3D y es potente y fácil

de utilizar. La versión que se ha utilizado para este proyecto ha sido 2.74 que solucionaba un par de problemas que llevaban arrastrando las versiones anteriores.

Blender ha sido una buena herramienta para el desarrollo del proyecto. Se ha utilizado para la edición de los modelos para dotarlos de nuevos miembros como han sido brazos, piernas, cabezas etc. También se ha utilizado para poder añadir huesos “rigging” a los modelos y de esta forma poder crear animaciones que se reproducirán en el juego.

4.3. Adobe Photoshop / Illustrator

Photoshop [20] es un programa creado por la compañía de Adobe que es utilizado para crear, editar, componer, retocar y transformar imágenes. La gran facilidad que tiene para tratar y crear distintas capas superpuestas, nos permite combinar distintos objetos y efectos sin necesidad de modificar nada de la imagen original. Alguna de las funciones que incluye Photoshop es la posibilidad de corregir errores de las imágenes, subir brillo, cambiar tonalidad, aplicar filtros etc.

Illustrator [21], como Photoshop, es un programa de edición de imágenes que se puede utilizar para impresión y multimedia. Se trata esencialmente de una aplicación de creación y manipulación vectorial en forma de taller de arte que trabaja sobre un tablero de dibujo y está destinado a la creación artística de dibujo y pintura para ilustración.

Estas dos herramientas se han utilizado dentro del proyecto para poder diseñar y editar todo lo que sería el tema del HUD del usuario donde tendremos barras de vida, mirillas, balas, textos informativos etc. También se ha usado para los menús de transición de las pantallas o de opciones.

Otra utilidad que se le ha dado ha sido para poder implementar el multidioma del juego, ya que era más fácil trabajar con los textos en imágenes que de otra forma.

Esta parte de implementaciones la podremos ver más en detalle en el Capítulo 5 Implementación.

4.4. Audacity

Audacity [22] es un programa de grabación y edición de sonidos fácil de usar, multiplataforma y de código abierto. Puede grabar y reproducir sonidos, como a su vez importar y exportar archivos de tipo WAV, AIFF, MP3 etc. Audacity es uno de los programas de edición más fiable y avanzado que existe actualmente.

El programa ofrece la posibilidad de grabar sonidos en vivo, convertir grabaciones a sonido digital, editar archivos de varios tipos, permite acciones como cortar sonido, pegar sonido, mezclar sonido, cambiar velocidad de reproducción etc.

Audacity se ha utilizado en el proyecto para lo siguiente:

- Editar efectos de sonido para las acciones de recarga del personaje según el arma utilizada.
- Para efectos de sonido de partículas.
- Para efectos de botones.

- Para audios de enemigos.
- Para la música de los menús.

4.5. GitHub

Github [23] es una plataforma de desarrollo de software que nos permite alojar proyectos en la web de forma gratuita, permitiéndonos la posibilidad que estén de forma pública para que el resto de usuarios puedan ver los trabajos realizados o de forma privada para que solo el propietario pueda acceder al código.

El uso que se le ha dado a este repositorio se podría diferenciar en dos tareas:

- Mantenimiento y gestión del código.
- Planificación de tareas haciendo Milestones.

4.6. SurveyMonkey

SurveyMonkey [24] es una empresa que ofrece vía web la posibilidad a los usuarios de realizar encuestas online de forma dinámica. SurveyMonkey tiene una base de datos con cientos de preguntas predefinidas que los usuarios pueden utilizar en sus propias encuestas.

A medida que los usuarios vayan rellenando las encuestas SurveyMonkey nos irá generando, de forma automática, los resultados de las encuestas realizadas creando gráficas sobre los datos obtenidos.

Se ha utilizado este aplicativo para crear una encuesta que evalúe las características del proyecto para que una vez que el usuario lo haya testeado pueda dar su propia opinión sobre el proyecto.

5. Implementación

En este capítulo entraremos en detalle de algunos elementos implementados en el proyecto que he creído interesante profundizar para entender como se ha realizado.

5.1. HUD

El HUD (Heads-Up-Display) de los videojuegos hace referencia a los elementos que se muestran en pantalla dando información al usuario de elementos como la cantidad de vida, la puntuación, número de armas etc. En el proyecto se han implementado una serie de elementos como son la barra de vida, la munición de cada arma, punteros y puntuaciones.

Para poder desarrollar estos elementos se ha utilizado un nuevo sistema que incorpora Unity 3D en las últimas versiones llamado Canvas. El Canvas es un GameObject con un componente Canvas donde todos los elementos de la UI (User Interface) deberían estar dentro para pintarse en la Escena.

5.1.1. HealthBar

Para el desarrollo de la HealthBar de los jugadores se diseñó un boceto de barra de vida que posteriormente se pasó a Illustrator y Photoshop dando el siguiente resultado (véase Figura 5.1).

Figura 5.1: HealthBar 1P Zombie Assault.

Para hacer funcional la HealthBar se tuvo que descomponer cada elemento por separado para dotar de comportamiento a cada una de las partes (véase Figura 5.2).

Figura 5.2: Partes del diseño de la HealthBar. De izq. a der. Tipo bala por muerte, barra de vida, base de barra de vida con score y tambor con nivel de combo.

El tambor de la HealthBar contendrá el nivel del combo del personaje e irá girando a medida que vayamos eliminando enemigos añadiendo dentro un tipo de bala según si ha eliminado al enemigo de headshoot o no. Por cada muerte, efectuada, la puntuación irá aumentando en relación al nivel actual de combo. La barra de vida irá disminuyendo cada vez que un enemigo nos ataque reiniciándonos el nivel de combo a 1.

5.1.2. Munición

Un elemento importante en este tipo de juegos es que el jugador necesita saber el número de balas que le quedan en todo momento para poder recargar antes de quedarse sin. Cada jugador tendrá un tipo de arma diferente con características como son fuerza de tiro, velocidad de recarga, número de balas etc.

El tipo de balas que se han utilizado han sido del tipo pistola, revólver, escopeta y metralleta (véase Figura 5.3).

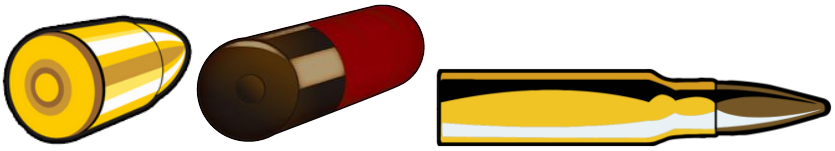


Figura 5.3: Imagen de la munición de pistola/ revolver, escopeta y metralleta.

Cada una de estas balas es un objeto prefabricado “prefab” que contiene un componente “Image”, donde asignaremos la imagen de la bala, que al iniciar la escena creará tantas balas como marque el cargador. Por cada vez que se dispare con el arma la bala hará un efecto de rotación en descendencia (véase Figura 5.4) simulando el vaciado del cargador hasta que nos quedemos sin balas. Cuando no tengamos más balas nos aparecerá un mensaje en pantalla pidiéndonos que “Recarguemos”, al recargar las balas volverán a la posición inicial de la primera instancia del objeto.

Figura 5.4: En la siguiente imagen mostramos el efecto de descarga del cargador del 2P.

5.2. Menús

El uso de menús es uno de los elementos más importantes de un videojuego ya que será el medio por el que los jugadores podrán interactuar con las opciones de este para poder acceder a niveles de juego, ajustes etc.

En este proyecto se han implementado varios tipos de menús entre ellos menú principal, menú de pausa y menú de opciones (véase Figura 5.5).

Para poder desarrollar estos menús se ha utilizado un GameObject que hará la función de controlador de la escena, GameEngine. Este GameObject estará analizando cada una de las acciones del jugador como puede ser: pulsar un botón del periférico, cambiar alguna opción del menú, hacer que el juego pare su ejecución una vez dentro del menú de pause etc.

Figura 5.5: Imagen que muestra los diferentes menús del juego.

5.3. Periféricos

Uno de los propósitos de este proyecto era el poder usar algún tipo de periférico que nos permitiese comunicarnos con el videojuego. De modo que hemos implementado varios GameObjects con el comportamiento adecuado según el periférico que vayamos a utilizar, recordamos que los periférico escogidos fueron el mando de PS3 y el WiiMote.

Para permitir la comunicación entre el periférico y el proyecto se ha creado una pequeña API que contiene llamadas, a métodos y funciones, que acceden a cada uno de los inputs comprobando el funcionamiento de este.

Una vez comprobada que la detección de los inputs es correcta, se ha procedido a la implementación del GameObject, que se usará como mirilla en el juego, dotándolo de un comportamiento, creado a partir de la API generada, haciendo que responda a cualquiera de los eventos implementados.

De esta forma se ha conseguido que a partir del periférico podamos mover el GameObject de la mirilla del jugador a través de la pantalla o para otras utilidades como disparar, recargar, cambiar ajustes etc.

5.4. Personajes

Como en todos los juegos existen personajes que serán aquellos que acompañen al jugador a través de la historia. Por este motivo vamos a explicar cómo hemos creado a nuestros personajes las propiedades y acciones que le hemos añadido.

Antes de nada aclarar que hemos cogido unos modelos ya creados [25] y se han editado con Blender para que le queden visibles solamente los brazos ya que es lo único que se verá cuando el jugador quiera recargar o lance una granada (véase Figura 5.6).

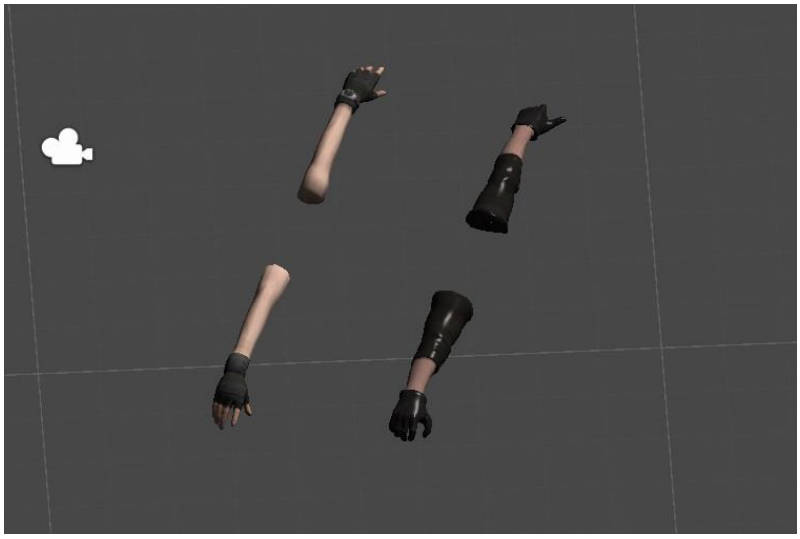


Figura 5.6: En la siguiente imagen muestra el modelo de los brazos de cada player.

El siguiente paso es dotar a estos brazos de huesos mediante “rigging” para poder añadirlos a la estructura del modelo así que una vez tenga huesos podremos crear animaciones con los movimientos del jugador. Para realizar esta operación, como en el caso anterior, desde Blender le añadiremos un esqueleto ya predefinido por este e iremos eliminando los huesos que no nos interesen a la vez que editamos los huesos que queremos para poder adaptarlo al modelo (véase Figura 5.7).

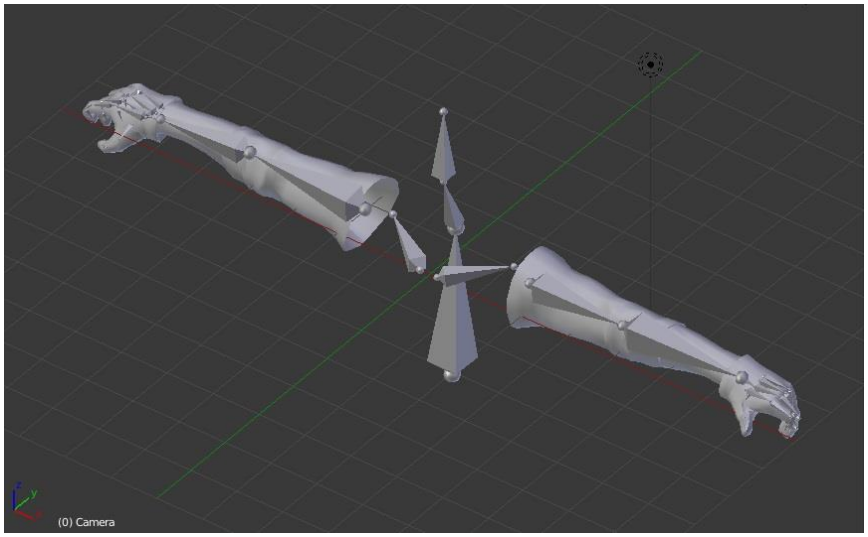


Figura 5.7: Imagen donde se muestra el modelo del personaje con los huesos aplicados.

Este procedimiento es una operación delicada porque si no se ajustan bien los huesos al modelo tendremos problemas con los vértices del este a la hora de realizar las animaciones del jugador.

Para finalizar la implementación del jugador, solo nos faltaría el proceso de animación que permitirá al personaje realizar una serie de movimientos que nosotros le hayamos preestablecido. Cargaremos el modelo en Unity 3D y desde la herramienta de Animation [26] empezaremos a animar diciendo el número de frames que va a durar y creando movimiento a partir de translaciones y rotaciones que hagamos sobre los huesos del modelo (véase Figura 5.8).

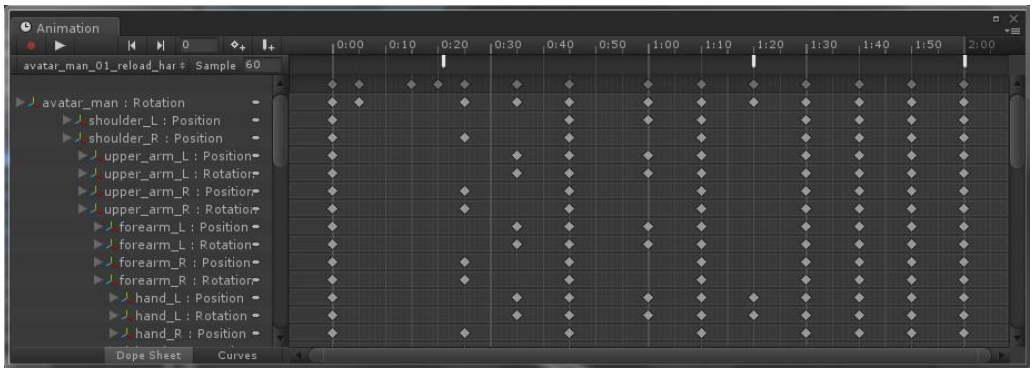


Figura 5.8: En esta imagen se muestra la animación de recargar del modelo.

5.5. I.A.

La inteligencia artificial en los juegos es el comportamiento que tienen los NPC para actuar de una manera u otra según el personaje que tenga delante. En esta sección vamos a hablar de la forma en que se ha implementado la I.A. de los enemigos y las técnicas que se han utilizado para llevarla a cabo.

Empezaremos hablando de la I.A. de los enemigos comunes y después continuaremos con la del Boss. El tipo de I.A implementada en ambos casos es de máquinas de estados con transiciones para poder ir de un estado a otro y con sub-estados para que dentro de un estado pueda escoger diferentes acciones que presente dicho estado.

La máquina de estados de los NPC (véase Figura 5.9) empezaría desde el estado “idle” que sería el punto de partida para empezar y pasaría al estado de “Initial Pose” donde de forma automática se le asignaría uno de los sub-estados que contiene. Una vez finalice la animación pasará al estado de “Walk&Run” que según el nivel de dificultad del juego el enemigo ejecutará el sub-estado de caminar o correr hacia el personaje. Cuando el enemigo este lo suficiente cerca de nuestros personajes se parará delante nuestro y pasará al estado de “Attack” propinándonos un golpe o mordisco.

Cuando el personaje dispare al enemigo se cancelará el estado que este ejecutando y pasará directamente al estado de “Hit”, cuando haya finalizado la animación de este estado volverá al último estado ejecutado para continuar con el proceso de llegar hasta el personaje. Si golpeamos varias veces al enemigo y el comportamiento de este nos avisa que está muerto pasará de manera directa al estado de “Death” ejecutando la animación de muerto

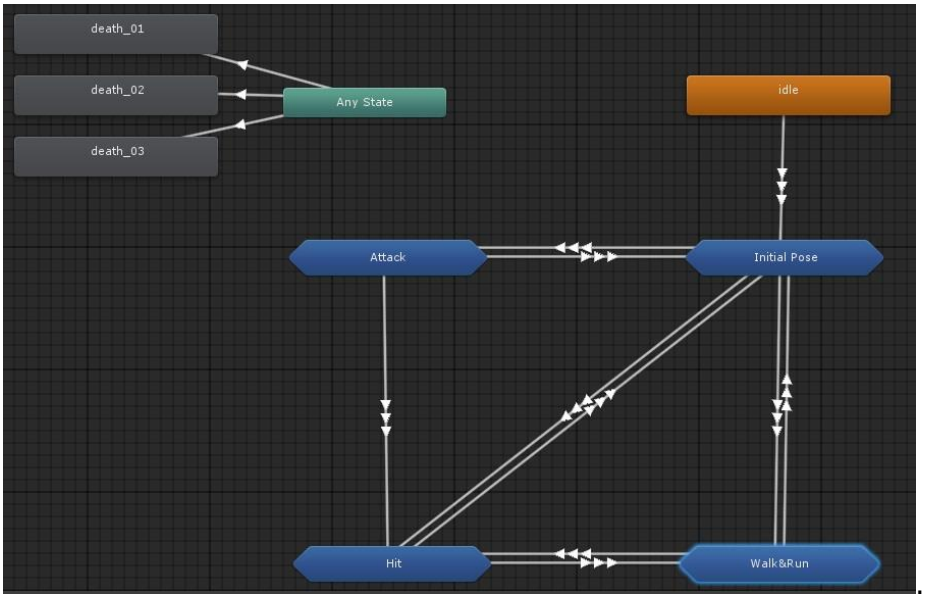


Figura 5.9: En esta imagen se muestra la máquina de estados utilizada por los NPCs.

La I.A. del Boss utiliza la misma máquina de estados que la de los NPCs la diferencia es que esta actúa sobre una matriz de “nodos” (véase Figura 5.10) por la que crea un camino para llegar al personaje y poder atacarle. La manera de usar la máquina de estados es algo diferente ya que podemos cancelar el ataque del Boss infligiéndole mucho daño y de esta forma vuelva al punto de partida por el que empezó para crear una nueva ruta para atacar al personaje.

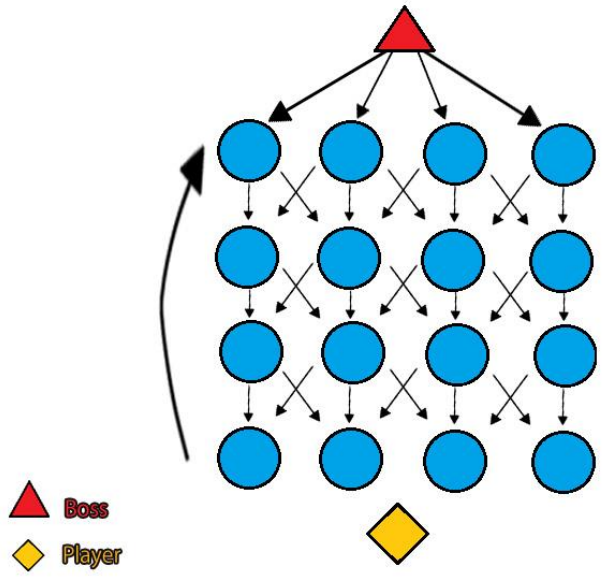


Figura 5.10: La siguiente imagen muestra la matriz de nodos que el Boss utilizará.

5.6. Colliders

Para poder dotar a nuestros objetos con físicas es necesario el uso de Colliders. Los Colliders definen la forma de los objetos a los que están asociados dotándolos de físicas que reaccionarán al contacto con otros objetos.

En el proyecto se han utilizado los Colliders para la detección de las partes de los enemigos cuando el jugador les dispara. Se han cogido los modelos de los enemigos y se les han ido añadiendo colliders en forma de cubo (véase Figura 5.11) a cada una de las extremidades del cuerpo de manera que cuando disparemos seamos capaces de diferenciar un brazo de una pierna.

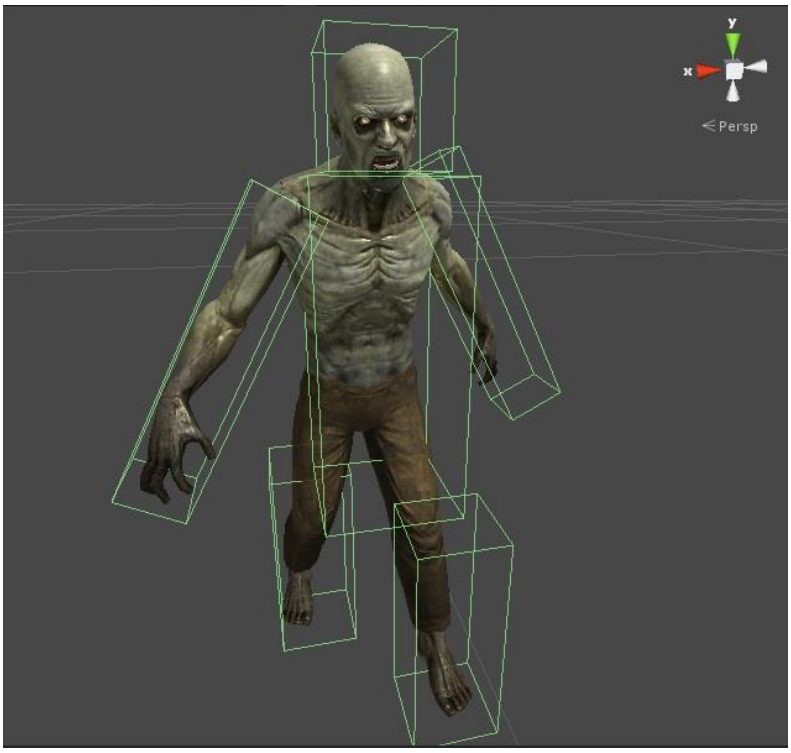


Figura 5.11: En la siguiente figura se muestra al enemigo con los colliders aplicados en cada parte del cuerpo [27].

El efecto que provocará esto en el enemigo será el desmembramiento de esa parte del cuerpo causando la perdida de este y según la extremidad destruida habrá un cambio en la I.A. del enemigo haciendo que pase de caminar a arrastrarse por la falta de alguna pierna o que de atacar con los brazos solo muerda por no tener ninguno de los dos.

5.7. Iluminación

En los videojuegos uno de los factores importantes que existe para que haya una buena ambientación es la iluminación. Se han utilizado algunas características que nos ofrece Unity 3D para poder crear efectos lumínicos en nuestras escenas incluso pre-renderizados de luces para poder crear sombras.

En este proyecto se han utilizados luces de tipo “Directional Light”, “Spot Light”, “Halos” y “Lightmapping”.

Las luces tipo “Directional Light” emitirán luz en la dirección en la que se indique haciendo que la parte que este dentro de la dirección del haz de luz este iluminada mientras que la que se encuentra fuera este menos. Este tipo de luz ha sido utilizada principalmente para poder iluminar las escenas de los niveles ya que elevando un poco la intensidad lumínica del

componente es capaz de iluminar toda la escena sin consumir muchos recursos. El único inconveniente es que no genera sombras en los objetos que incide la luz.

Las luces tipo “Spot Light” es un tipo de luz fija que emite haces de luz en todas direcciones y se caracteriza por la forma cónica que tiene para emitir los rayos de luz. Este tipo de luz ha sido utilizado en el proyecto para crear la iluminación de las farolas de una de las escenas del proyecto. También se ha implementado conjuntamente a este tipo de luz los “Halos” que son áreas de luz que se colocan cerca de fuentes de estas dando el efecto de pequeñas partículas de luz.

Finalmente el “Lightmapping” es un procesado que tiene Unity 3D que es capaz de renderizar la luz para crear sombras de toda la escena. Esto es un proceso muy costoso para la máquina ya que en cada frame deberá calcular la sombra de cada uno de los elementos de la escena.

Este proceso da unos resultados extraordinarios, pero por temas de rendimientos me he visto obligado a quitarlo aunque había pensado que era interesante comentarlo ya que es una buena característica que ofrece el motor.

5.8. Partículas

Las partículas son un aspecto muy común en los videojuegos donde se trata de recrear algún tipo de efecto dada la circunstancia. En este proyecto se han utilizado dos tipos de partículas el “Particle Emitter” y un derivado de este el “Ellipsoid Particle Emitter”. Ambos sistemas trabajan de la misma manera, la única diferencia es la forma de emitir las partículas ya que el “Ellipsoid” es capaz de generar partículas dentro de la esfera en que lo envuelve.

Para crear nuevos sistemas de partículas que se adapten al proyecto se han cogido partículas ya creadas por el Asset de Unity Elementals [28]. Se han creado partículas de sangre para cuando disparemos a los enemigos y charcos de sangre para cuando estos mueran (véase Figura 5.12).

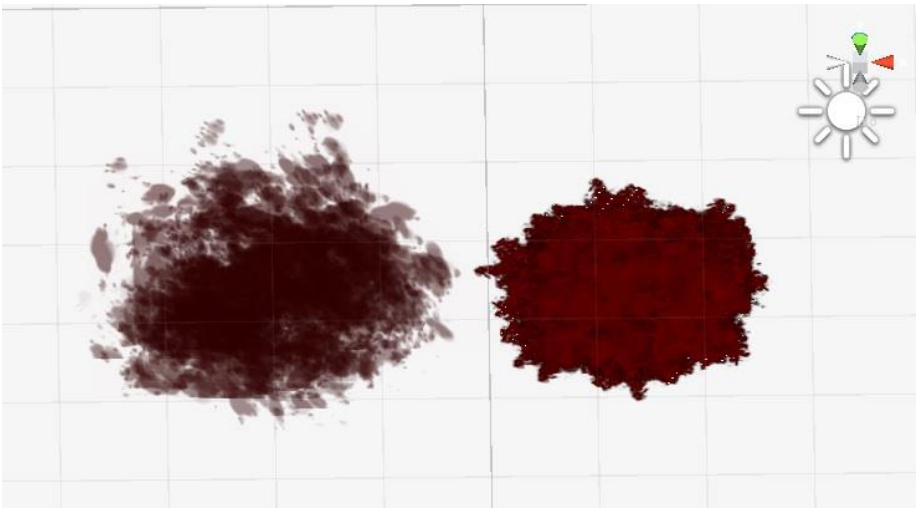


Figura 5.12: En la siguiente figura se muestra a mano izquierda el efecto sangrado al disparar y a la derecha el efecto desangrado del enemigo al morir.

Otro tipo de efectos que se han creado han sido para los objetos destruibles de la escena como el efecto de chispas al destruir la lámpara de una farola, el chorro de agua al destruir una boca de incendios o el de explosión al reventar un barril de gasolina (véase Figura 5.13).

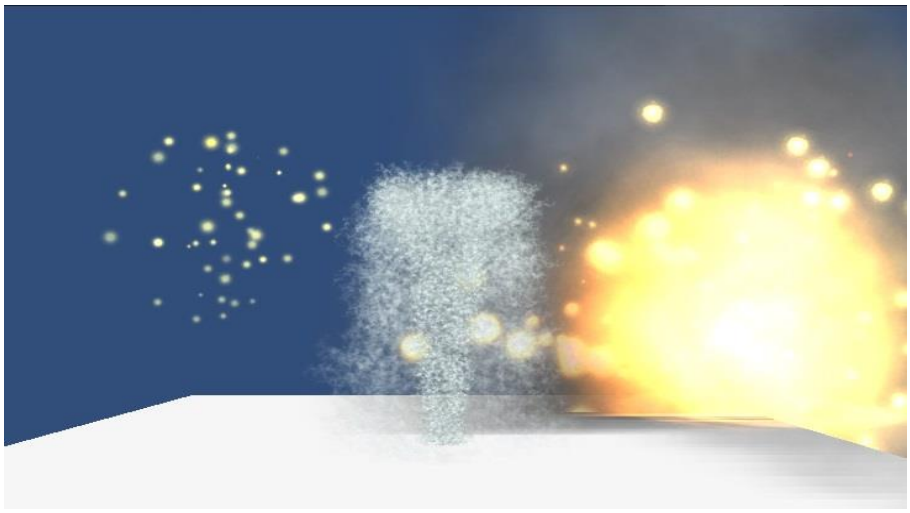


Figura 5.13: En la siguiente figura se muestran los efectos de chispas, chorro de agua y explosión.

.

5.9. Cámara

Otro elemento importante a la hora de desarrollar cualquier juego es la cámara. La cámara, en los videojuegos, muestra al jugador una particular visión del entorno gráfico en el que se sitúa el juego. Según el tipo de juego pueden haber varios tipos de cámara, en este proyecto se ha utilizado una cámara en primera persona con vista perspectiva para poder mostrar todo lo que el personaje estaría viendo y de esta forma tener un mayor campo de visión.

Uno de los problemas que surgió a la hora de crear la cámara fue el tener en cuenta el tipo de resolución del monitor en el que se ejecutaba el proyecto. Por eso se modificó el *script* de Pierre Semaan [29] en el que nos permitía escoger el tipo de resolución, entre 4:3 para los monitores cuadrados y 16:9 para monitores con forma panorámica (véase Figura 5.14), para poder mostrar de forma escalada todos los objetos de la escena.

Figura 5.14: La siguiente imagen muestra la resolución del proyecto en 16:9 (izq.) y 4:3 (der.) en una pantalla panorámica.

Este script nos detecta que si la resolución no era del tipo especificado nos adapta la pantalla añadiendo unas franjas negras tanto laterales o frontales haciendo encajar la imagen según la resolución establecida de tal forma que quede como la resolución deseada.

5.10. Rail Engine

Como se ha comentado anteriormente el desplazamiento de los juegos ‘Shooter on Rails’ se hace sobre un rail que va moviendo al jugador por todo el mapeado de la escena.

Para poder recrear este tipo de rail en el proyecto se ha utilizado un Asset [30] en el que incorpora una serie de objetos que permiten realizar el desplazamiento entre diferentes puntos de la escena mediante nodos y splines. La unión de dos nodos da lugar a un spline que es lo que llamaremos rail (véase Figura 5.15).

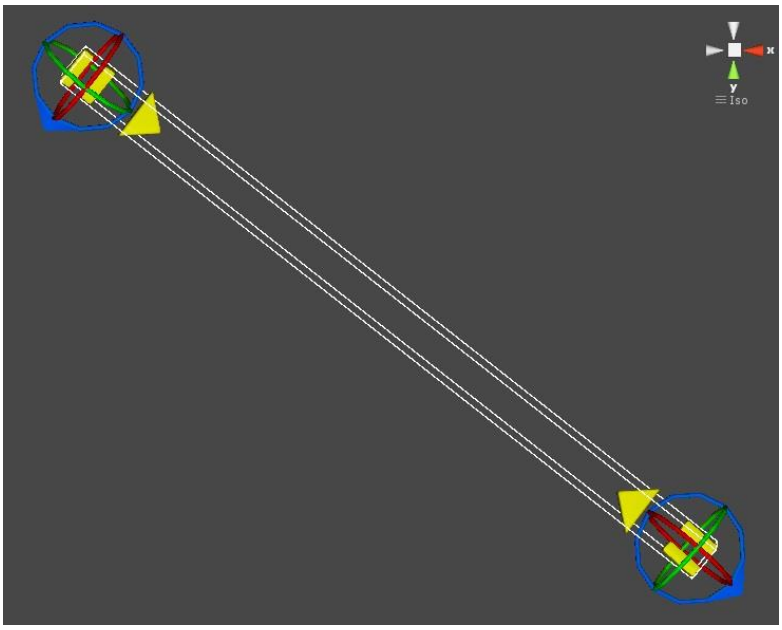


Figura 5.15: La siguiente imagen muestra la unión de dos nodos dando lugar a un rail.

La cámara será el objeto que se irá moviendo por la escena junto a los modelos de los brazos de los jugadores. Esta debe tener el componente del controlador del spline y se le deberá asignar un nodo de inicio. De esta forma se consigue que la cámara realice el recorrido por los raíles que se han creado.

Cada uno de los nodos tiene una serie de acciones a realizar cuando el objeto que tiene el controlador llega al nodo. Las opciones que tiene son: continuar, pausa, pararse o marcha atrás. Estas acciones se han aprovechado para simular la llegada de hordas donde el personaje al llegar a un nuevo nodo se quedará parado hasta que haya eliminado a todos los enemigos de esta forma podrá continuar hacía el siguiente nodo.

5.11. Optimización del juego

Algo que se tiene que tener en cuenta cuando se está desarrollando un videojuego es el número de recursos que se necesitan para poder funcionar de manera óptima. En este

proyecto se han utilizado algunas técnicas y configuraciones para poder optimizar parte del videojuego y que de esta forma vaya más fluido evitando pérdidas del framerate.

Si tenemos objetos en nuestra escena que no se van a mover y van actuar de manera estática una de las recomendaciones que se da es marcar ese objeto como *Static* (véase Figura 5.16). Con esta acción lo que se consigue es que dichos objetos no se vuelvan a recalcular en cada frame.

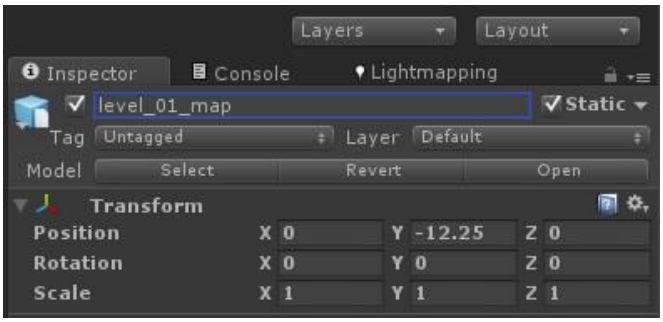


Figura 5.16: La siguiente imagen muestra un GameObject de tipo estático.

Otro tipo de recomendación que se da es evitar el uso excesivo de *Draw Calls.* Los Draw Calls son el número de llamadas que se hace a la GPU para volver a pintar la textura de un objeto. Para reducir lo máximo posible el número de estas se recomienda el uso de materiales que se puedan reaprovechar para varios objetos y de Mapas de Atlas que contengan todas las texturas de un solo objeto. De esta forma se consigue reducir el número de Draw Calls haciendo que el batching pueda renderizar todas estas llamadas en una sola (véase Figura 5.17).

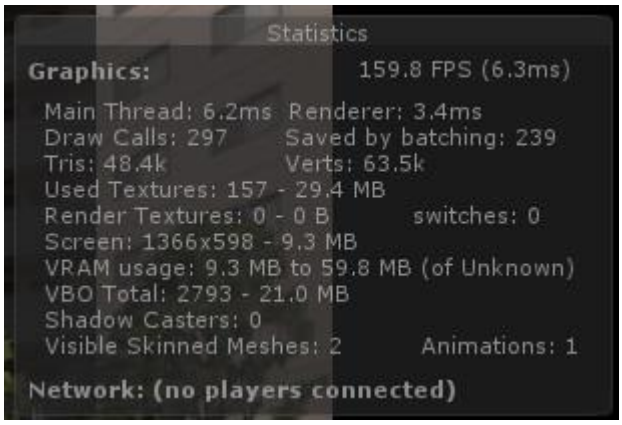


Figura 5.17: La siguiente imagen muestra el número de Draw Calls de la escena y los draws guardados por batching.

Una de las características que incluye la versión Pro de Unity es la herramienta de *Oclussion Culling.* Esta herramienta realizará un preprocesado de cálculos de oclusión en los que irá analizando todos elementos de la escena desde el punto de vista de la cámara hasta la distancia que nosotros le especifiquemos ocultando todos los elementos que no estén dentro del rango de la cámara evitando de esta manera cálculos y draws innecesarios.

6. Planificación del proyecto

En este capítulo se explicará el tipo de metodología que se ha escogido para realizar el trabajo y la planificación de las tareas del proyecto en formato de diagrama de Gannt. También hablaremos de cuánto costaría realizar este proyecto si quisiéramos venderlo y formas de poder subvencionarlo como las famosas ‘kickstarters’.

6.1. Metodología Scrum

La metodología utilizada en el proyecto ha sido *Scrum*. Scrum es una metodología de trabajo orientada en hacer sprints cada 2 o 3 semanas realizando una serie de tareas que deberán completarse. Al final de cada sprint se muestra un release del trabajo realizado para evaluar posibles errores y mejoras.

Esta metodología se ha podido llevar a cabo gracias al gestor de código Github [23] que nos permite la creación de tareas y sprints.

6.2. Planificación

En esta sección mostraremos mediante diagramas de Gantt como se ha planificado todo el trabajo del proyecto. La planificación de proyecto está pensada en sprints y en cada sprint las tareas a realizar.
Primeramente explicaré como había pensado el desarrollo de la planificación inicial (planificación ideal) y después pasaré directamente al diagrama de Gantt que reflejará la planificación real de todo el proyecto.

6.2.1 Planificación ideal

A continuación se muestra de manera detallada toda la planificación que se creó al comenzar el proyecto. En esta planificación habría separado cada tarea por categorías según el tipo de desarrollo.

Sprint 0

Research:

- Pruebas y análisis de los periféricos.

Sprint 1 (23-02-15, 08-03-15)

Code

- Controlador básico para PC.
- Controlador básico para PS3.
- Controlador básico para WiiMote.

Design

- Barra de vida de los jugadores.
- Munición de las armas.
- Granadas.
- Punteros de la mirilla.

Sprint 2 (9-03-15, 22-03-15)

Code
- Script Barra de vida.
- Script puntuación.

Model 3D
- Edición de modelo del jugador.
- Rigging.

Animations
- Animación de recargar.
- Animación de lanzar granada.

Effects
- Partículas de efecto al disparar.

Audio
- Efectos de audio al disparar.

Bugs
- Reparar bugs del sprint anterior.

Sprint 3 (23-03-15, 5-04-15)

Code
- Script IA enemigos.
- Script comportamiento enemigos.
- Script añadir Player 2.
- Script Audio Engine.

Design
- Texto de “Recargar”

Audio
- Efectos de audio de enemigos.
- Audio “Recarga”.

Bugs
- Reparar bugs del sprint anterior.

Sprint 4 (06-04-15, 19-04-15)

Scene
- Implementar Level 01.

Code
- Actualizar script PC, PS3, WiiMote controllers.
- Script Menú Pausa.
- Script GameOver.

Models 3D
- Editar modelo de enemigo.

Effects
- Sistema de partículas de entorno: Explosiones, chispas, fuentes de agua etc.

Hardware
- Crear barra sensora IR.

Bugs
- Reparar bugs del sprint anterior.

Sprint 5 (20-04-15, 03-05-15)

Scene
- Implementar Rail.
- Implementar posibilidad de elegir caminos.

- Implementar posibilidad de salvar supervivientes.

Code

- Script Engine Level 01.
- Script Menú Principal.
- Script Menú Opciones.
- Script Pantalla de Carga.

Designs

- Diseño del Menú Principal.
- Diseño Menú Opciones.
- Diseño Pantalla de Carga.

Bugs

- Reparar bugs del sprint anterior.

Sprint 6 (04-05-15, 17-05-15)

Scene

- Implementar MiniGame, posibles opciones: Sala de tiro, Supervivencia, Contrareloj.

Code

- Script Engine MiniGame.
- Script GameCredits.
- Script Logros Engine.

Designs

- Diseño de los créditos del juego.
- Diseño de los logros del juego.

Bugs

- Reparar bugs del sprint anterior.

Sprint 7 (18-05-15, 31-05-15)

Scene

- Implementar Multijugador.

Code

- Script Multiplayer Engine.
- Script Menú Multiplayer.

Designs

- Diseño Menú Multiplayer..

Bugs

- Reparar bugs del sprint anterior.

Sprint 8 (01-06-15, 14-06-15)

Scene

- Implementar Level 02.

Code

- Script Engine Level 02.

Bugs

- Reparar bugs del sprint anterior.

Sprint 9 (15-06-15, fecha de entrega)

- Documentación.
- Presentación.
- Tráiler

6.2.2 Planificación real

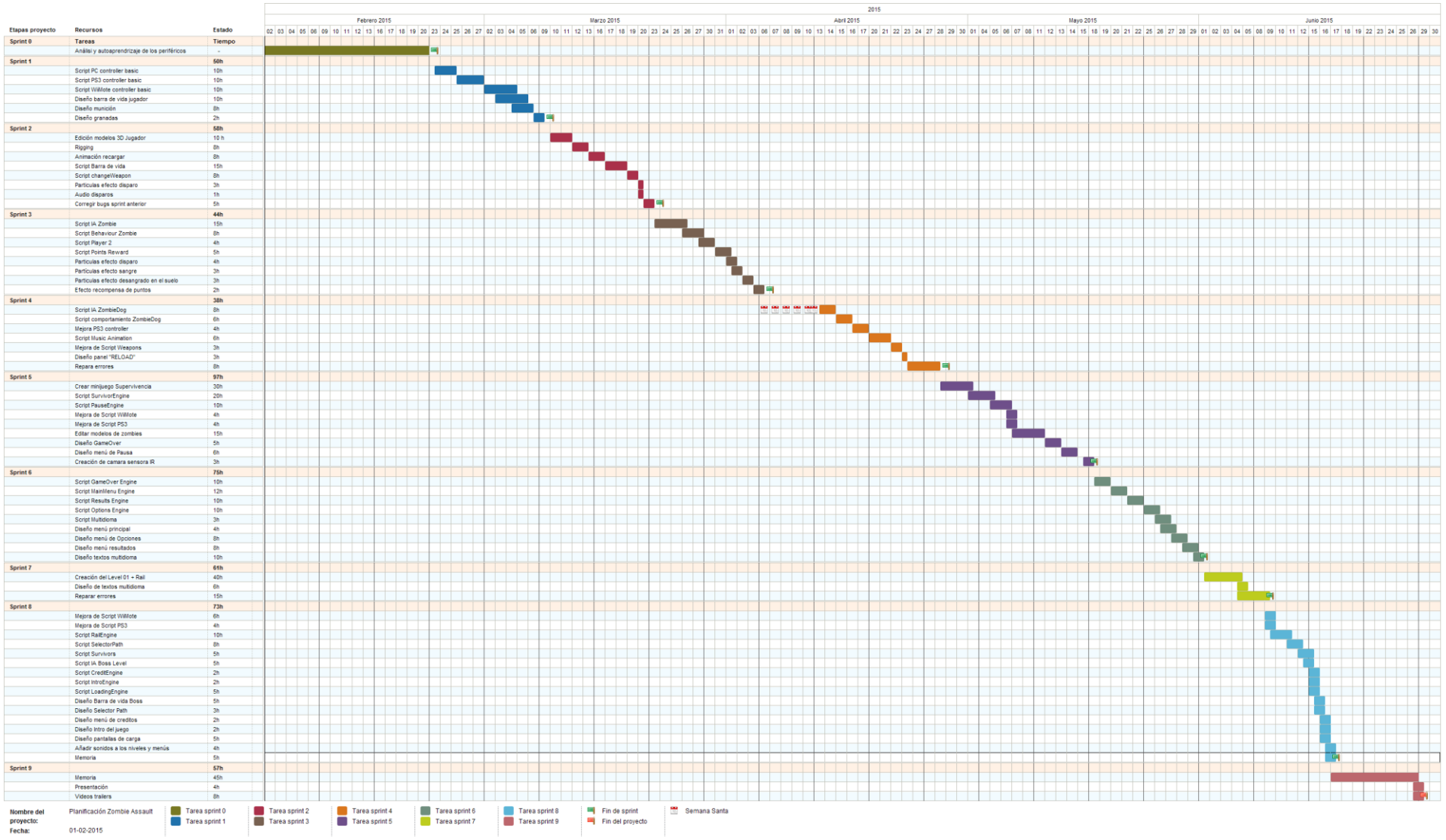


Figura 6.1: La siguiente imagen muestra la planificación según el diagrama de Gantt.

6.3 Costes del proyecto

Para poder desarrollar este proyecto se ha hecho una evaluación económica de cuanto necesitaríamos invertir para que el proyecto saliese a la luz. Se ha realizado un estudio de mercado sobre los profesionales, que empiezan en el sector del videojuego, para saber cuánto sería su salario base, aproximadamente, en las categorías de desarrollo que se necesitaría.

Los salarios que se presentan a continuación [31][32], son los salarios de los profesionales Junior con menos de 3 años de experiencia en el sector:

- Programador 48600€/anual bruto.
- Diseñador modelador 40000€/anual bruto.
- Artistas de efectos y sonidos 29200€/anual bruto.
- Tester 21500€/anual bruto.

Si desglosamos estos salarios según el sistema que se aplica en España, se debería descontar aproximadamente un 5% de Seguridad Social y otro, como mínimo, 5% del IRPF después fraccionarlo en 14 pagas anuales y finalmente dividirlo entre las 160 horas de la jornada laboral del mes. El resultado que obtendríamos sería el precio hora, en neto, de cada uno de los profesionales:

- Programador 19.5€/h netos.
- Diseñador modelador 16€/h netos.
- Artistas de efectos y sonidos 11.8€/h netos.
- Tester 8.7€/h netos.

Ahora con estos valores vamos hacer una previsión de las horas que se han dedicado a desarrollar el juego para calcular la inversión que se necesitaría para llevar a cabo el proyecto.

Categoría	**Horas**	**€/Horas**	**Precio (€)**
Programación del proyecto	251h	19.5€/h	4895€
Diseño y modelador del proyecto	205h	16.0€/h	3280€
Efectos y sonidos	20h	11.8€/h	236€
Tester	60h	8.7€/h	522€
Márquetin y publicidad			1000€
Herramientas de desarrollo			225€
Licencias			500€
Total	**536h**		**9758€**

El coste total del proyecto realizado sería alrededor de unos 10.000€ netos. Pero ahora la pregunta es, como podría financiar esto?. Recientemente han aparecido las famosas webs *Kickstarter*, que se dedican a la financiación de proyectos creativos donde los usuarios depositan parte de la ayuda económica para que el proyecto pueda ver la luz algún día.

Uno de los ejemplos más claros de este tipo de financiación ha sido el video que en 1 día tuvo más de 10M de reproducciones en Youtube, Kung Fury [33] o el desarrollador de la saga Shenmue que en este último E3 ha pedido ayuda a los fans para poder desarrollar una nueva entrega del título [34].

Este tipo de financiaciones recompensa a todos aquellos que han participado con colaboraciones dentro del proyecto o con parte del material de desarrollo.

7. Resultados

En este capítulo vamos a presentar los resultados que se han obtenido a lo largo del proyecto. Por eso se van a mostrar imágenes del juego in-game y vamos a analizar las encuestas realizadas a las personas que, muy amablemente, han podido testear el proyecto pudiendo dar su opinión de algunos de los apartados técnicos del juego.

Estas son algunas de las imágenes extraídas de Zombie Assault (véase Figuras 7.1, 7.2, 7.3, 7.4, 7.5, 7.6, 7.7, 7.8, 7.9).

Figura 7.1: La siguiente imagen muestra pantalla del menú principal.

Figura 7.2: La siguiente imagen muestra el menú de opciones.

Figura 7.3: La siguiente imagen muestra los controles del juego para PC.

Figura 7.4: La siguiente imagen muestra las opciones del menú de opciones.

Figura 7.5: La siguiente imagen muestra la pantalla de cargar para acceder al nivel o supervivencia.

Figura 7.6: La siguiente imagen muestra el nivel 01 junto al menú de pausa.

Figura 7.7: La siguiente imagen muestra el inicio del modo supervivencia.

Figura 7.8: La siguiente imagen muestra el Fin del juego del jugador.

Figura 7.9: La siguiente imagen muestra la pantalla de créditos del juego.

A continuación vamos a poner los resultados de las encuestas que han rellenado los testers con la crítica del juego.

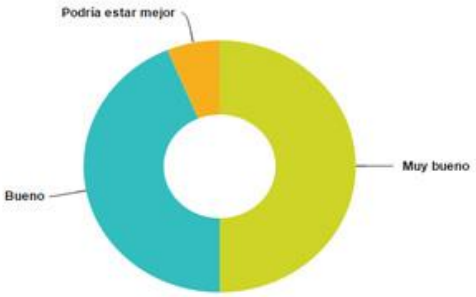
Que te ha parecido la interfaz de usuario?

Respondido: 16 Omitido: 0

Opciones de respuesta	Respuestas	
Muy buena	50,00%	8
Buena	50,00%	8
Podría estar mejor	0,00%	0
N/C	0,00%	0
Total		16

Que te ha parecido el apartado gráfico?

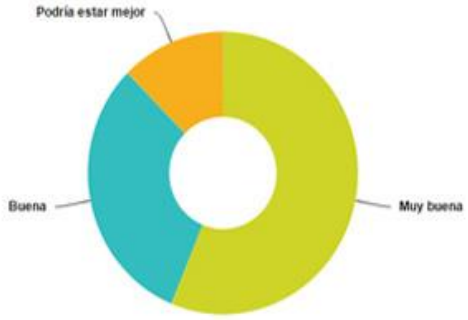
Respondido: 16 Omitido: 0



Opciones de respuesta	Respuestas	
Muy bueno	50,00%	8
Bueno	43,75%	7
Podría estar mejor	6,25%	1
N/C	0,00%	0
Total		16

Que te ha parecido la jugabilidad del juego?

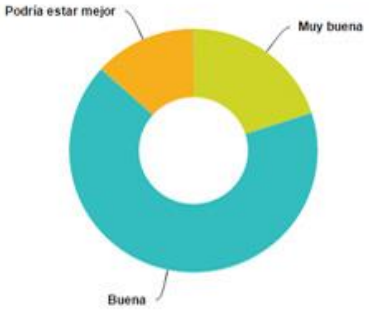
Respondido: 16 Omitido: 0



Opciones de respuesta	Respuestas	
Muy buena	56,25%	9
Buena	31,25%	5
Podría estar mejor	12,50%	2
N/C	0,00%	0
Total		16

Que te ha parecido la durabilidad del juego?

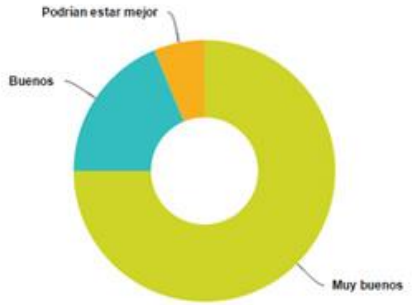
Respondido: 15 Omitido: 1



Opciones de respuesta	Respuestas	
Muy buena	20,00%	3
Buena	66,67%	10
Podría estar mejor	13,33%	2
N/C	0,00%	0
Total		15

Que te han parecido los efectos de sonido?

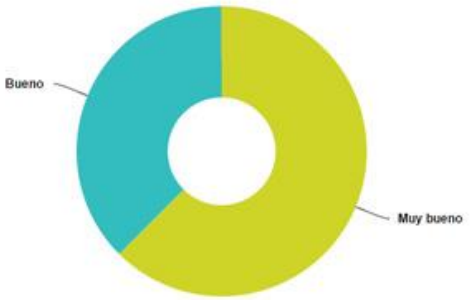
Respondido: 16 Omitido: 0



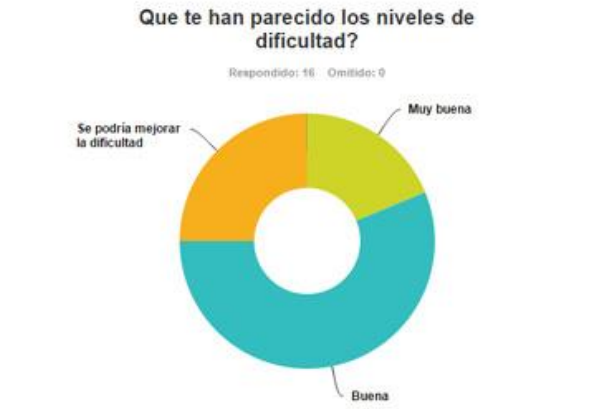
Opciones de respuesta	Respuestas	
Muy buenos	75,00%	12
Buenos	18,75%	3
Podrían estar mejor	6,25%	1
N/C	0,00%	0
Total		16

Que opinas del framerate del juego?

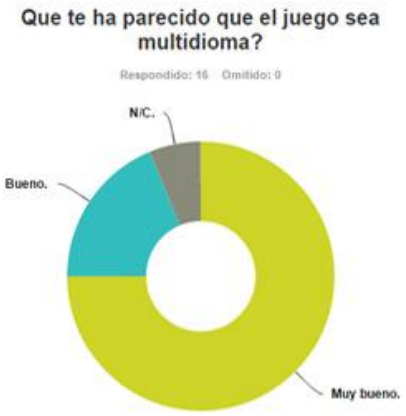
Respondido: 16 Omitido: 0



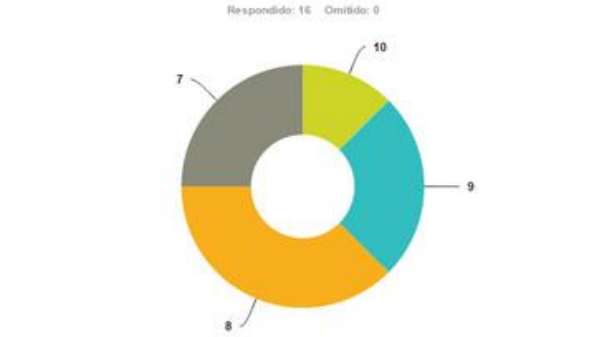
Opciones de respuesta	Respuestas	
Muy bueno	62,50%	10
Bueno	37,50%	6
Podría estar mejor	0,00%	0
N/C	0,00%	0
Total		16



Opciones de respuesta	Respuestas	
Muy buena	18,75%	3
Buena	56,25%	9
Se podría mejorar la dificultad	25,00%	4
Se podría reducir la dificultad	0,00%	0
N/C	0,00%	0
Total		16



Opciones de respuesta	Respuestas	
Muy bueno.	75,00%	12
Bueno.	18,75%	3
Habría añadido más idiomas.	0,00%	0
N/C.	6,25%	1
Total		16



Opciones de respuesta	Respuestas	
10	12,50%	2
9	25,00%	4
8	37,50%	6
7	25,00%	4
6	0,00%	0
5	0,00%	0
<5	0,00%	0
Total		16

Los resultados extraídos son bastante favorables ya que la valoración media es muy positiva y parece que el juego ha gustado. Algunos de los encuestados han dado la opinión sobre algunos problemas que han encontrado para poder mejorarlos en una futura revisión.

8. Conclusiones y trabajo futuro

Después de todo el proceso de desarrollo de Zombie Assault se podría decir que se ha conseguido aquello que se quería realizar. Inicialmente se había concebido este proyecto como el reto de poder crear un videojuego, desde cero, utilizando todo tipo de técnicas y herramientas como lo haría un profesional.

Personalmente pienso que el resultado obtenido es realmente bueno, para alguien que no es experto en el sector, ya que se ha sabido sacar parte del potencial del motor Unity 3D para la realización del proyecto utilizando algunas de las técnicas que utilizan, normalmente, los desarrolladores de videojuegos.

También se ha conseguido el reto de poder establecer comunicación entre varios periféricos de marcas de consolas conocidas como son SONY o Nintendo, haciendo que la experiencia del juego sea mayor para el jugador.

El proceso de llevar a cabo este proyecto ha sido largo y costoso pero estoy realmente orgulloso del resultado final obtenido de Zombie Assault. La idea principal era poder realizar un juego sobre raíles utilizando periféricos pero en ningún momento llegué a pensar en poder realizar algo como este proyecto.

El estado actual del proyecto no lo daría como finalizado, simplemente pienso que esto es un base para poder seguir trabajando en un futuro. Al proyecto se le pueden seguir añadiendo mejoras como:

- Nuevos niveles para que la durabilidad del juego sea más larga.
- Nuevos modos de juego.
- Implementar el Multijugador.
- Implementar sistema de logros.
- Implementar diálogos del personaje.
- Etc.

Como he descrito creo que hay trabajo para implementar de sobras para poder realizar otro proyecto a partir de este.

Bibliografía

[1] Pong : ¡El Primer Videojuego De La Historia! **[Online] [Accessed: 08-June-2015]**
Available: https://www.youtube.com/watch?v=Yruv7QUiOJA

[2] Dualshock Wireless Controller **[Online] [Accessed: 15-Feb-2015]**
Available: https://www.playstation.com/en-us/explore/accessories/dualshock-3/

[3] Nintendo WiiMote **[Online] [Accessed: 15-Feb-2015]**
Available: https://es.wikipedia.org/wiki/Wiimote/

[4] Retro Informática el pasado del futuro **[Online] [Accessed: 09-June-2015]**
Available: http://www.fib.upc.edu/retro-informatica/historia/videojocs.html/

[5] Nolan Bushnell - Wikipedia **[Online] [Accessed: 09-June-2015]**
Available: https://en.wikipedia.org/?title=Nolan_Bushnell

[6] Atari **[Online] [Accessed: 09-June-2015]**
Available: https://www.atari.com/es/

[7] Retro Máquinas **[Online] [Accessed: 09-June-2015]**
Available:http://www.retromaquinitas.com/index.php/consolas/tercera-generacion/famicom/

[8] Super Mario Bros **[Online] [Accessed: 09-June-2015]**
Available: https://es.wikipedia.org/wiki/Super_Mario_Bros/

[9] Sega Megadrive **[Online] [Accessed: 09-June-2015]**
Available: https://es.wikipedia.org/wiki/Sega_Mega_Drive/

[10] GameBoy | Empresa | Nintendo **[Online] [Accessed: 09-June-2015]**
Available: https://www.nintendo.es/Empresa/La-historia-de-Nintendo/Game-Boy/Game-Boy-627031.html/

[11] Sony PlayStation **[Online] [Accessed: 09-June-2015]**
Available: https://es.wikipedia.org/wiki/PlayStation/

[12] Nintendo 64 **[Online] [Accessed: 09-June-2015]**
Available: https://es.wikipedia.org/wiki/Nintendo_64/

[13] Genre Game Shooter **[Online] [Accessed: 09-June-2015]**
Available: http://www.mobygames.com/game-group/genre-rail-shooter/offset,125/so,1d/

[14] Star Wars Wiki **[Online] [Accessed: 09-June-2015]**

Available: http://es.starwars.wikia.com/wiki/Caza_estelar_Ala-X_T-65/

[14] Nintendo NES Zapper **[Online] [Accessed: 09-June-2015]**

Available: https://es.wikipedia.org/wiki/Nintendo_Zapper/

[15] Original Version Japanese Otaku City by ZENRIN is licensed under a Creative Commons Attribution 4.0 International License(CC-BY). **[Online]**

Available: http://www.zenrin.co.jp/product/service/3d/asset/

[16] Curva de dificultad de los videojuegos **[Online] [Accessed: 10-June-2015]**

Available: http://es.ign.com/reportaje/92847/feature/la-curva-de-dificultad-en-videojuegos

[17] VRWiki | PS Move **[Online] [Accessed: 12-June-2015]**

Available: https://vrwiki.wikispaces.com/PS+Move

[18] Unity **[Online] [Accessed: 15-June-2015]**

Available: https://unity3d.com/es/

[19] blender.org – Home of the Blender project **[Online] [Accessed: 15-June-2015]**

Available: https://www.blender.org/

[20] Adobe Photoshop **[Online] [Accessed: 17-June-2015]**

Available: http://www.adobe.com/es/products/photoshopfamily.html/

[21] Diseño gráfico | Adobe Illustrator CC **[Online] [Accessed: 17-June-2015]**

Available: http://www.adobe.com/es/products/illustrator.html/

[22] Audacity (spanish) **[Online] [Accessed: 17-June-2015]**

Available: http://audacity.es/

[23] Github · Build software better, together **[Online] [Accessed: 17-June-2015]**

Available: https://github.com/

[24] Survey Monkey **[Online] [Accessed: 17-June-2015]**

Available: https://es.surveymonkey.com/home/

[25] TF3DM – 3D Free Models **[Online] [Accessed: 18-June-2015]**

Available: http://tf3dm.com/

[26] Animation - Unity **[Online] [Accessed: 18-June-2015]**

Available: http://docs.unity3d.com/ScriptReference/Animation.html/

[27] Asset Store | Zombie **[Online] [Accessed: 18-June-2015]**

Available: https://www.assetstore.unity3d.com/en/#!/content/30232/

[28] Asset Store | Elementals **[Online] [Accessed: 18-June-2015]**

Available: https://www.assetstore.unity3d.com/en/#!/content/11158/

[29] Aspect ratio switching, Full Screen Unity | Pierre’s Blog **[Online] [Accessed: 19-June-2015]**

Available: http://pierresemaan.com/aspect-ratio-switching-full-screen-and-unity/

[30] Hermite Spline Controller – Unify Community Wiki **[Online] [Accessed: 19-June-2015]**

Available: http://wiki.unity3d.com/index.php?title=Hermite_Spline_Controller/

[31] ¿Cuánto gano un desarrollador de videojuegos? **[Online] [Accessed: 20-June-2015]**

Available: https://www.niubie.com/2011/01/cuanto-gana-un-desarrollador-de-videojuegos/

[32] Sueldo de un…. **[Online] [Accessed: 20-June-2015]**

Available: http://about-why.com/negocios/recursos-humanos/sueldo-de-un-artista-foley.php/

[33] Kung Fury | Youtube **[Online] [Accessed: 20-June-2015]**

Available: https://www.youtube.com/watch?v=bS5P_LAqiVg/

[34] Shenmue 3 by Ys | KickStarter**[Online] [Accessed: 20-June-2015]**

Available: https://www.kickstarter.com/projects/ysnet/shenmue-3/

[35] MotionInJoy**[Online] [Accessed: 26-June-2015]**

Available: http://motioninjoy.uptodown.com/

[36] BlueSoleil**[Online] [Accessed: 26-June-2015]**

Available: http://www.bluesoleil.com/

[37] Toshiba Bluethoot Stack **[Online] [Accessed: 26-June-2015]**

Available: https://aps2.toshiba-tro.de/bluetooth/?page=download/

Glosario

A

API: conjunto de subrutinas, funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

Asset: componentes de objetos, modelos, imágenes etc, que puedes utilizar para tus proyectos

B

Backup: copia total o parcial de información importante del disco duro u otro medio de almacenamiento.

Batching: proceso donde el motor intenta combinar el renderizado de múltiples objetos en un único draw call para reducir la sobre carga de la CPU.

Bluethoot: especificación industrial para Redes Inalámbricas de Área Personal que posibilita la transmisión de voz y datos entre diferentes dispositivos.

Bug: error o defecto en el software o hardware que hace que un programa funcione incorrectamente.

C

Canvas: área en que todos los elementos de la interfaz de usuario deben estar dentro.

Collider: objeto que se activa cuando algún componente choca con él.

Combo: multiplicador de puntos utilizado en los videojuegos.

D

Dongle: pequeño dispositivo que puede ser conectado y usado por un PC, especialmente para permitir el acceso a redes inalámbricas.

Draw Calls: total de mallas dibujadas tras el proceso de batching.

E

EDSAC: (Electronic Delay Storage Automatic Calculator), primera computadora operacional que podía almacenar programas electrónicos.

F

Framerate: frecuencia de reproducción de las imágenes o frames.

G

GameObject: cada objeto en el juego es un GameObject. Este concepto solo es aplicable para Unity 3D.

H

Headshot: término que hace referencia a un disparo en la cabeza.

I

Input: datos que se introducen en un sistema o un programa informáticos.

Interfaz: medio que permite a una persona comunicarse con una máquina.

IR(infrarrojo): es un tipo de radiación electromagnética y térmica, de mayorlongitud de onda que la luz visible, pero menor que la de las microondas

K

Kickstarter: sitio web de financiación en masa para proyectos creativos.

M

Malla: colección de triángulos y vértices que aproximan una superficie en 3D.

Máquina de estado: modelo de comportamiento de un sistema con entradas y salidas, en donde las salidas dependen no sólo de las señales de entradas actuales sino también de las anteriores

N

Node: punto de intersección, conexión o unión de varios elementos que confluyen en el mismo lugar.

NPC: personajes que aparecen en un juego, por lo general aliados o no, que tú no puedes controlar sus acciones.

O

Osciloscopio: Aparato que sirve para registrar oscilaciones de ondas y las presenta en una pantalla.

OXO: versión electrónica del juego del tres en línea.

P

Plugin: Un complemento es una aplicación que se relaciona con otra para aportarle una función nueva y generalmente muy específica.

Prefab: gameobject con características establecidas por el usuario.

Pre-renderizado: imagen o textura en un juego que fue renderizada a través de un motor gráfico que se usa en el juego, por lo cual el motor gráfico del juego sólo se ocupa de calcular la posición de esa textura y no de todo su contenido.

R

Release: sacar, poner en venta, nueva versión, la más reciente.

Renderizar: proceso de generar una imagen o vídeo mediante el cálculo de iluminación GI partiendo de un modelo en 3D.

Runtime: proceso que se crea al iniciar el programa.

S

Script: programa usualmente simple, que por lo regular se almacena en un archivo de texto plano.

Shooter on-rails: tipo de juego en que el personaje va sobre una guía y interactúa con los elementos de la pantalla.

Spline: curva diferenciable definida en porciones mediante polinomios.

Sprint: proyecto que se ejecuta en bloques temporales cortas y fijas.

Sticks: Palancas analógicas de los mandos de las videoconsolas.

Storyboard: conjunto de ilustraciones mostradas en secuencia con el objetivo de servir de guía para entender una historia.

T

Tester: persona que pone a prueba el sistema, aplicación etc.

Textura: agregación de materiales que se percibe como variaciones o irregularidades en una superficie continúa.

U

UI: Componentes de un sistema de informático empleado para la comunicación del usuario con la computadora.

Upgrade: aumentar de categoría o versión incluyendo nuevas funciones.

Anexo

A1 Organización del proyecto

El proyecto de Zombie Assault está organizado de la siguiente manera (véase Figura A1.1).

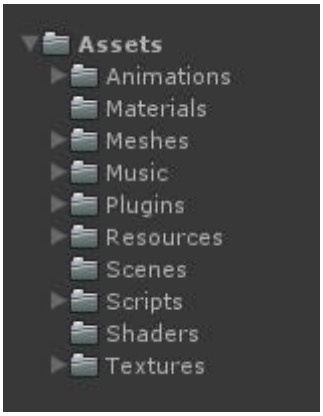


Figura A1.1: La siguiente imagen muestra la organización del proyecto dentro de Unity 3D.

El directorio de carpetas cuelga de la carpeta raíz “Assets” que es la que contiene todos los componentes que se van a utilizar para desarrollar el videojuego. Los directorios que incluye esta carpeta son:

- **Animations:** Esta carpeta contiene los archivos de animaciones de todos los modelos del videojuego.
- **Materials:** Esta carpeta contiene todos los materiales del videojuego que no se instancian en tiempo de ejecución.
- **Meshes:** Esta carpeta contiene todos los modelos utilizados en el videojuego.
- **Music:** Esta carpeta contiene todos los ficheros de audio utilizados en el videojuego.
- **Plugins:** Si queremos cargar algún plugin compilado de otra plataforma es necesario que esta carpeta se llame de este modo ya que sino no será capaz de cargar el componente.
- **Scenes:** Esta carpeta contendrá todas las escenas del juego.
- **Shaders:** Esta carpeta contendrá ficheros que trabajan sobre la GPU para dar efectos sobre los materiales aplicados.
- **Textures:** Esta carpeta contendrá las texturas de los modelos que no se instancien en tiempo de ejecución.
- **Scripts:** Esta carpeta contendrá todos los componentes de programación del juego. Cada uno de estos componentes están separados en subdirectorios dependiendo del comportamiento que desempeñe (véase Figura A1.2).

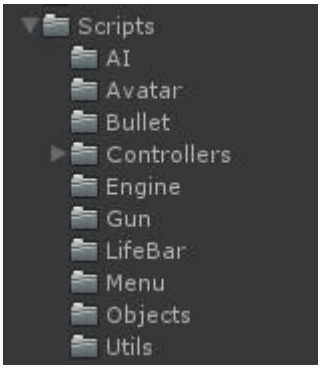


Figura A1.2: La siguiente imagen muestra los subdirectorios de la carpeta Scripts.

Dentro del directorio Script tenemos los siguientes subdirectorios:

- **AI:** Esta carpeta contendrá los Scripts que tienen que ver con la inteligencia artificial de los enemigos.
- **Avatar:** Esta carpeta contiene todos los Scripts del jugador.
- **Bullet:** Esta carpeta contiene el comportamiento de la munición.
- **Controllers:** Esta carpeta contiene los métodos y funciones que permite la comunicación entre los inputs de los periféricos.
- **Engine:** Esta carpeta contiene el motor que controlan la inteligencia de todo el juego.
- **Gun:** Esta carpeta contiene toda la programación de las armas del juego del personaje.
- **LifeBar:** Esta carpeta contiene todo el comportamiento de todo lo que envuelve el HUD de la HealthBar.
- **Menu:** Esta carpeta contiene el comportamiento de las opciones de los menús.
- **Objects:** Esta carpeta contiene el comportamiento de los objetos del videojuego.
- **Utils:** Esta carpeta contiene scripts que realizan comportamientos específicos, ej: multilanguage.

- **Resources:** Si queremos instanciar o cargar elementos en tiempo de ejecución esta carpeta deberá tener este nombre ya que al compilar el videojuego no será capaz de crear estos elementos. Esta carpeta actuará de la misma forma que la de Assets (véase Figura A1.3).

Figura A1.3: La siguiente imagen muestra los subdirectorios de la carpeta Resources.

Dentro del directorio Resources tenemos los siguientes subdirectorios:

- **Elementals:** Esta carpeta contiene partículas de diferentes tipos.
- **Music:** Esta carpeta contiene efectos de audios que se instancian en runtime.

- **Particles:** Esta carpeta contiene partículas creadas por mí.
- **Splines:** Esta carpeta contiene todos los objetos y scripts para crear el rail del juego.
- **Standard Assets:** Esta carpeta contiene los assets de los skyboxes que incorpora Unity como ejemplo.
- **Textures:** Esta carpeta contiene todas las texturas que serán cargadas en runtime.

A2 Manual del juego

Para la correcta ejecución del juego hay que seguir los siguientes pasos:

1. Seleccionar el proyecto con el sistema compatible con tu sistema operativo (véase Figura A2.1).

Figura A2.1: La siguiente imagen muestra los proyectos compatibles con cada SO.

2. Escogemos uno de los proyectos según el periférico que vayamos a utilizar y la arquitectura que sea compatible con nuestro PC (véase Figura A2.2).

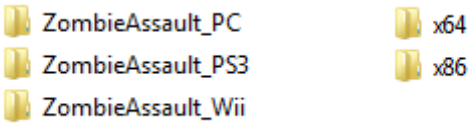


Figura A2.2: La siguiente imagen muestra el proyecto con el periférico que funciona y la arquitectura compatible.

3. Ejecutamos el ejecutable, ZombieAssault.exe para Windows (véase Figura A2.3).

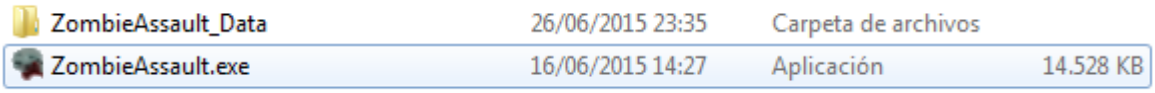


Figura A2.3: La siguiente imagen muestra el ejecutable del juego.

4. Nos aparecerá una ventana donde podemos configurar los ajustes gráficos y de los controles del juego (véase Figura A2.4).

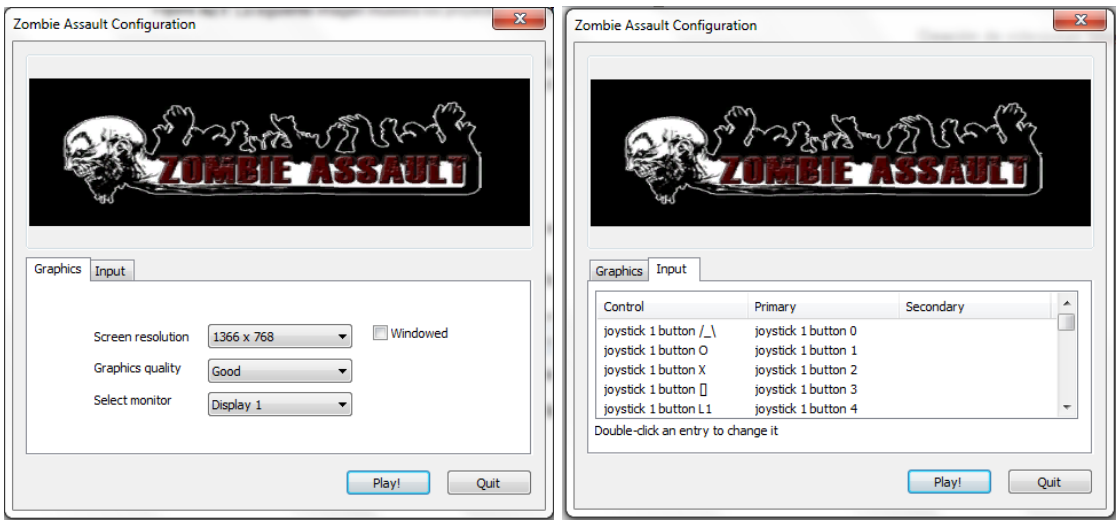
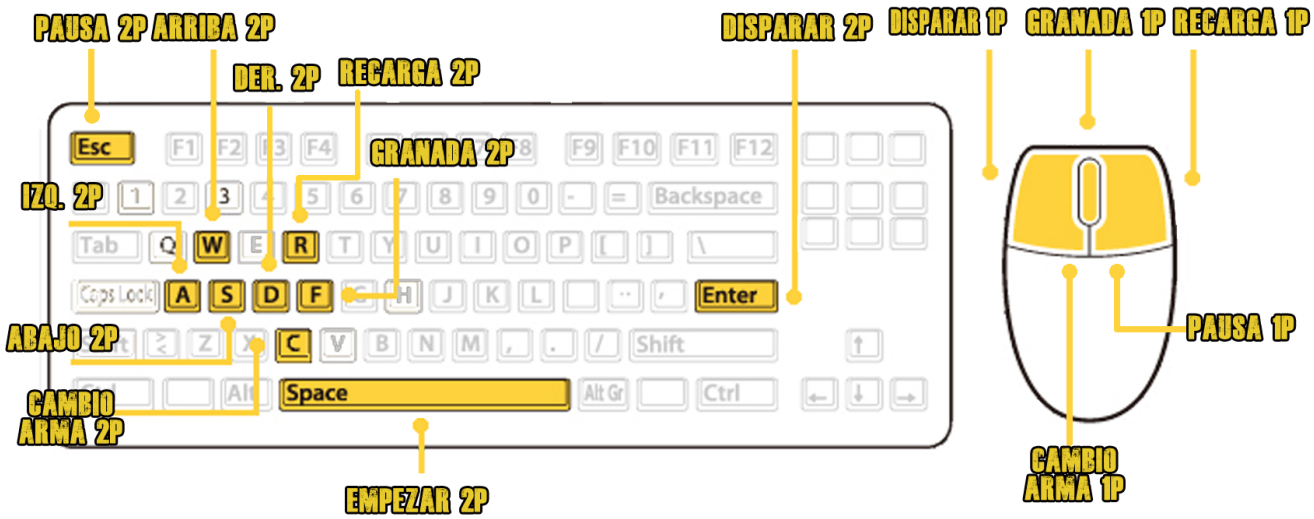


Figura A2.4: La siguiente imagen muestra los ajustes gráficos y de los controles.

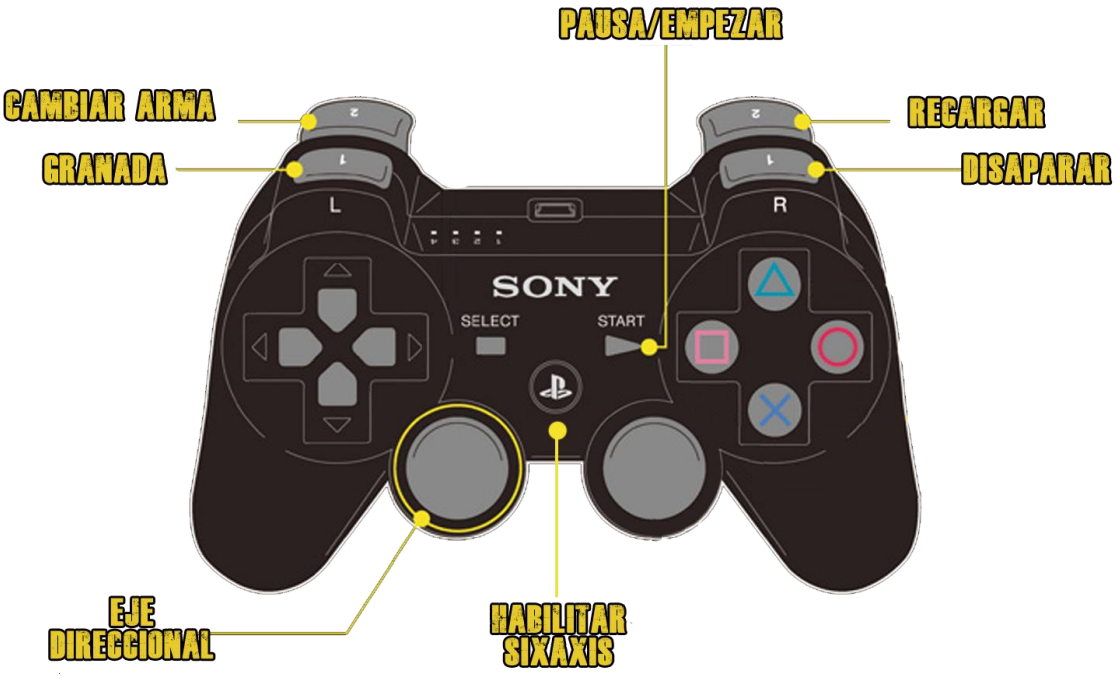
5. Una vez configurado pulsamos Play! y a jugar.

La configuración de los controles según el periférico son los siguientes:

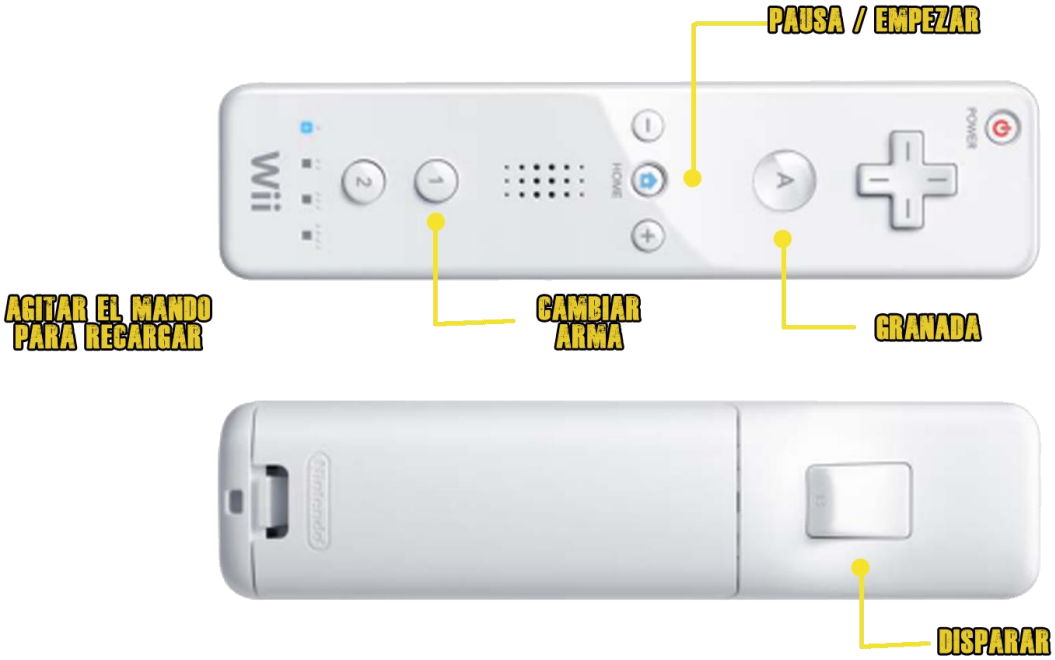
- Controles para PC



- Controles para PS3



- Controles para WiiMote



A3 Encuesta a los usuarios

1. Que te ha parecido la interfaz de usuario?

- Muy buena
- Buena
- Podría estar mejor
- N/C

2. Que te ha parecido el apartado gráfico?

- Muy bueno
- Bueno
- Podría estar mejor
- N/C

3. Que te ha parecido la jugabilidad del juego?

- Muy buena
- Buena
- Podría estar mejor
- N/C

4. Que te ha parecido la durabilidad del juego?

- Muy buena
- Buena
- Podría estar mejor
- N/C

5. Que te han parecido los efectos de sonido?

- Muy buenos
- Buenos
- Podrían estar mejor
- N/C

6. Que opinas del framerate del juego?

- Muy bueno
- Bueno
- Podría estar mejor

- N/C

7. Que te han parecido los niveles de dificultad?

- Muy buena
- Buena
- Se podría mejorar la dificultad
- Se podría reducir la dificultad
- N/C

8. Que te ha parecido que el juego sea multidioma?

- Muy bueno.
- Bueno.
- Habría añadido más idiomas.
- N/C.

9. Que valoración le darías al juego?

- 10
- 9
- 8
- 7
- 6
- 5
- <5

10. Sí has encontrado algún bug, que no sea de la lista de abajo, podrías escribir cuál ha sido y como te ha sucedido?

Bugs conocidos:

o) Problema GameOver con 2 players, si uno muere se acaba el juego.
o) Problema con el Tambor del revolver de las balas.
o) Problema al morir y continuar el cargador de balas puede salir mal posicionado.
o) Problema al recargar y cambiar de arma a la vez.
o) No muestra bien puntuaciones grandes.

A4 Conexión periférico PS3

Para establecer la conexión del controlador de PS3 hay que seguir los siguientes pasos:

1. Instalar la aplicación MotioninJoy [35] para detectar el controlador.
2. Una vez instalado, ejecutamos la aplicación (véase Imagen A4.1).

Figura A4.1: La siguiente imagen muestra el menú de la aplicación.

3. Conectamos el controlador PS3 al PC mediante el cable USB de carga del mando. Nos detectará el mando en la combo list de Control(es) conectados (véase Imagen A4.2).

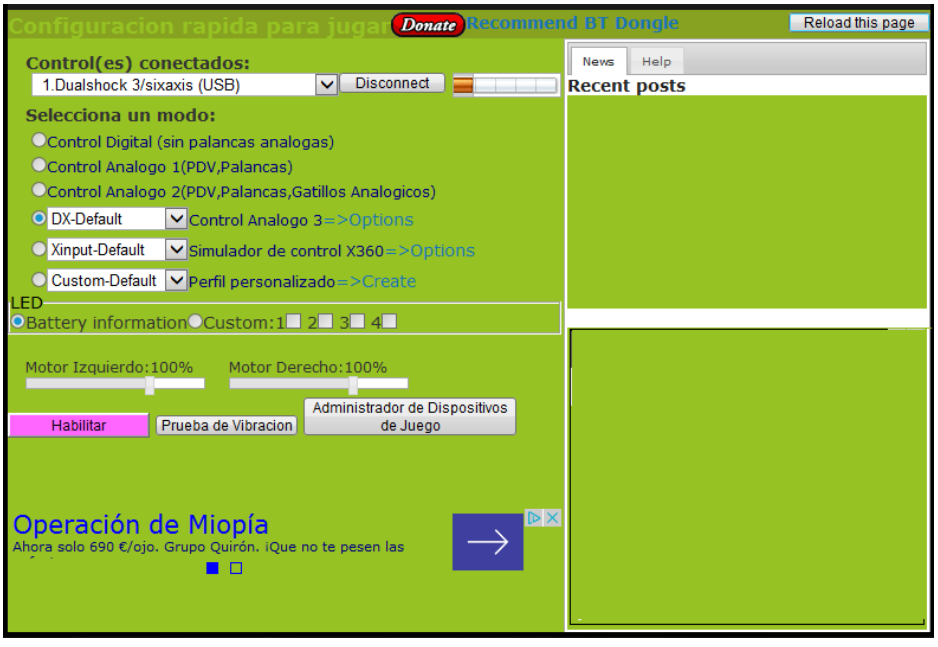


Figura A4.2: La siguiente imagen muestra los mandos conectados al PC.

4. Presionamos sobre la pestaña del Administrador de control y pulsamos sobre el check del Port que nos aparezca en la lista. Finalmente pulsamos sobre el botón Load driver para cargar los drivers del controlador.

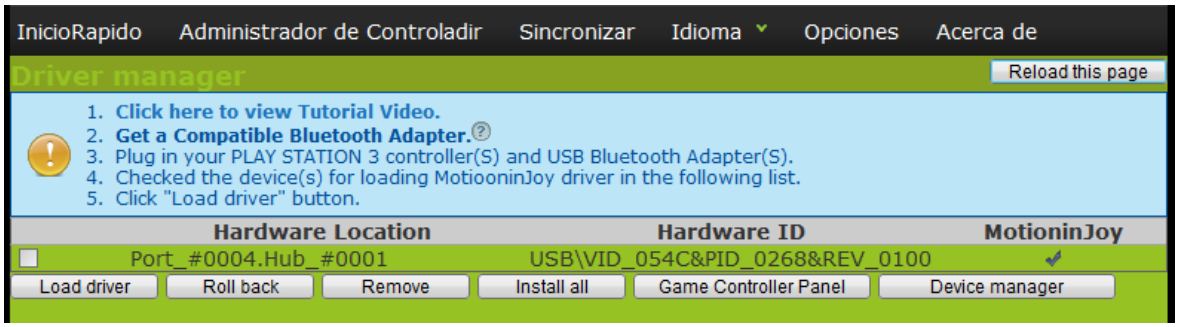


Figura A4.3: La siguiente imagen muestra como cargar los drivers del controlador.

5. Volvemos a Inicio rápido y presionamos sobre Habilitar, si nuestro mando es del tipo DualShock podemos hacer una prueba de vibración, y pulsamos sobre el botón de Administrador de Dispositivos de Juego para que se habrá la ventana de configuración de los Joysticks del sistema. En esta ventana podemos comprobar los inputs del controlador (véase Figura A4.4).

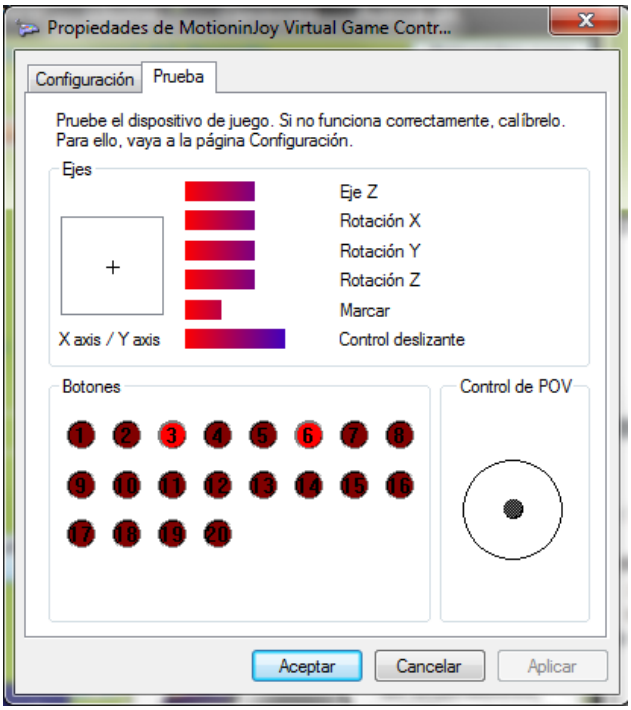


Figura A4.4: La siguiente imagen muestra algunos de los Inputs presionados.

A5 Conexión periférico WiiMote

Para establecer la conexión del periférico WiiMote hay que seguir los siguientes pasos:

1. Bajarse o tener instalado algún tipo de software que sea capaz de encontrar dispositivos que tengan Bluethoot como por ejemplo: BlueSoleil [36] o TOSHIBA Bluethoot Stack [37] (véase Figura A5.1).

Figura A5.1: La siguiente imagen muestra la ventana de Nuevas Conexiones de Toshiba Bluethoot.

2. Retiramos la tapa trasera de nuestro WiiMote y dejamos pulsado el botón rojo que pone Sync (véase Figura A5.2).

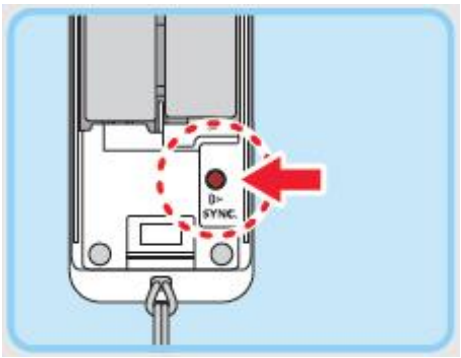


Figura A5.1: La siguiente imagen muestra donde se encuentra el botón de Sync.

3. Mientras tenemos presionado Sync desde el menú del Bluethoot le damos a “*Nueva conexión*” y esperamos a que nos detecte el periférico (véase Figura A5.3).

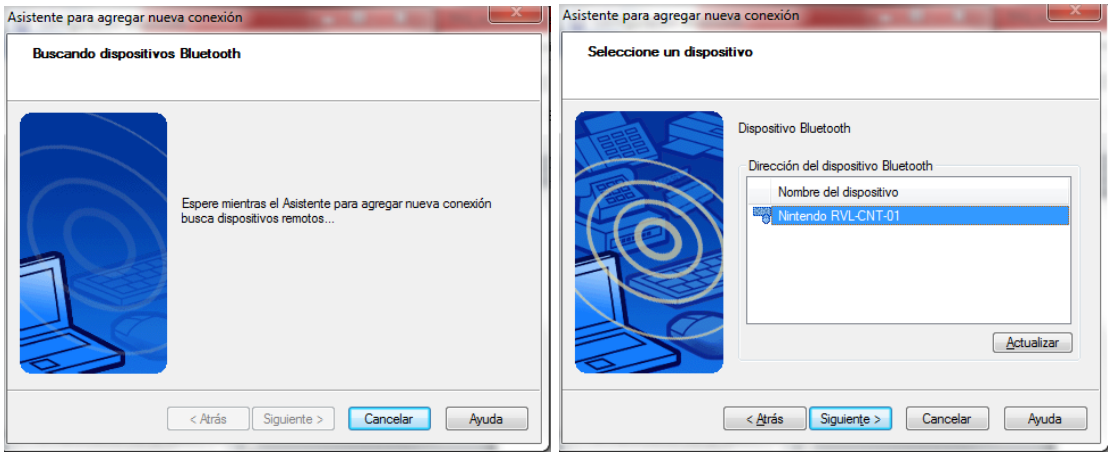


Figura A5.3: La siguiente imagen muestra como el Bluethoot está buscando nuevos dispositivos. Cuando encuentre alguno lo mostrará en la lista.

4. Una vez nos ha detectado el periférico se añadirá al menú principal y lo único que nos queda por hacer es darle doble-click encima del periférico que queremos que se conecte mientras tenemos pulsado el botón de Sync (véase Figura A5.4).

Figura A5.4: La siguiente imagen muestra como el segundo periférico está conectado al PC.

5. Como último paso lo que se podría hacer es testear que todo vaya correctamente desde la aplicación WiinRemote que nos muestra los movimientos e Inputs que se está haciendo con el WiiMote (véase Figura A5.5)

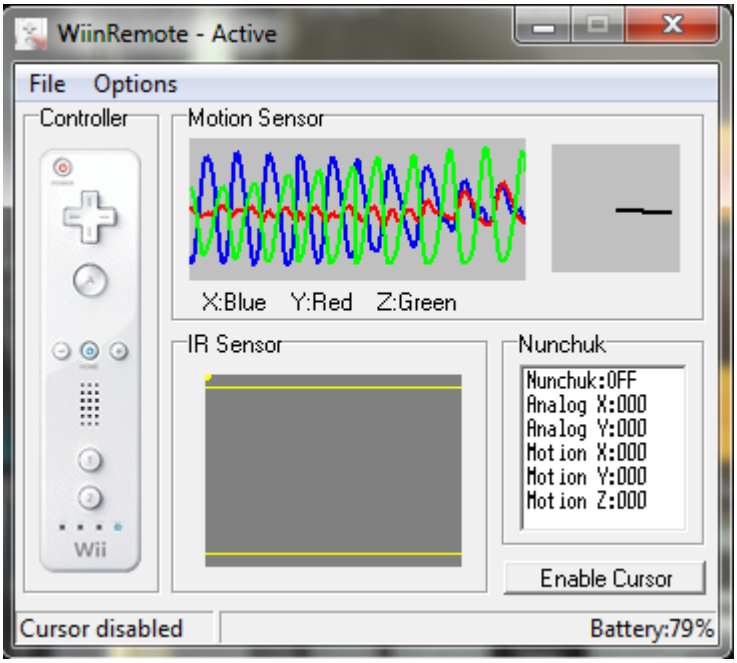


Figura A5.4: La siguiente imagen muestra los movimientos que se realizan con el periférico.

WiinRemote es una aplicación que sirve para testear todas las funcionalidades del WiiMote desde el PC. Como se observa en la Figura A5.4 cada uno de los apartados que tiene sirve para ver cómo está funcionando el WiiMote conectado.

En el apartado de “*Controller”* tenemos la imagen del WiiMote, a medida que se pulse un botón del periférico la imagen iluminará el botón que este presionado en ese momento.

WiinRemote también muestra los cambios que hay en el acelerómetro del periférico y se puede ver reflejado en la parte de “*Motion Sensor”*. También nos dice hacía donde estamos apuntando con el mando y a qué velocidad en la ventana de la derecha de la gráfica que muestra los cambios en el acelerómetro.

Si tuviésemos la barra sensora conectada en el apartado de “*IR Sensor”* nos mostraría un punto entre las franjas amarillas con la posición del periférico.

A6 Creación de barra sensora

Para la creación de la barra sensora se necesitan los siguientes materiales:

1. Placa de topos.
2. De 10-15 LEDs IR, de tipo emisor y a poder ser que el cono de emisión sea de 36 grados.
3. Cable USB de tipo B.
4. Conector USB tipo B.
5. 2 resistencias de 22 ohm cada una.
6. Cautín.
7. Estaño.
8. Cable.

Una vez ya tenemos todos los materiales el circuito que se ha diseñado es el siguiente (véase Figura A6.1)

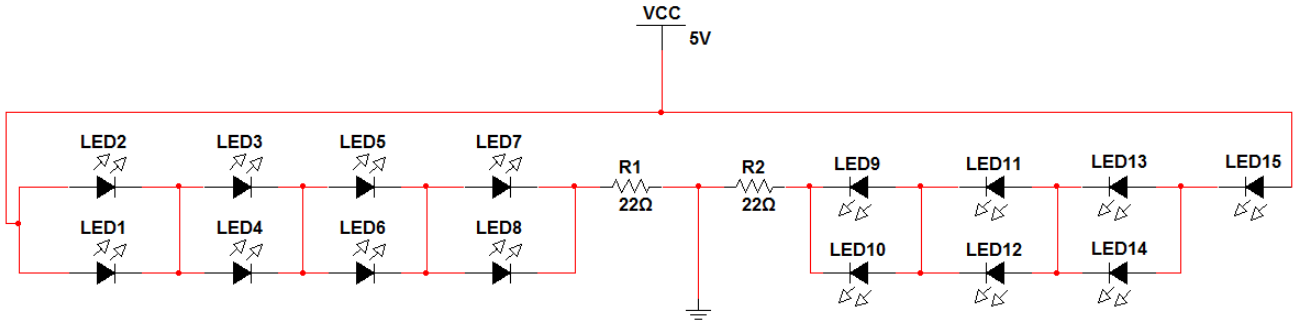


Figura A6.1: La siguiente imagen muestra el circuito que se ha creado para la placa.

Como vemos en la Figura A6.1 el circuito estará alimentado por una fuente de entrada de 5V para alimentar los LEDs que hay en el circuito. La cantidad de resistencias a añadir depende de cómo se diseñe el circuito, en este caso con 2 ya sería suficiente para tener controlados posibles cortocircuitos y así evitar que se nos queme la placa.

La fuente de alimentación de entrada de nuestra placa será el PC. Esto es posible debido a que se ha añadido a la placa un conector USB tipo B que estará conectado al PC mediante un USB.

El resultado final sería el siguiente (véase Figura A6.2)

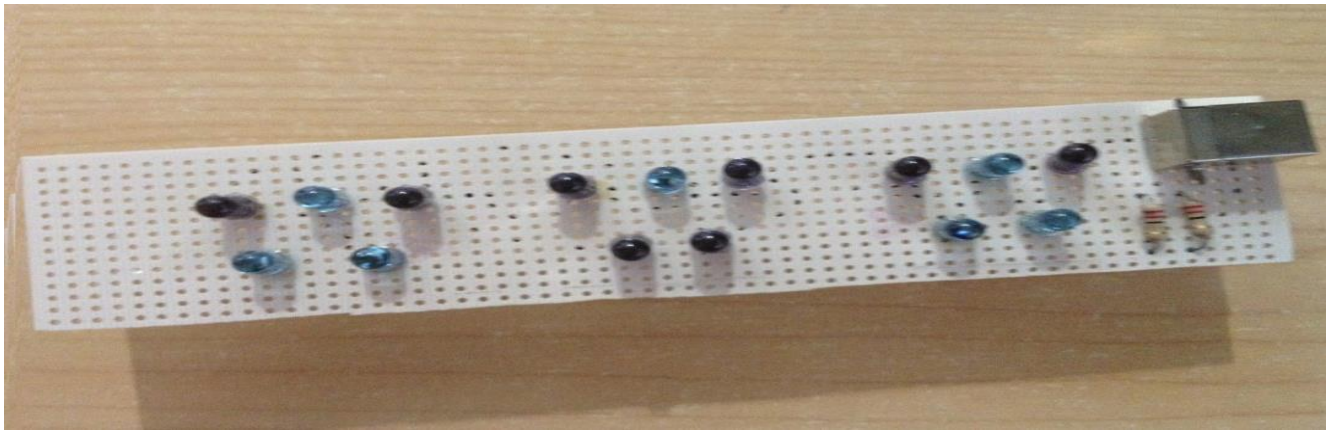


Figura A6.2: La siguiente imagen muestra el resultado final de la barra sensora creada.