ugr | Universidad de Granada

TRABAJO FIN DE GRADO

INGENIERÍA INFORMÁTICA

Videojuego de carreras de coches

Videojuego de carreras, enfocado en Inteligencia Artificial

Autor

Julio Antonio Fresneda García

Tutor

José Manuel Benítez Sánchez

Escuela Ténica Superior de Ingenierías Informática y de Telecomunicación

—

Granada, junio de 2019

Palabras clave: Inteligencia Artificial, IA, videojuego, coches, carrera, red neuronal, algoritmos genéticos

Resumen

El proyecto presentado es un videojuego de carreras de coches, donde la mayor parte del desarrollo ha sido relativo a la inteligencia artificial.

El juego debe proporcionar suficiente variedad de contextos (circuitos, trazados, tipos de vehículos, …). Uno de los aspectos prioritarios en el diseño del videojuego será la incorporación de modelos adaptativos de Inteligencia Artificial para hacer atractivo para jugadores con niveles distintos de habilidad del usuario.

Keywords: Artificial Intelligence, AI, videogame, car, race, neural network, genetic algorithm

Abstract

The project presented is a videogame of racing cars, where most of the development has been relative to artificial intelligence.

The game must provide a variety of contexts (circuits, routes, types of vehicles, ...). One of the priority aspects in the design of the game will be the incorporation of adaptive models of Artificial Intelligence to make it attractive for players with different levels of user ability.

Yo, **Julio Antonio Fresneda García**, alumno de la titulación de grado de Ingeniería Informática de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, con DNI 49215154F, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

D. **José Manuel Benítez Sánchez**, Profesor del grado de del Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada.

Informa:

Que el presente trabajo, titulado *Videojuego de carreras de coches: Videojuego de carreras enfocado en la Inteligencia Artificial*, ha sido realizado bajo su supervisión por **Julio Antonio Fresneda García**, y autorizo la defensa de dicho trabajo ante el tribunal que corresponda.

Agradecimientos

A mi familia por el apoyo y paciencia.

A mis amigos, por interesarse y darme ánimos en la recta final.

A mi tutor, por todos los consejos y ayuda recibida.

1. Introducción

1. En qué consiste este Trabajo de Fin de Grado

Este trabajo es un videojuego de carreras de coches con un enfoque arcade. Puesto que soy de la rama de Computación y Sistemas Inteligentes, la parte más destacable de este proyecto es la inteligencia artificial.

2. Motivación

Hacer un videojuego no es una tarea sencilla. Durante el desarrollo de un videojuego hay que abarcar y tocar muchos campos, algunos totalmente distintos (aparentemente) del mundo de la informática. Programación, modelado, interfaces, diseño del videojuego, sonido, son sólo algunas de las piezas que se necesitan en el desarrollo de un videojuego.

Para tener la motivación necesaria para desarrollar un videojuego, no basta con que te guste la programación. Tampoco con que te guste modelar piezas en tres dimensiones. Para desarrollar un videojuego, necesitas tener motivación e interés en todas y cada una de las partes que componen dicho videojuego, porque si dejas alguna sin trabajar, el videojuego se sentirá cojo, incompleto.

Por supuesto con un equipo de personas con distintos perfiles se puede dividir la tarea, donde nadie tiene que ser experto en todo. Pero en este caso, el desarrollo de un TFG, debo ser yo quien abarque cada una de los campos (o al menos intentarlo, con más o menos éxito).

En mi caso, tengo motivaciones de distinto origen (pero todas útiles) para cada una de las partes de un videojuego. ¿Y qué motivación me lleva a entrar en este mundo? Es un clásico el típico chaval que entra en la carrera de ingeniería informática porque le apasionan los videojuegos, los ha jugado desde pequeño, y quiere pasar de jugarlos a hacerlos. Quiere formar parte de este mundo, y trabajar en la empresa que desarrolla su videojuego favorito. No es mi caso.

En mi caso, siempre me ha interesado el desarrollo de videojuegos, pero por otras razones. Hay dos principales motivaciones que han desembocado en un interés por este tema, y curiosamente son las mismas que me empujaron a entrar a esta carrera.

La primera motivación es mi interés por las ciencias exactas. En esto incluyo desde las matemáticas que se usan en el desarrollo del videojuego, hasta la programación. La programación me gusta por la misma razón que las matemáticas, la algorítmica o el ajedrez, en el fondo consisten en resolver un puzle donde los movimientos o son precisos, sin ambigüedad y tienen sentido, o el puzle no se resuelve.

Mi segunda motivación es totalmente opuesta, y es mi interés por las tareas que se pueden resolver de forma creativa. Soy malísimo dibujando, a sí que tengo que buscar otras aficiones que me permitan ser creativo, y la informática y el desarrollo de videojuegos son dos de ellas. Pocas ramas (si no ninguna) de la informática es más creativa que la de desarrollar videojuegos, si es que puede considerarse a esto rama de la informática.

Tanto los videojuegos, como la informática en sí, son la intersección perfecta entre estas motivaciones.

El diseño de un nivel, el aspecto de los personajes, la historia de trasfondo, cómo se mueven los elementos, y todos los casi infinitos aspectos de un videojuego constan de dos fases: Imaginárselos y diseñarlos, los cuales dependen de tu creatividad, y desarrollarlos, que dependen de la programación y el resto de herramientas.

Para imaginarlos y diseñarlos no hay algoritmos ni técnicas exactas, y eso me atrae. Para desarrollarlos sí que hay algoritmos y técnicas exactas, y eso me atrae.

3. Objetivos

Si no fuese consciente de mis limitaciones, el objetivo sería claro: Desarrollar un videojuego completo listo para publicar. Sin embargo, dado que el desarrollo de un videojuego de calibre puede llevar años, tendré que rebajar mis expectativas.

En el caso de este trabajo de fin de grado, el objetivo no es crear un videojuego completo, si no la base de la cual podría salir ese videojuego soñado. Un punto de partida donde se toquen cada uno de los ámbitos antes mencionados que tiene un videojuego, en la medida de lo posible. Que sea jugable evidentemente, que sea entretenido, y que refleje al menos las ganas e ilusión que tengo por este mundo.

Ya que el tiempo y recursos son limitados, y no puedo desarrollar en profundidad cada una de las partes que componen un videojuego, voy a centrarme en la que mejor se me da (o eso creo), y la que supongo que a un informático más le interesa: La Inteligencia Artificial.

Por tanto, el objetivo de este trabajo de fin de grado será crear un prototipo de videojuego donde la parte que más brille, y la que más en profundidad se tocará, sea la Inteligencia Artificial del videojuego.

4. Estructura de esta memoria

En los siguientes capítulos de la memoria, se explicará todo lo relacionado con el trabajo de fin de grado.

En el capítulo 2: Preliminares, se explicará el punto de partida en el que estamos cuando queremos desarrollar un videojuego. Desde el contexto de los videojuegos a día de hoy, hasta la historia de la I.A. enfocada en este campo, pasando por las herramientas que se han usado en este proyecto.

En el capítulo 3: Planificación y Metodología, se explicará la planificación que he seguido a lo largo de este proyecto, y en qué medida se ha cumplido esto, así como la metodología usada.

En el capítulo 4: Análisis de requisitos, se verán qué requisitos debe cumplir nuestro videojuego, de forma que el producto final cumpla con unas expectativas deseadas.

El capítulo 5: Diseño, será junto con el siguiente, el capítulo más extenso con diferencia, pues se explicará a fondo en qué consiste el videojuego, sus modos de juego, su mecánica, y por supuesto, cómo se ha hecho todo esto. En este capítulo debería ir la inteligencia artificial, pero por su notable importancia en este trabajo, se explicará en el siguiente capítulo.

El capítulo 6: I.A. en el videojuego, contendrá todo lo relacionado con la inteligencia artificial usada en el videojuego, tanto su funcionamiento como los métodos usados, así como los problemas a los que me he enfrentado, entre otras cosas.

En el capítulo 7: Conclusión, se plasmarán mis pensamientos finales sobre el proyecto, una vez acabado.

La memoria acabará con un capítulo de bibliografía, donde se citarán las fuentes de todo el material usado, tanto intelectual como imágenes, texturas y sonidos, entre otras cosas.

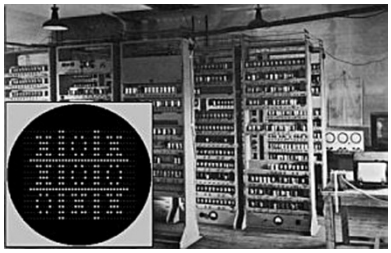
2. Preliminares

Antes de comenzar a explicar el trabajo realizado, es necesario poner un poco de contexto. Porque a quien esté leyendo esta memoria y no esté relacionado con el mundo de los videojuegos, le vendrá bien saber desde qué punto parte este trabajo.

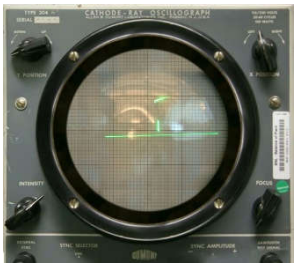
1. Historia de los videojuegos

1. Inicios

Los videojuegos, al ser una temática tan extensa y profunda, tiene tantas definiciones que es difícil decidir cual fue el primer videojuego.



Se podría considerar que el primer videojuego fue el “OXO”, o “Nought and crosses”, desarrollado por Alexander S.Douglas en 1952. Este videojuego no era otra cosa que un tres en raya, donde el jugador competía contra la máquina. Este videojuego se ejecutaba sobre la EDSAC, la cual fue el primer calculador electrónico en el mundo en contar con órdenes internas, aunque no la primera computadora con programas internos (ese honor le corresponde a la SSEM).



Poco más adelante, en 1958, William Higgibotham creó Tennis for Two, el cual era un simulador de ping pong. Este videojuego se creó gracias a un programa de cálculo de trayectorias y un osciloscopio, y su propósito fue el entretenimiento de los visitantes de la exposición Brookhaven National Laboratory.

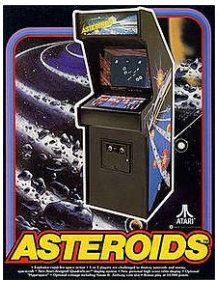
A diferencia que OXO, en este videojuego no intervenía la máquina, si no que jugaban dos jugadores humanos.

El primer videojuego en tener relativo éxito, fue Spacewar. Steve Russell, estudiante del instituto de Tecnología de Massachussets, dedicó seis meses en crear este videojuego, usando gráficos vectoriales. El videojuego consistía en dos naves espaciales que luchaban entre ellas, donde dos jugadores controlaban su dirección y velocidad. El videojuego se ejecutaba sobre un PDP-1, y tuvo relativo éxito en el ambiente universitario.

Pero el comienzo de los videojuegos domésticos no comenzó hasta que Ted Dabney empezó a desarrollar un proyecto de videojuego llamado Fox and Hounds, dando inicio al videojuego doméstico. Este proyecto evolucionó hasta convertirse en la Magnavox Odyssey, el primer sistema doméstico de videojuegos, el cual se conectava a la televisión y permitía jugar a varios juegos pregrabados. Este producto salió a la venta en 1972.

2. La eclosión de los videojuegos, década de los 70s

Tanto Space War como Tennis for Two tuvieron sus versiones comerciales. Computer Space fue una versión de Space War desarrollada por Nolan Bushnell en 1971, y Pong, la máquina recreativa basada en Tennis For Two, comenzó la ascensión de los videojuegos. El sistema fue diseñado por Al Alcom, en la entonces poco conocida Atari.



Pong se presentó en 1972 y fue la base de la industria de los videojuegos. Durante los años siguientes, con los avances de la época en tecnología (como microprocesadores y chips de memoria) se implantaron numerosos avances técnicos en los videojuegos, dando lugar a la aparición de juegos como Space Invaders o Asteroids en los salones recreativos.

3. La década de los 8 bits: Los 80s

La popularidad de los salones de máquinas recreativas y las primeras videoconsolas de los 70 llevaron a un fuerte crecimiento en el sector de los videojuegos. En cuanto a videoconsolas, aparecieron sistemas como Oddyssey 2 (Phillips), Intellivision (Mattel), Colecovision (Coleco), Atari 5200, Commodore 64, Turbografx (NEC). En cuanto a máquinas recreativas, triunfaron juegos como el famoso Pacman (Namco), Battle Zone (Atari), Pole Position (Namco), Tron (Midway) o Zaxxon (Sega).

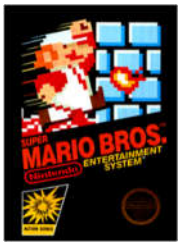
Mientras tanto, en Japón, iba a nacer la que ha sido una de las mayores referencias en el mundo de los videojuegos. La empresa de naipes, Nintendo, dio un giro hacia los videojuegos lanzando la Famicon, o Nintendo Entertainment System (NES), en 1983.

Los norteamericanos adoptaron la NES como su principal sistema de videojuegos, y a lo largo de la década fueron apareciendo nuevos sistemas domésticos, como la Master System (SEGA), el Amiga (Commodore) y el 7800 (Atari), con el ahora clásico Tetris.

A finales de los 80, comenzaron a aparecer consolas de 16 bits como la Mega Drive de Sega, y mientras tanto los microordenadores fueron lentamente sustituidos por las computadoras personales basadas en arquitecturas de IBM.

En 1985 apareció un punto de inflexión en el desarrollo de videojuegos: Super Mario Bros. Mientras que en la mayoría de juegos anteriores la mecánica era repetir en bucle las mismas

pantallas hasta obtener la máxima puntuación posible, el videojuego desarrollado por Nintendo supuso un estallido de creatividad. Por primera vez había un objetivo y una meta en un videojuego. Fue un cambio de estilo, y en años posteriores otras compañías seguirían esta corriente.



En el campo de las recreativas, Japón pasó a ser la mayor productara, y aparecieron videojuegos como Defender, Rally-X, Dig Dug, Bubble Bobble, Gauntlet, Out Run o Shinobi.

Por primera vez aparecieron los videojuegos portátiles, pero su evolución definitiva no llegaría hasta que en 1989 se lanzó la Game Boy, de Nintendo.

4. La revolución de las 3 dimensiones: Década de los 90s

A principios de los años 90 las videoconsolas dieron un importante salto técnico gracias a la competición de la llamada "generación de 16 bits" compuesta por la Mega Drive, la Super Nintendo Entertainmet de Nintendo, la PC Engine de NEC, conocida como Turbografx en occidente y la CPS Changer de (Capcom).

Esta generación explotó la cantidad de jugadores, y gracias a los avances técnicos de la época, como la aparición del CD-ROM, hubo una importante evolución dentro de los diferentes géneros de videojuegos.

Diversas compañías comenzarían a desarrollar videojuegos con entornos tridimensionales, principalmente en el campo de los PC, apareciendo obras como Doom. Las consolas de 16 bits empezaron a considerarse antiguas, y su último logro se produciría gracias a la SNES mediante la tecnología 3D de pre-renderizados de SGI, con videojuegos como Donkey Kong Country y Killer Instinct. La competencia de Nintendo, la Mega Drive lanzó el primer videojuego poligonal en consola, el Virtual Racing, que tuvo gran éxito y marcó un antes y un después en los juegos 3D de consola.

Los videojuegos 3D, dado su gran éxito, se hicieron un importante hueco en el mercado, principalmente gracias a la generación de consolas de 32 bits: Sony PlayStation y Saga Saturn, y la generación de consolas de 64 bits: Nintendo 64 y Atari Jaguar.

La famosísima PlayStation de Sony, originalmente fue un proyecto en colaboración con Nintendo, denominado SNES PlayStation, el cual consistía en un periférico para SNES con lector de CD. Al final Nintendo rechazó la propuesta, puesto que Sega desarrolló un proyecto parecido con un éxito muy pobre, y Sony se vio obligado a lanzar la PlayStation de forma independiente, con un resultado que nadie se esperaría.

Conforme las consolas y ordenadores aumentaban en demanda, los arcades comenzaron con un lento pero imparable declive.

En el mundo de los videojuegos portátiles, producto de las nuevas tecnologías más poderosas, comenzaron su verdadero auge, uniéndose a la Game Boy máquinas como la Game Gear (Sega), Linx (Atari) o la Neo Geo Pocket (SNK), aunque ninguna pudo hacerle frente a la popularidad de la Game Boy, siendo esta y sus descendientes (Game Boy Pocket, Game Boy Color, Game Boy Advance, Game Boy Advance SP) las dominadoras del mercado.

El final de milenio se acercaba, y la consola más popular era la PlayStation con videojuegos como Final Fantasy VII (Square), Resident Evil (Capcom), Winning Eleven 4 (Konami), Gran Turismo (Polyphony Digital) y Metal Gear Solid (konami).

En PC eran muy populares los FPS (juegos de acción en primera persona) como Quake (id Softare), Unreal (Epic Megagames) o Half-Life (Valve), y los RTS (juegos de estrategia en tiempo real) como Command & Conquer (Westwood) o Starcraft (Blizzard). Además, conexiones entre ordenadores mediante internet facilitaron el juego multijugador, convirtiéndolo en la opción predilecta de muchos jugadores, y fueron las responsables del nacimiento de los MMORPG (juegos de rol multijugador online) como Ultima Online (Origin). Finalmente en 1998 apareció en Japón la Dreamcast (Sega) y daría comienzo a la "generación de los 128 bits".

5. Época actual: Desde el 2000 hasta ahora

En el año 2000, Sony lanzó la que sería la consola más vendida de la historia, con más de 155 millones de copias: La clásica PlayStation 2.

Microsoft también entró en escena sacando la Xbox, en 2001. Nintendo lanzó la Gamecube, sucesora de Nintendo 64, y la Game Boy Advance. Ante tal competencia, Sega anunció que ya no produciría hardware, convirtiéndose sólo en desarrolladora de software en 2002.

PlayStation 3, 4, Xbox 360, One, Nintendo DS, 3DS… la cantidad de videoconsolas en el mercado desde los 2000 hasta el día de hoy es inmensa, al igual que la cantidad de videojuegos. El salto en calidad estas dos últimas décadas ha sido estratosférico. Sin embargo, la máquina que más calidad soporta, la más utilizada y la que mejores especificaciones tiene a día de hoy, no es una videoconsola: El ordenador personal.

El ordenador ha llegado a un punto el cual no sólo permite jugar los videojuegos más exigentes desarrollados, si no también crearlos. Gracias al ordenador personal, la comunidad de jugadores es más grande y unida que en toda la historia.

2. I.A. en los videojuegos

Ya que en este trabajo de fin de grado se da gran importancia a la parte de la inteligencia artificial, no está de más recordar el camino de la I.A. en los videojuegos, desde sus humildes inicios hasta el día de hoy.

1. Historia del diseño de la I.A. en los videojuegos

Los videojuegos nacieron sin IA. Los primeros videojuegos, OXO, Tennis for Two, Spacewar, etc no tenían ningún tipo de inteligencia artificial. Y es que, durante los primeros pasos de los videojuegos, la IA no es una característica necesaria. Esto se debe a que esos juegos eran relativamente simples y, durante la mayor parte del tiempo, se jugaba persona contra persona, no contra la máquina.

En 1970, Atari lanzó su primer videojuego, "Computer Space". Y no fue hasta entonces que los diseñadores de juegos comenzaron a realizar su primer intento de incorporar IA en sus juegos. En esta época, las inteligencias artificiales se diseñaron principalmente para juegos arcade con el fin de garantizar que las personas siguieran jugando el máximo de tiempo posible en las recreativas. Pong, Space Invaders y Donkey Kong estuvieron entre los primeros videojuegos también. Estos juegos se ejecutaban bajo reglas muy simples y acciones guionizadas. Los agentes no tenían la capacidad de tomar decisiones. A veces, las decisiones fueron diseñadas

para tomarse al azar, de modo que los comportamientos parecían más impredecibles. Por lo tanto, la llamada Inteligencia fue en realidad precompilada en el juego y no podía actuar en tiempo de ejecución. Por tanto, la primera IA apareció en forma de patrones almacenados. Un ejemplo de una IA tan rígida es el diseño de alienígenas en "Space Invaders". En este juego, el jugador debe disparar a los alienígenas antes de que alcancen la parte inferior de la pantalla. La forma en que se mueven estos alienígenas estaba precompilada en el juego. Tenían un patrón almacenado.

Se crearon muchos más videojuegos basados en este tipo de IA, pero la evolución de la IA acababa de comenzar. La aparición del oponente de la computadora en "Pong" hizo que la gente creyera que la computadora estaba pensando. También fue considerado como la primera inteligencia artificial real en los juegos. La forma en que se desarrolla el juego Pong hace que sea imposible precompilar los comportamientos de objetos no humanos. Las raquetas deben tomar decisiones basadas en las acciones de los jugadores humanos. Las decisiones pueden no ser tan difíciles de tomar: es un cálculo simple de dónde deben ir las raquetas, pero esto hizo que las personas experimenten la misma sensación de jugar contra un jugador humano real.

La influencia que tuvo el diseño de la IA en PacMan es tan significativa como la influencia del videojuego en sí. Este clásico videojuego de arcade hace que el jugador crea que los enemigos en el juego lo están persiguiendo, pero no de una manera burda. Los fantasmas están persiguiendo al jugador (o evadiéndolo) de una manera diferente, como si tuvieran una personalidad individual. Esto le da a la gente la ilusión de que realmente están jugando contra 4 o 5 fantasmas individuales en lugar de copias de un mismo enemigo de computadora.

A finales de la década de 1980, la locura de los videojuegos arcade comenzaba a desvanecerse. Con el desarrollo de la industria de la computación, los ordenadores y consolas domésticos lideraron la nueva dirección del desarrollo de videojuegos. Los nuevos juegos diseñados para estos dispositivos se volvieron más complejos debido a la mayor capacidad de los procesadores modernos. A pesar de que se deben aplicar más recursos a los contenidos y al rendimiento gráfico de mayor calidad, la IA todavía tiene su propia necesidad de desarrollar. Durante la década de 1980, surgieron rápidamente más géneros de juegos. El viejo estilo de diseño de IA estaba desactualizado. Los diseñadores tuvieron que tratar seriamente la IA de los juegos, ya que la IA se ha convertido en una característica estándar de los videojuegos.

Entre los nuevos tipos de juegos, los juegos de estrategia estimularon el desarrollo de la inteligencia artificial debido a que, en estos tipos de videojuegos, la IA es clave. La IA es más importante para los juegos de estrategia y contribuye más al contenido del juego que otros tipos de juegos tales como juegos de rompecabezas o juegos de rol. El juego de estrategia en tiempo real (RTS) se introdujo a finales de los 80 como un nuevo género. Una IA altamente competente y entretenida era lo que este género de juego ofrecía a los jugadores. Pero tal IA era desafiante, por lo que tenía requisitos exigentes. Desde entonces, el diseño de la IA en los juegos de estrategia en tiempo real se ha convertido en una tarea propia y desarrollado como un nuevo campo de investigación.

Los juegos de mesa siempre han sido un clásico en cuanto desarrollo de IA. En 1997, como se explicará en el siguiente apartado, DEEP BLUE consiguió superar al por aquel entonces campeón del mundo Gary Kasparov. En 2016, AlphaGo, desarrollado por Google DeepMind, consiguió derrotar al 18 veces campeón del mundo Lee Sedol, por 4 a 1.

Otros tipos de videojuegos también requerían un desarrollo propio de IA. Half-Life de Valve Software ha recibido grandes elogios por su diseño de IA en el campo de disparos en primera persona. SimCity fue el primero en probar el potencial de los enfoques de vida artificial ("A-LIFE"). Los videojuegos de deportes y carreras también necesitaban de una IA propia, con peculiaridades propias del género.

Pero el desarrollo de IA no ha llegado a su límite. Los videojuegos han recorrido un largo camino desde la década de 1950, al igual que las técnicas de inteligencia artificial que los acompañan. Los últimos años han sido testigos de más y más ideas novedosas, y por tanto también los métodos para los videojuegos que la IA ha unido al proceso de desarrollo del juego.

2. IA teórica vs IA en videojuegos

Normalmente, un programador de IA no necesita una cantidad significativa de conocimiento de Inteligencia Artificial en el campo académico para crear IA de videojuegos de alta calidad. La razón radica en dos hechos. En primer lugar, la inteligencia artificial es una gran rama de la informática. No es fácil adaptar los modelos complejos a los videojuegos. En segundo lugar, el objetivo principal de la IA del videojuego hoy en día es entretener a la gente. Hay trucos u otras formas fáciles de resolver los problemas perfectamente. La gente no estará interesada en cómo funciona la IA en un videojuego. No comprarán un videojuego solo por el hecho de que el diseñador programó complicados algoritmos para resolver un problema matemático a la perfección. En realidad, en la mayoría de los casos, lo que los diseñadores usan para la IA del videojuego no coincide con la rama teórica de IA. Pero siempre viene bien entender los conceptos de IA en el campo académico antes de investigar las técnicas de AI en los videojuegos.

Los conceptos de inteligencia artificial existían mucho antes de que este término fuera utilizado por primera vez. Las raíces intelectuales de la IA pueden remontarse a la mitología griega. Aparecieron artefactos inteligentes en la literatura con algunos dispositivos mecánicos reales que realizan cierto grado de inteligencia. En el siglo V a. C., Aristóteles primero inventó la lógica silogística. También dio la primera definición de inteligencia: la capacidad de poner las cosas en categorías. Pero no fue hasta la historia moderna que la investigación sobre Inteligencia Artificial se hizo más madura. En 1950, A.M. Turing publicó "Computing Machinery and Intelligence", que introdujo la prueba de Turing como una forma de operacionalizar una prueba de comportamiento inteligente. En su ejemplo ilustrativo original, un juez humano se involucra en una conversación en lenguaje natural con un humano y una máquina diseñada para generar un rendimiento indistinguible del de un ser humano. Los conceptos que trajo Turing parecían obvios, pero formaban un concepto esencial en la filosofía de la inteligencia artificial. Para los videojuegos, este concepto también se trata como el objetivo del diseño de IA en videojuegos, naturalmente.

En 1956, el término "Inteligencia Artificial" apareció por primera vez. Fue acuñado por un científico informático llamado John McCarthy. En ese momento, no había videojuegos disponibles. Sin embargo, los videojuegos dedicados a la investigación de la IA en los campos teóricos ya habían existido. En 1950, Claude Shannon publicó su trabajo sobre cómo una computadora puede jugar al ajedrez. Es la primera vez que se utiliza la IA para crear un oponente virtual. Desde entonces, el ajedrez ha sido un pilar en la investigación de la IA. Después de alrededor de medio siglo, en 1997, la computadora DEEP BLUE ganó en una

partida de 6 juegos contra el Gran Maestro de ajedrez Gary Kasparov. DEEP BLUE pudo evaluar 200 millones de posiciones por segundo en comparación con 2 por segundo por un jugador humano. Este logro ha demostrado que la investigación de IA tiene sus frutos.

Sin embargo, este hito no repercutió demasiado en la industria del videojuego. Aunque las plataformas de ajedrez online están disponibles para todo el mundo, el significado de la IA del ajedrez en el campo de la investigación es mucho más importante que la forma en que se aplica para entretener a los jugadores. En la mayoría de videojuegos, de hecho, se supone que la IA debe dejar que el jugador gane, pero de una manera entretenida.

Por otro lado, invertir demasiado en la inteligencia artificial del videojuego para elevarla a niveles académicos no es realista. Remontándonos a la época en que surgieron los videojuegos arcade, los desarrolladores han tenido dificultades para poner incluso IA simple en los juegos debido a las limitaciones de RAM. Incluso hoy en día, el porcentaje de ciclos de CPU que quedan para procesar la IA del juego todavía es limitado. Pero la razón por la que la IA del videojuego no tiene que ser tan buena como la AI académica es que la IA en los videojuegos no necesita ser profunda. El diseño de PacMan es evidencia de que el diseño puede no ser tan difícil como sea posible. La forma en que trabaja PacMan es agregar aleatoriedad a la decisión cuando los fantasmas llegan a un cruce, lo que garantiza que los fantasmas no sigan la misma ruta cada vez. Se trata de la variación del mismo algoritmo de búsqueda de ruta, y los programadores no necesitan hacer que el algoritmo sea más complicado para lograr ese efecto. Las IA en muchos videojuegos son exitosas sin ser profundas.

Pero esto no quiere decir que algunas veces no hagan falta herramientas complejas de la IA teórica para resolver problemas en videojuegos. De hecho, se aplican cada vez más técnicas de la IA académica para resolver problemas en videojuegos. El hecho es que la IA de videojuegos ha hecho uso completo de algunas teorías y métodos básicos de la IA real. Los métodos como las Máquinas de Estado Finito (FSM), los Árboles de Decisión y la Lógica Difusa son herramientas simples pero poderosas, y se usan ampliamente para modelar agentes de IA en juegos.

3. Videojuegos independientes: Contexto

El videojuego desarrollado aquí como trabajo de fin de grado es, sin ninguna duda, un videojuego independiente. Por tanto, no viene mal saber por qué terrenos se mueve el desarrollador independiente a la hora de crear un videojuego.

Los videojuegos independientes o “indie” son videojuegos que no han tenido una base económica para su desarrollo proporcionada por una distribuidora de videojuegos. Estos videojuegos generalmente son diseñados por un equipo de no más de 20 personas, por lo que su desarrollo puede llegar a durar varios años (o horas, depende de la complejidad evidentemente).

Estos videojuegos se pueden obtener principalmente en dos plataformas: PC y dispositivos móviles. Esto es porque para un desarrollador independiente es mucho más fácil obtener las herramientas necesarias para desarrollar en PC y móviles que en videoconsolas, además de que estratégicamente tiene más lógica: La mayoría de jugadores están en estas dos plataformas.

Un desarrollador independiente tiene varias plataformas para exponer su producto al público. La más importante es, sin duda, Steam. Steam tiene aproximadamente 8900 videojuegos bajo la categoría “Indie”, con una valoración media entre el 70% y 77% (fuente: SpySteam). La cantidad de copias vendidas bajo esta categoría se encuentra aproximadamente en 885.800.000 unidades. Esta cantidad de unidades nos da una idea de la importancia de los desarrolladores independientes en el mundo de los videojuegos.

Hay algunos ejemplos de videojuegos indie con un éxito mayor que muchos videojuegos AAA. Por ejemplo, Hotline Miami (2012) ha vendido más de 2 millones de copias, a un precio de 10$ por copia. Minecraft (2015) ha vendido la increíble cantidad de 100 millones de copias vendidas.

Estos ejemplos son la prueba de que un pequeño equipo, con la idea y ejecución adecuada pueden llegar a crear obras de éxito sobresaliente. Evidentemente son la excepción, no la regla.

Otra plataforma por todos conocida es Google Play Store, la cual permite publicar cualquier videojuego, indeferentemente del creador.

Uno de los aspectos a considerar en el crecimiento de esta industria “indie” es el del aumento de las herramientas disponibles para la creación de los mismos. Entre ellas, están las engine, o herramientas de desarrollo de videojuegos de uso global. En el caso de este trabajo, la herramienta usada es Unity.

Unity es uno de los principales exponentes dentro de las aplicaciones gratuitas para el desarrollo de videojuegos. Este engine permite a los desarrolladores la capacidad de crear contenidos a través de una plataforma simplificada en la que se puede crear videojuegos sin un conocimiento demasiado profundo. Unity se puede usar de forma gratuita siempre y cuando la empresa desarrolladora no genere más de 100,000$ en ingresos, en caso contrario, habría que pagar una licencia.

Unity es uno de los engines más sencillos y populares dentro del mercado ya que su integración es muy sencilla y varios de sus métodos y librerías son sencillas y accesible. La información para usarse está disponible a lo largo de muchos documentos por parte de los desarrolladores del engine así como información generada por terceros para el mejor uso del engine. Unity permite a los desarrolladores generar videojuegos y contenido a través de su engine mientras éste no se comercialice, todo contenido que sea comercializado deberá ser realizado en sus versiones de paga o generar un porcentaje de ganancias para Unity.

3. Análisis de requisitos

Los requisitos software de un proyecto son una explicación en detalle de lo que debe ser implementado, en función de las necesidades finales del proyecto. Un requisito es algo que identifica una función, restricción u otra propiedad necesaria que debe realizarse o satisfacerse para cumplir las necesidades propuestas por los usuarios de un sistema informático.

El proceso de ingeniería de requisitos, es un conjunto de actividades ordenado que sirven para obtener los requisitos de nuestro proyecto, los cuales servirán de guía a la hora de desarollarlo.

Una definición no demasiado correcta o ambigua de los requisitos software que buscamos en nuestro proyecto puede inducir a problemas, por lo que es conveniente tener claros qué requisitos vamos a cumplir antes de empezar a desarrollar el proyecto.

Una concepción clara de los requisitos nos ayudará a desarrollar el proyecto de forma más fluida y con menos problemas, ya que dispondremos de una guía a seguir.

En este proyecto se van a obtener requisitos siguiendo las directrices de Ian Somerville, el cual divide el proceso de obtención de requisitos en cuatro fases:

1. Estudio de viabilidad
2. Obtención y análisis de requisitos
3. Especificación de requisitos
4. Validación de requisitos

En el estudio de visibilidad, se estudia si las necesidades del cliente (en este caso, el propio desarrollador) se pueden satisfacer usando las tecnologías existentes en el mercado, y en el tiempo disponible. Si se determina que el proyecto es viable, se continuará con el proceso de desarrollo.

En la obtención y análisis de requisitos, se obtienen cada uno de los requisitos. Estos requisitos se pueden obtener de distintas formas, por ejemplo, sabiendo qué buscamos en el videojuego, u observando proyectos similares.

En la especificación de requisitos, toda la información recopilada en los pasos anteriores, será descrita en detalle, donde cada uno de los requisitos serán clasificados dependiendo de unos criterios determinados.

En la validación de requisitos, se verifica que toda la información obtenida de la fase anterior es real, completa y consistente. En este punto es posible que se encuentren errores, por los que debe corregirse.

Estas cuatro fases no tienen por qué seguirse en este orden estricto, y en el caso de este proyecto podemos ser un poco flexibles.

1. Estudio de viabilidad

En este apartado vamos a ver si nuestra idea de proyecto podría ser viable. El proyecto consiste en un videojuego de carreras de coches de carácter arcade. El videojuego debe ser jugable, y se debe centrar en la inteligencia artificial de los coches.

¿Tenemos las tecnologías necesarias para realizar el proyecto?

En cuanto al engine del videojuego, disponemos de Unity. Unity nos permite desarrollar videojuegos relativamente fácil y gratuitamente en este caso. Para el apartado visual disponemos de Photoshop para imágenes y 3ds Max para el modelado. Además, tenemos algunas páginas con recursos de libre acceso. Por tanto, podríamos decir que en principio tenemos todas las tecnologías necesarias.

¿Se puede realizar el proyecto en el tiempo disponible?

En mi caso personal, solo puedo desarrollar el proyecto desde enero hasta mediados de junio, teniendo las mañanas de enero, marzo, abril ocupadas. Por tanto, suponiendo que de enero a abril puedo invertir 4h diarias, y 8h diarias en mayo y junio, dispondríamos de un total de 720h aproximadamente, lo que serían 90 días trabajando 8h al día.

Dado que el objetivo de este proyecto no es un videojuego de nivel profesional, en principio el proyecto se podría realizar en este periodo de tiempo.

Ya que vemos que disponemos de las tecnologías y el tiempo necesario para llevar a cabo el proyecto, podemos concluir que su desarrollo es viable.

2. Obtención y análisis de requisitos

En esta fase, trataremos de plasmar toda la información necesaria para las siguientes fases.

Las principales actividades del proceso de obtención y análisis de requisitos, son:

- Descubrimiento de requisitos
- Clasificación y organización de requisitos
- Ordenación por prioridades y negociación de requisitos
- Documentación de requisitos

El descubrimiento de requisitos es la fase donde se obtiene la información en bruto, y a partir de aquí obtenemos los requisitos.

La clasificación y organización de requisitos consiste en ordenar en grupos lógicos los requisitos obtenidos en la fase anterior.

La ordenación y negociación de requisitos trata de ordenar por prioridad los requisitos, así como resolver incidencias o contradicciones entre ellos.

La documentación de requisitos consiste en escribir formalmente los requisitos obtenidos.

Estas fases son iterativas y se retroalimentan entre ellas continuamente. Las dos últimas fases no las he desarrollado en este caso pues no las he visto necesarias.

Normalmente para obtener esta información se usan entrevistas con el cliente, cuestionarios, se estudia documentación, etc. En este caso en particular, como es un proyecto propio, la información la proporcionaré yo mismo.

Descubrimiento de requisitos

En este apartado se describirá cómo deberá ser el proyecto, y a partir de aquí se obtendrán los requisitos necesarios.

Menús:

Este proyecto debe ser un videojuego de carreras de coches. En el menú de inicio, debe haber al menos dos modos de juego: Carrera Rápida, y Campeonato. También debe de haber una gestión de perfiles, en la cual cada perfil tiene un nickname, un total de puntos ganados y un total de monedas ganadas.

El modo de Carrera Rápida debe consistir en una carrera única. Cuando seleccionamos este modo, primero debemos seleccionar qué circuito correr, cuantas vueltas y con qué nivel de dificultad de IA jugar. Una vez elegido todo esto, pasaremos a elegir marca y modelo de coche. Deben de haber varias marcas de coches, cada una con distintos modelos, donde todos los modelos tengan ciertas características distintas, haciendo algunos coches mejores que otros para determinados circuitos. En este modo todos los modelos y marcas están desbloqueados. Una vez elegimos coche, comienza la carrera. Una vez acabada la carrera, obtenemos una serie de puntos y monedas según la posición final y según el nivel de dificultad, número de vueltas y modelo elegido, y volvemos a la pantalla principal.

El modo Campeonato consiste en una serie de carreras, una detrás de otra. En este modo se elige dificultad antes de empezar el campeonato, dependiendo de esta selección tendremos mejores o peores premios. Una vez elegido procedemos a correr en todos los circuitos, pudiendo elegir el coche que queremos correr para cada circuito. En este modo no todos los coches están desbloqueados, por lo que desbloquear un coche nos costará monedas. Una vez se desbloquea el coche, se guarda en el perfil que estamos usando. Una vez acabamos todas las carreras, se dictamina la posición final del campeonato, se nos entrega el premio, y volvemos al menú principal.

Gestión de perfiles:

Se debe de poder crear, seleccionar y eliminar perfiles de usuario. Cada perfil de usuario tendrá unos puntos totales, unas monedas que se pueden gastar, y un porcentaje de coches desbloqueados.

Jugabilidad:

El coche debe de poder alcanzar altas velocidades, además de disponer de un dispositivo de boost o nitro, el cual es limitado y se recarga con el paso de los segundos. Este nitro debe darle potencia al coche mientras se usa. El ángulo de giro de la dirección del coche debe de ser inversamente proporcional a la velocidad del coche. Esto hace que se necesite frenar antes de tomar algunas curvas a altas velocidades. Cuando un coche se choque y no pueda seguir, debe de poder respawnear cerca de su ultima posición para poder continuar la carrera. Debe de haber diferencias reales entre distintos modelos de coches, de forma que se note a primera vista la diferencia entre uno de los mejores coches y uno de los peores.

Inteligencia Artificial:

Los coches conducidos por IA deben de poder suponer un reto, no debe de ser fácil ganarles. Estos coches deben ser capaces de acelerar, frenar y girar de forma autónoma para completar el circuito en el menor tiempo posible. Además, deben ser capaces de adelantar a otros coches.

Físicas:

Al ser un videojuego de enfoque arcade, las físicas no necesitan ser perfectamente realistas. Se busca que los choques entre coches o con los bordes del circuito no obstaculicen demasiado la continuidad de la carrera (por ejemplo, que no empiecen a dar vueltas de campana). Los coches deben derrapar un poco cuando giran a altas velocidades, pero no demasiado. No es necesario simular daños por golpes en los coches.

Clasificación y organización de requisitos

A partir del apartado anterior, vamos a obtener todos los requisitos posibles, clasificándolos en requisitos funcionales y no funcionales.

Los requisitos funcionales detallan como el sistema debe actuar, y qué es lo que debe hacer. En la especificación de estos requisitos se suelen especificar las entradas y salidas que necesitan.

Los requisitos no funcionales se refieren a los requisitos que especifican las propiedades emergentes del sistema como disponibilidad, tiempo de respuesta, rendimiento, etc.

En este apartado se obtendrán los requisitos, y en el siguiente apartado (Especificación de requisitos) detallaremos cada requisito con sus entradas, salidas, etc.

Requisitos funcionales

Interfaz de menú:

- Crear, borrar o cambiar de perfil de usuario
- Seleccionar modo de juego (Carrera Rápida o Campeonato)
- Opción de salir del juego

Interfaz en Carrera Rápida:

- Seleccionar circuito
- Seleccionar número de vueltas
- Seleccionar dificultad
- Seleccionar marca y modelo de coche

Interfaz en Campeonato:

- Seleccionar dificultad
- Seleccionar marca y modelo de coche
- Desbloquear un coche

Interfaz dentro de la carrera:

- Disponibilidad de menú de pausa
- Calcular puntos
- Ir al menú

Jugabilidad:

- El coche debe poder acelerar, frenar y girar de forma que pueda completar el circuito
- El coche debe poder alcanzar altas velocidades
- El coche debe tener un boost o nitro
- El nitro debe de gastarse mientras se usa, y recargarse mientras no se usa
- El ángulo de giro de la dirección del coche debe ser inversamente proporcional a su velocidad
- Cuando un coche no pueda continuar, debe respawnear cerca de su última posición
- Debe haber diferencias reales entre distintos modelos de coches

Inteligencia Artificial:

- Los coches conducidos por IA deben ser capaces de acelerar, frenar y girar de forma autónoma para completar el circuito en el menor tiempo posible.
- Los coches conducidos por IA deben poder completar el circuito en un tiempo suficientemente bajo para que suponga un reto para el jugador
- Los coches conducidos por IA deben ser capaces de adelantar a otros coches

Físicas:

- Los choques entre coches o con los bordes del circuito no deben entorpecer bruscamente la carrera
- Los coches deben derrapar al girar bruscamente a altas velocidades, pero no demasiado
- No se deben simular daños por golpes

Requisitos no funcionales

- El videojuego debe de consumir pocos recursos, para que un PC de gama media pueda ejecutarlo de forma fluida
- El videojuego debe poder ejecutarse en cualquier momento
- El videojuego debe ser fiable y no tener bugs o crashes espontáneos
- El videojuego debe poder estar abierto a ampliación de contenido o características en un futuro
- El videojuego debe de usar contenido multimedia con derechos que permitan su uso
- La ejecución del videojuego debe no suponer un problema de seguridad para el sistema
- El videojuego debe adaptar su resolución a la del monitor, para que se visualice correctamente

A priori estos serían los requisitos más importantes. En proyectos de videojuegos es muy difícil escribir todos y cada uno de los posibles requisitos, muchos son directamente programados. Sin embargo, se ha intentado que los más importantes estén plasmados antes de la fase de diseño.

3. Especificación de requisitos

Ya tenemos los requisitos, ahora vamos a especificarlos y aumentar el nivel de detalle.

Requisitos funcionales

Para los requisitos funcionales usaremos la siguiente plantilla.

Nombre	
Resumen	
Descripción	
Entradas	
Salidas	
Procesos	
Precondiciones	
Postcondiciones	

Nombre	RF-IM1
Resumen	Crear, borrar o cambiar de perfil de usuario
Descripción	Se debe poder crear o borrar un perfil de usuario, así como cambiar de perfil
Entradas	Click en botón de gestión de usuarios
Salidas	Información de usuario actualizada
Procesos	Al agregar o borrar un usuario, se agrega o borra de la base de datos. Al cambiar de usuario, simplemente se cambia de datos de usuario seleccionado
Precondiciones	Debe estar en la pantalla de menú
Postcondiciones	

Nombre	RF-IM2
Resumen	Seleccionar modo de juego
Descripción	Se debe poder seleccionar el modo de juego deseado: Carrera Rápida o Campeonato
Entradas	Click en el botón
Salidas	Carga de la pantalla correspondiente
Procesos	Cuando se hace click en el botón correspondiente, el sistema lo detecta y procede a cargar la pantalla correspondiente
Precondiciones	Debe estar en la pantalla de menú
Postcondiciones	

Nombre	RF-IM3
Resumen	Opción de salir del juego
Descripción	Se debe poder salir del juego pulsando un botón
Entradas	Click en el botón
Salidas	Cerrar el juego
Procesos	Cuando se hace click en el botón correspondiente, el sistema lo detecta y cierra el juego
Precondiciones	Debe estar en la pantalla de menú o de pausa en la carrera
Postcondiciones	

Nombre	RF-ICR1
Resumen	Seleccionar circuito
Descripción	Se debe poder seleccionar el circuito en el cual correr la Carrera Rápida
Entradas	Click en el circuito deseado
Salidas	El circuito clickado se selecciona

Procesos	Cuando se hace click en el circuito, la información del circuito seleccionado se guarda para saber qué circuito cargar
Precondiciones	Debe estar en modo Carrera Rápida, antes de correr
Postcondiciones	Se debe actualizar el circuito seleccionado

Nombre	**RF-ICR2**
Resumen	Seleccionar número de vueltas
Descripción	Se debe poder seleccionar el número de vueltas a correr en la Carrera Rápida
Entradas	Click en el botón del número de vueltas deseado
Salidas	El número de vueltas se actualiza
Procesos	Cuando se hace click en el botón con el número de vueltas deseado, la información se guarda para saber el número de vueltas en futuras pantallas
Precondiciones	Debe estar en modo Carrera Rápida, antes de correr
Postcondiciones	Se debe actualizar el número de vueltas seleccionado

Nombre	**RF-ICR-ICAMP1**
Resumen	Seleccionar dificultad
Descripción	Se debe poder seleccionar la dificultad de la IA
Entradas	Click en el nivel de dificultad deseado
Salidas	El nivel clickado se selecciona
Procesos	Cuando se hace click en el nivel de dificultad deseado, la información del nivel seleccionado se guarda para saber qué circuito cargar
Precondiciones	Se debe seleccionar antes de correr la carrera rápida o antes de comenzar el torneo
Postcondiciones	En modo torneo, se debe actualizar los premios finales

Nombre	**RF-ICR-ICAMP2**
Resumen	Seleccionar modelo y marca de coche
Descripción	Se debe poder seleccionar modelo de coche deseado para correr la carrera rápida o el campeonato
Entradas	Click en el modelo deseado
Salidas	El modelo clickado se selecciona

Procesos	Cuando se hace click en el modelo deseado, se marca como seleccionado y se guarda la información
Precondiciones	Si es modo Campeonato, el coche debe estar desbloqueado
Postcondiciones	Se debe actualizar el modelo seleccionado

Nombre	**RF-ICAMP1**
Resumen	Desbloquear modelo de coche
Descripción	Se debe poder desbloquear un modelo de coche bloqueado
Entradas	Click en el modelo deseado
Salidas	El modelo clickado se desbloquea
Procesos	Cuando se hace click en el modelo deseado, se desbloquea para el perfil actual, y se podrá usar siempre
Precondiciones	Debe estar en modo Campeonato, el modelo debe estar previamente bloqueado, se deben tener suficientes monedas
Postcondiciones	El coche se desbloqueará permanentemente para ese perfil, se restará el precio del coche a las monedas del perfil

Nombre	**RF-ICARRERA1**
Resumen	Disponibilidad de menú de pausa
Descripción	Se debe de tener un menú de pausa disponible, con botones para ir al menú, escritorio o continuar en el juego. Mientras está el menú de pausa, el juego puede continuar
Entradas	Presionar Esc
Salidas	Se abre una pantalla de pausa
Procesos	Cuando se pulsa Esc, se abre un menú de pausa con botones funcionales
Precondiciones	Se debe estar dentro de una carrera
Postcondiciones	Se debe abrir una ventana

Nombre	**RF-ICARRERA2**
Resumen	Calcular puntos
Descripción	Se calculan los puntos y monedas ganados en función del modelo, dificultad, etc deseados
Entradas	
Salidas	
Procesos	Al finalizar la carrera, se calculan los puntos y monedas ganados
Precondiciones	Debe haber finalizado la carrera

Postcondiciones	Las monedas ganadas se deben añadir al perfil actual

Nombre	**RF-ICARRERA3**
Resumen	Ir al menú
Descripción	Al finalizar la carrera, volvemos al menú
Entradas	
Salidas	
Procesos	Al finalizar la carrera, después de calcular los puntos, volvemos al menú
Precondiciones	Debe haber finalizado la carrera y se deben haber calculados los puntos
Postcondiciones	Se debe cargar la pantalla de menú

Nombre	**RF-JCARRERA1**
Resumen	El coche debe poder acelerar, frenar y girar de forma que pueda completar el circuito
Descripción	El coche debe poder acelerar, frenar y girar de forma que pueda completar el circuito
Entradas	Teclas de Input
Salidas	Reacción en el coche
Procesos	Se detectan las teclas de input pulsadas, y en correspondencia se efectúan los cambios necesarios en el coche
Precondiciones	Debe estar en carrera
Postcondiciones	Debe haber un cambio en el coche

Nombre	**RF-JCARRERA2**
Resumen	El coche debe poder alcanzar altas velocidades
Descripción	El coche debe poder alcanzar altas velocidades en carrera
Entradas	
Salidas	
Procesos	
Precondiciones	
Postcondiciones	

Nombre	**RF-JCARRERA3**
Resumen	El coche debe tener un boost o nitro
Descripción	El coche debe disponer de una opción de nitro, que le de un impulso temporal al coche
Entradas	Tecla Input de nitro
Salidas	Impulso temporal
Procesos	Se genera un impulso mientras la tecla input esté pulsada

Precondiciones	La tecla Input de nitro debe estar pulsada, y debe quedar suficiente nitro
Postcondiciones	Se impulsa el coche

Nombre	**RF-JCARRERA4**
Resumen	El nitro debe de gastarse mientras se usa, y recargarse mientras no se usa
Descripción	Mientras se usa el nitro, éste debe ir gastándose hasta que no se pueda seguir usando. Mientras no se usa, se recarga
Entradas	Tecla Input de nitro (o no pulsada)
Salidas	Resta al nitro restante si se está usando nitro, suma al nitro restante si no se está usando nitro
Procesos	Se resta el nitro que se está usando si se está usando, se suma si no se está usando
Precondiciones	El nitro se debe estar usando para gastarlo, y no usando para recargarlo
Postcondiciones	Debe haber menos nitro (si se está usando) o más (si no se está usando)

Nombre	**RF-JCARRERA5**
Resumen	El ángulo de giro de la dirección del coche debe ser inversamente proporcional a su velocidad
Descripción	El ángulo de giro de la dirección del coche debe ser inversamente proporcional a su velocidad. Esto es para evitar giros bruscos a altas velocidades
Entradas	Velocidad
Salidas	Angulo máximo de giro
Procesos	Se calcula el ángulo máximo con una función cuyo input es la velocidad
Precondiciones	Debemos estar en carrera
Postcondiciones	Se debe actualizar el ángulo máximo. Si el nuevo ángulo es menor que el ángulo en el que las ruedas están en ese momento, las ruedas se centran hasta cumplir con el ángulo máximo

Nombre	**RF-JCARRERA6**
Resumen	Cuando un coche no pueda continuar, debe respawnear cerca de su última posición
Descripción	Cuando un coche no pueda continuar, debe respawnear cerca de su última posición. Esto es para evitar que algún coche se quede “pillado” en un muro, por ejemplo

Entradas	Velocidad del coche
Salidas	Nueva posición del coche
Procesos	Se lleva el coche al último checkpoint pasado, reseteando su velocidad
Precondiciones	El coche lleva más de x segundos a menos de y velocidad
Postcondiciones	El coche tendrá reseteadas sus propiedades físicas, como velocidad o velocidad angular

Nombre	**RF-JCARRERA7**
Resumen	Debe haber diferencias reales entre distintos modelos de coches
Descripción	Los modelos de coches de deben de crear de tal forma que haya diferencia notable entre coches tanto en manejo y aceleración como en tiempo de completar una vuelta
Entradas	
Salidas	
Procesos	
Precondiciones	
Postcondiciones	

Nombre	**RF-IA1**
Resumen	Los coches conducidos por IA deben ser capaces de acelerar, frenar y girar de forma autónoma para completar el circuito en el menor tiempo posible
Descripción	Los coches conducidos por IA deben ser capaces de acelerar, frenar y girar de forma autónoma para completar el circuito en el menor tiempo posible
Entradas	Posición del coche respecto al circuito, velocidad actual del coche
Salidas	Aceleración/freno, giro necesario para continuar completando la vuelta
Procesos	La IA dictamina cuánto acelerar, frenar o girar para completar la vuelta en el menor tiempo posible
Precondiciones	El coche debe estar controlado por IA no por el jugador
Postcondiciones	El coche debe completar la vuelta en el menor tiempo posible

Nombre	**RF-IA2**
Resumen	Los coches conducidos por IA deben poder completar el circuito en un tiempo

	suficientemente bajo para que suponga un reto para el jugador
Descripción	Los coches conducidos por IA deben poder completar el circuito en un tiempo suficientemente bajo para que suponga un reto para el jugador
Entradas	
Salidas	
Procesos	
Precondiciones	
Postcondiciones	

Nombre	**RF-IA3**
Resumen	Los coches conducidos por IA deben ser capaces de adelantar a otros coches
Descripción	Los coches conducidos por IA deben ser capaces de adelantar a otros coches, minimizando las colisiones en la medida de lo posible
Entradas	Posición del resto de coches
Salidas	Cambios en la aceleración/freno y giro
Procesos	La IA, a partir de la posición del resto de coches respecto al coche actual, dictamina los cambios en aceleración/freno y giro
Precondiciones	El coche debe ser conducido por IA
Postcondiciones	El coche debe modificar su trayectoria para adelantar satisfactoriamente

Nombre	**RF-F1**
Resumen	Los choques entre coches o con los bordes del circuito no deben entorpecer bruscamente la carrera
Descripción	Los choques entre coches o con los bordes del circuito no deben entorpecer bruscamente la carrera. Por ejemplo, no pueden dar vueltas de campana, o entorpecer a terceros coches, en la medida de lo posible.
Entradas	
Salidas	
Procesos	
Precondiciones	
Postcondiciones	

Nombre	**RF-F2**
Resumen	Los coches deben derrapar al girar bruscamente a altas velocidades, pero no demasiado

Descripción	Los coches deben derrapar al girar bruscamente a altas velocidades, pero no demasiado
Entradas	
Salidas	
Procesos	
Precondiciones	
Postcondiciones	

Nombre	**RF-F3**
Resumen	No se deben simular daños por golpes
Descripción	No se deben simular daños por golpes
Entradas	
Salidas	
Procesos	
Precondiciones	
Postcondiciones	

Requisitos no funcionales

Rendimiento

- El videojuego debe de consumir pocos recursos, para que un PC de gama media pueda ejecutarlo de forma fluida
- El videojuego debe adaptar su resolución a la del monitor, para que se visualice correctamente

Seguridad

- La ejecución del videojuego debe no suponer un problema de seguridad para el sistema

Fiabilidad

- El videojuego debe ser fiable y no tener bugs o crashes espontáneos, y debe tener el menor número de bugs posibles

Disponibilidad

- El videojuego debe poder estar disponible para ejecutarse en cualquier momento

Mantenibilidad

- El videojuego debe poder estar abierto a ampliación de contenido o características en un futuro

Portabilidad

- Tanto el videojuego compilado como el proyecto deben tener la posibilidad de ser ejecutados en cualquier PC Windows

Propiedad Intelectual

- El videojuego debe de usar contenido multimedia con derechos que permitan su uso

4. Validación de requisitos

En esta cuarta y última fase se comprueba que el conjunto de requisitos es correcto, definen correctamente el sistema y satisfacen las necesidades que el videojuego tiene para cumplir su objetivo.

Durante esta fase de validación de requisitos, se han realizado un conjunto de comprobaciones en el documento de especificación de requisitos, los cuales son los siguientes:

- Verificación de validez: Todos los requisitos obtenidos son válidos y viables en su cumplimiento
- Verificación de consistencia: Ninguno de los requisitos obtenido se contradice con algún otro requisito
- Verificación de completitud: En la verificación de completitud se busca verificar que los requisitos contemplan todas las tareas y limitaciones en el videojuego. Este apartado es muy difícil de cumplir en su totalidad. Dada la naturaleza de los videojuegos, la lista de tareas, limitaciones, interacciones entre elementos, etc. es interminable, y conseguir todos los requisitos posibles sería una tarea casi utópica.
- Verificación de realismo: Todos los requisitos pueden ser implementados utilizando la tecnología existente, y en el tiempo disponible.
- Verificabilidad: Todos los requisitos descritos son comprobables mediante pruebas del sistema, y podemos ver fácilmente si se cumplen o no en el proyecto final.

Hay que tener en cuenta, que, dada la naturaleza del desarrollo de videojuegos, se debe ser muy flexible con la especificación de requisitos a priori, pues los cambios en el videojuego durante su desarrollo son constantes.

4. Planificación y metodología

En este capítulo se va a explicar la planificación en el desarrollo del videojuego. Para ello, es conveniente dividir el proyecto en partes o módulos, donde cada módulo ha tenido su propia planificación de desarrollo. También se han asignado los tiempos que se deberían haber empleado a cada módulo.

También se explica qué metodología se ha usado. En este apartado explicaremos tanto el modelo de proceso software utilizado como las herramientas que se han empleado.

Antes de nada, al videojuego hay que ponerle un título. Para este videojuego, ha sido “Hell’s Driver”.

Módulos del videojuego

Puesto que el desarrollo de un videojuego es un proceso en el cual se deben tocar muchos campos muy distintos, por comodidad a la hora de desarrollar vamos a especificar estos campos y agruparlos en distintos módulos. De esta forma tendremos una visión clara y global del desarrollo del videojuego.

- Interfaz de usuario:
  En este módulo incluiremos el desarrollo de las distintas interfaces de usuario del videojuego (menú, selección de coche, marcador de velocidad y posición en carrera, etc.). La tarea de diseñar interfaces de usuario no es exclusivamente visual, ya que hay que integrarla con el videojuego, de forma que las interfaces sean interactivas (por ejemplo, que los botones funcionen). En este módulo por lo tanto no se incluye solo el diseño visual de las IU, sino que también su funcionalidad.

- Gestión de escenas:
  Llamamos escena a una fase o "escenario" del videojuego. Una escena es una fase aislada del videojuego donde ocurren una serie de cosas concretas. Por ejemplo, el menú es una escena. Cuando corremos en un circuito, estamos en otra escena. Cuando estamos eligiendo coche, estamos en otra escena. Por tanto, las escenas contienen los elementos de IU, modelos, etc. que componen el videojuego. Este módulo se centra tanto en la jerarquía de escenas como en el contenido de éstas. En el capítulo de diseño se verá de forma mucho más clara este concepto.

- Modelado:
  Este módulo consiste en el modelado de todos los elementos 3D que se van a usar en el videojuego, incluyendo coches y circuitos. Aunque todos los coches y circuitos se han modelado desde cero, hay algún que otro objeto (árboles y rocas) que se han obtenido de paquetes gratuitos. Estos objetos también se incluyen en este módulo. Puesto que los modelos necesitan al menos un material adjunto para que sean visibles, la gestión de materiales y texturas también están incluidas en este módulo.

- Mecánicas de control del vehículo:
  Uno de los módulos más importantes del proyecto. En este apartado se define cómo funciona el vehículo: Cómo funciona la aceleración, cómo funciona el giro, cómo funciona el boost, etc. También se definen las propiedades físicas del vehículo, como la masa, el agarre, la potencia, la potencia de frenado, la posición del centro de masa, etc. En este módulo incluimos también los sensores que tiene cada vehículo para obtener información del exterior.
  Este módulo no incluye la inteligencia artificial del vehículo. A lo que respecta con esta parte del proyecto, de dónde le vengan las órdenes como acelerar o girar (IA o jugador) le es indiferente.

- Inteligencia Artificial:
  El módulo más importante y extenso del proyecto. En este módulo se define tanto qué sistema de inteligencia artificial es elegido para los vehículos, como el proceso de entrenamiento de este sistema.

- Sonido:

Incluimos tanto los sonidos de los coches como la música de fondo.

- ❖ Programación del videojuego:
  Conforme se van desarrollando el resto de módulos, hay que cohesionarlos para que el videojuego como tal vaya avanzando. Esto incluye, desde programar en qué posición está cada coche en la carrera, hasta la gestión de perfiles de usuario. Por tanto, es un módulo fundamental y que requiere mucho tiempo y trabajo detrás.

Planificación de tiempo

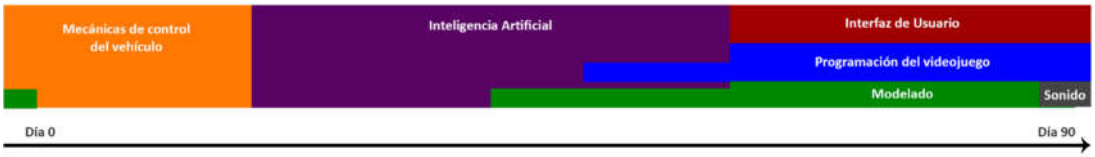
Ya tenemos claro los distintos módulos o tareas a desarrollar, ahora se explicará la planificación pensada para desarrollar el proyecto antes de empezarlo.

En el caso de este proyecto, el desarrollo se ha llevado a cabo desde el 1 de enero hasta el 18 de junio, teniendo en enero, febrero y marzo disponibles sólo las tardes. Por tanto, vamos a asumir que tenemos disponibles para el desarrollo 4h cada día desde enero hasta marzo, y 8h cada día mayo y junio. Esto hace un total aproximado de 420h, lo cual serían 90 días de trabajo, trabajando 8h cada día. La distribución de tiempos se ha planificado teniendo en cuenta esta escala de 90 días, para hacerlo más intuitivo.

La planificación sería la siguiente.

- ➔ Días 1-20: Se modelan un circuito y un coche de prueba, y se desarrolla el módulo de Mecánicas de control del vehículo. También se modelan algunos objetos de prueba para coger práctica a la tarea de modelar.
- ➔ Días 21-60: Se desarrolla el módulo de Inteligencia Artificial. En estos días realiza tanto la tarea de elegir el sistema de IA a implementar, como la implementación y entrenamiento. Paralelamente se modelan algunos coches y los cuatro circuitos, ya que son necesarios para entrenar la IA.
- ➔ Días 60-90: Se lleva a cabo casi la totalidad del modelado, del diseño de interfaces de usuario, de gestión de escenas, sonido y de programación de videojuego. Aunque bien algunos módulos como el modelado se tocaron bastante antes, la mayor carga de trabajo en modelado, IU y programación de videojuego se ha realizado en este tramo de tiempo.

Gráficamente, el desarrollo de los distintos módulos llevado a cabo podría representarse así:



La planificación que se pensó antes de comenzar el proyecto se ha llevado a cabo sin apenas cambios. Con el proyecto acabado, puedo decir que he cumplido con la planificación antes descrita. Cabe decir que los distintos módulos han estado más entrelazados, y por algunas circustancias relaconadas con la IA, ésta se ha alargado algo más de lo esperado, pero en general se ha cumplido con lo planificado.

Metodología

Cada módulo desarrollado necesita de sus propias herramientas. En este apartado voy a explicar las principales herramientas usadas.

Unity

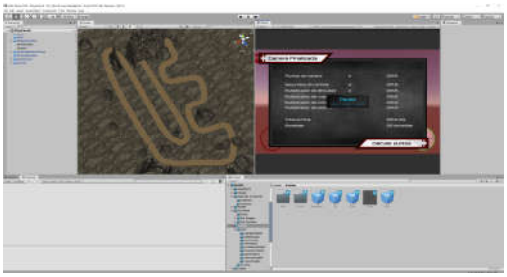
Unity es una “engine” o motor de videojuegos multiplataforma, creado por Unity Technologies. Unity se puede usar tanto en Windows como OS X y Linux, y tiene soporte de compilación para infinidad de plataformas: Desde Windows, OS X y Linux hasta Android, iOS o consolas como Xbox.

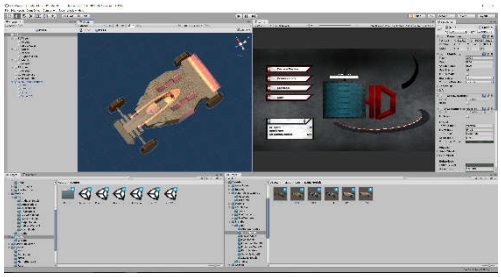
Unity está enfocado a desarrolladores independientes que no pueden crear su propio motor de juego, ofreciendo herramientas para desarrollar un videojuego gratis y potentes.

El motor gráfico utiliza OpenGL y Direct3D (Windows). El scripting viene a través de Mono, la implementación en código abierto de .NET Framework. Por tanto, los programadores pueden usar UnityScript, C# o Boo. En el caso de este proyecto se ha usado C#.

La versión de Unity usada es Unity 2019.3.

Unity tiene una gran cantidad de tipos de licencias, pero en nuestro caso sólo usaremos la licencia Unity Personal, la cual nos ofrece todas las prestaciones del motor siempre y cuando no superemos un tope de ingresos de 100.000 dólares.





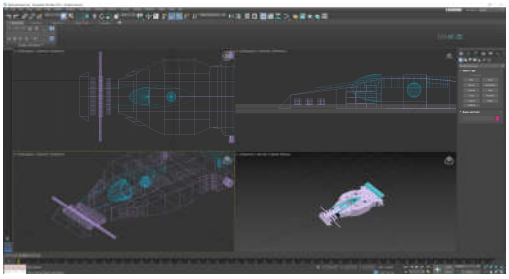
Autodesk 3DS Max

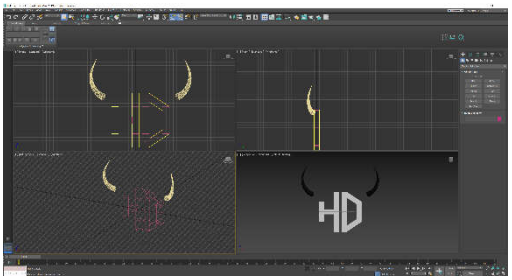
Autodesk 3DS Max es un programa de creación de gráficos y animación en tres dimensiones. 3DS Max es uno de los programas de modelado y animación más utilizado, especialmente para la creación de videojuegos, pero también para anuncios de televisión, películas, arquitectura, etc.

En este proyecto, el uso de este software ha sido exclusivamente para modelar objetos como coches y circuitos.

Autodesk 3DS Max no es software libre, sin embargo, he podido usarlo porque dispongo de licencia de estudiante.

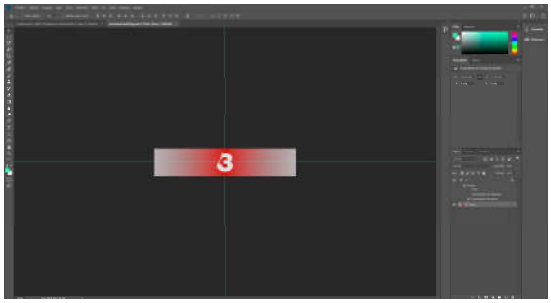
La versión utilizada ha sido Autodesk 3ds Max 2019.

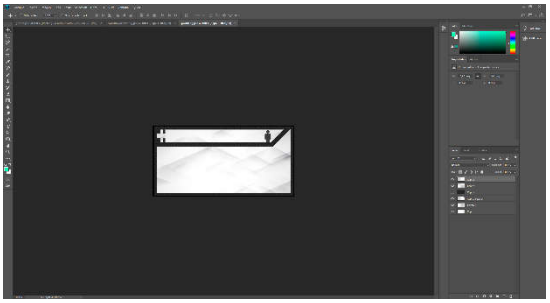




Adobe Photoshop CC

Adobe Photoshop es un editor de gráficos desarrollado por Adobe. Aunque su principal uso es el de retoque de fotografías, su versatilidad hace que se puedan crear todo tipo de imágenes desde cero.





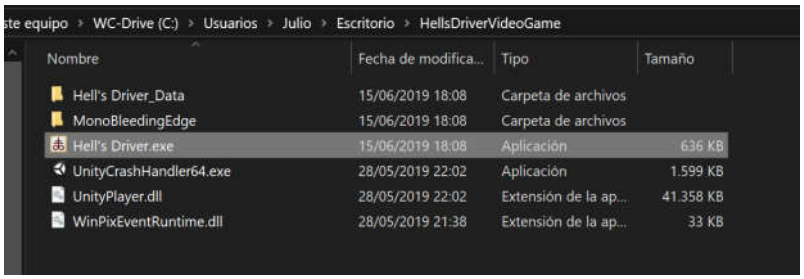
En este proyecto Adobe Photoshop ha sido usado principalmente para el desarrollo de Interfaces de Usuario.

5. Jugando al videojuego

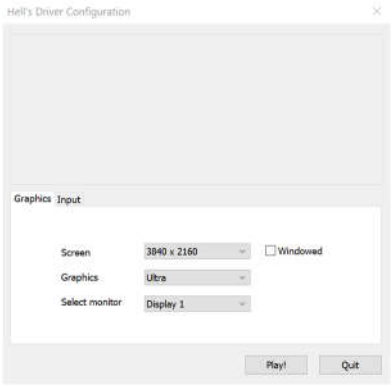
Para poder explicar correctamente cómo se ha hecho, primero debemos saber exactamente qué se ha hecho. Por tanto, antes de nada, vamos a dar un repaso al videojuego en sí, viendo cada una de las opciones, y enseñando cómo se juega. Una vez mostremos en qué consiste exactamente el videojuego, pasaremos a la parte donde se explica al detalle lo que hay detrás.

Iniciando el videojuego

La parte más evidente es abrir el propio videojuego. En la carpeta nos encontraremos varios archivos, abrimos el ejecutable.



Cuando lo abrimos, vemos la siguiente ventana:

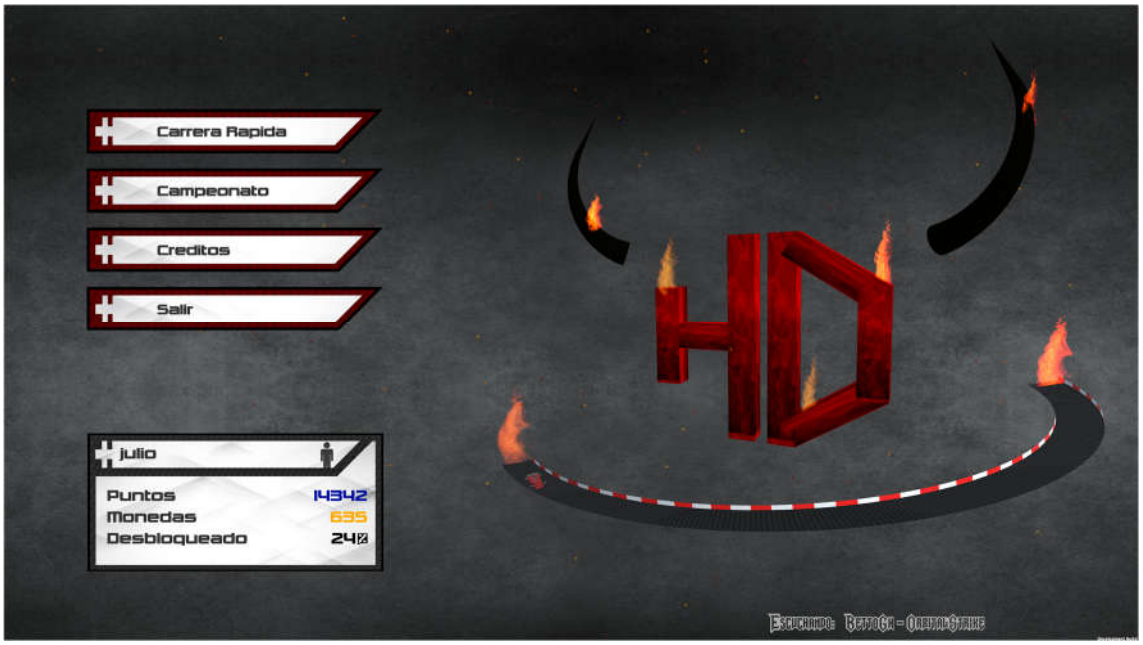


En esta ventana simplemente se elige qué resolución queremos para nuestro monitor. En el apartado “Graphics”, podemos elegir la calidad de texturas, pero en este videojuego todas tienen la misma calidad, por lo que es indiferente qué calidad elijamos en esta opción.

En la pestaña “Input” podemos modificar las teclas de entrada por defecto.

Le damos a jugar.

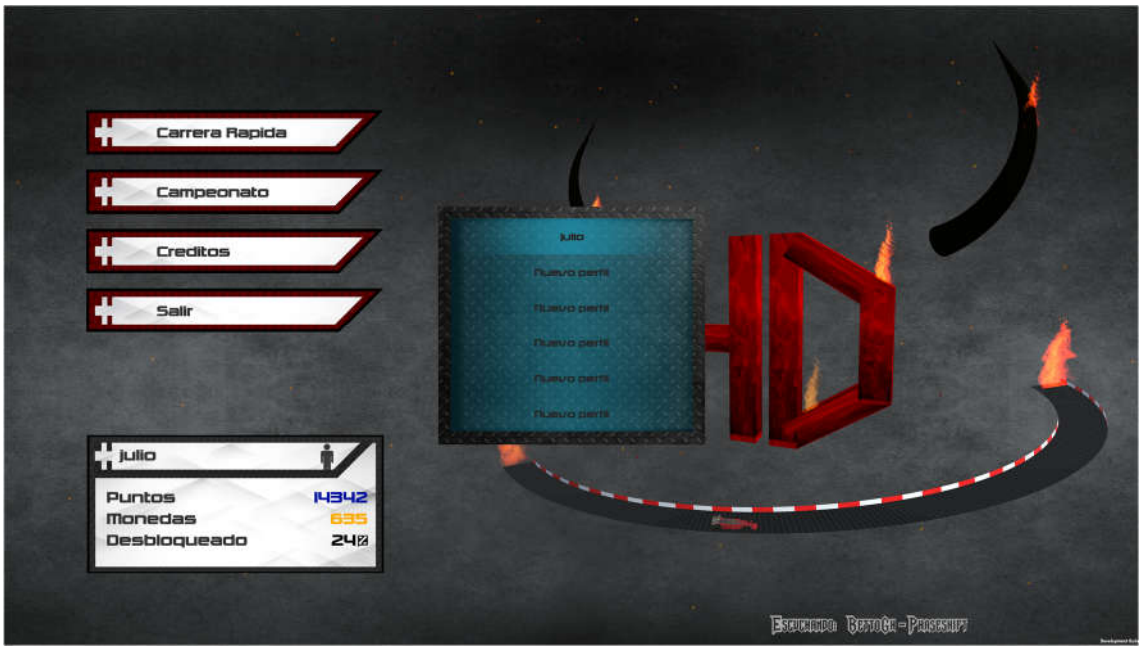
Lo primero que vemos es la escena de menú del videojuego. A la izquierda arriba tenemos cuatro botones, y a la izquierda abajo tenemos los datos de nuestro perfil. A la derecha vemos el logo del videojuego, y abajo a la derecha podemos leer el nombre de la canción que se está reproduciendo en ese momento.



Perfiles

Vamos a comenzar explicando la gestión de perfiles. Como ya se especificó en el análisis de requisitos, este videojuego tiene una gestión de perfiles, de forma que cada jugador puede tener su propio perfil, con sus propios puntos y monedas, y sus propios coches desbloqueados.

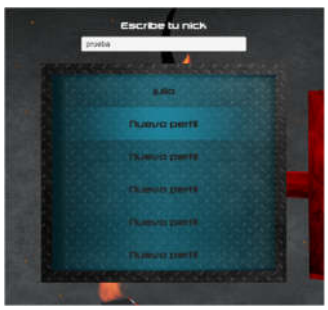
Para elegir un perfil, pulsamos en el nombre del perfil actual, en este caso, “Julio”.



Como vemos, se nos abre una ventana con 6 slots. Podemos crear por tanto un máximo de 6 perfiles. En las siguientes imágenes, mostraré el proceso de crear otro perfil, y también de borrarlo.

Primero vamos a crear un perfil. Para ello, seleccionamos algún slot libre, y arriba veremos un cuadro donde escribir nuestro nuevo nickname. Una vez escrito, pulsamos enter. Como vemos en las dos últimas imágenes, el nuevo usuario se ha creado correctamente, con ningún punto ni monedas, y solo algunos coches desbloqueados. En la última imagen vemos que nos sale una pequeña cruz al lado del perfil “Julio”. Esto sirve para eliminar el perfil. Dado que no podemos borrar nuestro propio perfil, al nuevo perfil no le sale la cruz.

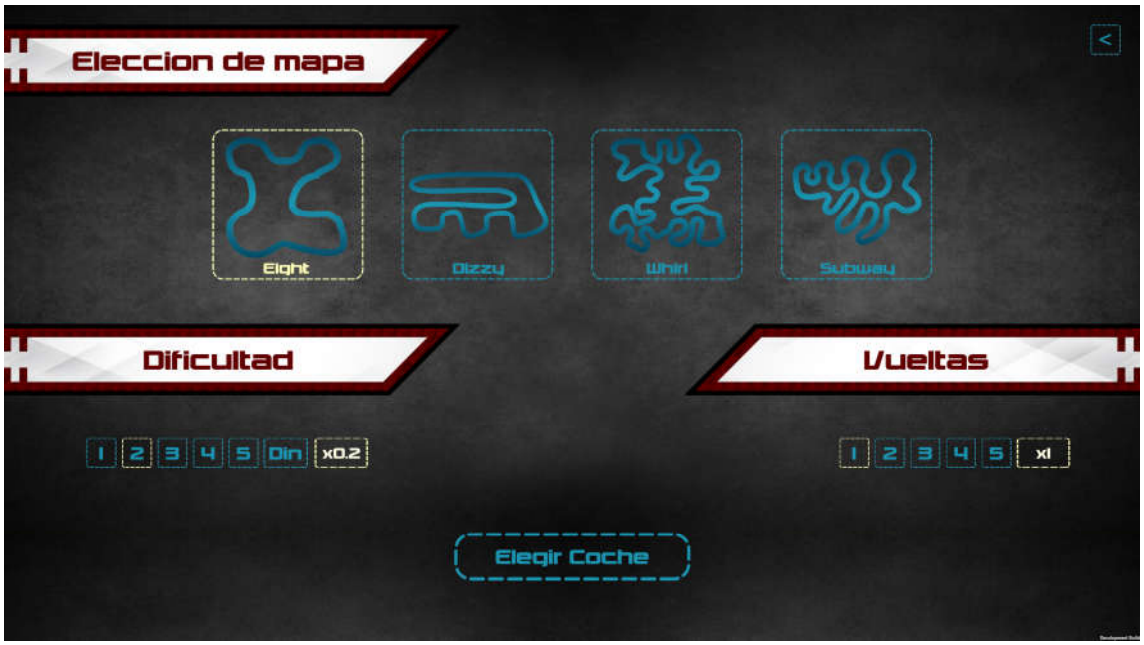
En el caso de que iniciemos el videojuego desde cero, sin ningún perfil, nos saldrá directamente el cuadro para crear uno.



Carrera Rápida

La carrera rápida consiste en una sola carrera, en el circuito que deseemos, con las vueltas y dificultad que deseemos, y con el modelo de coche que más nos guste (es este modo están todos desbloqueados).

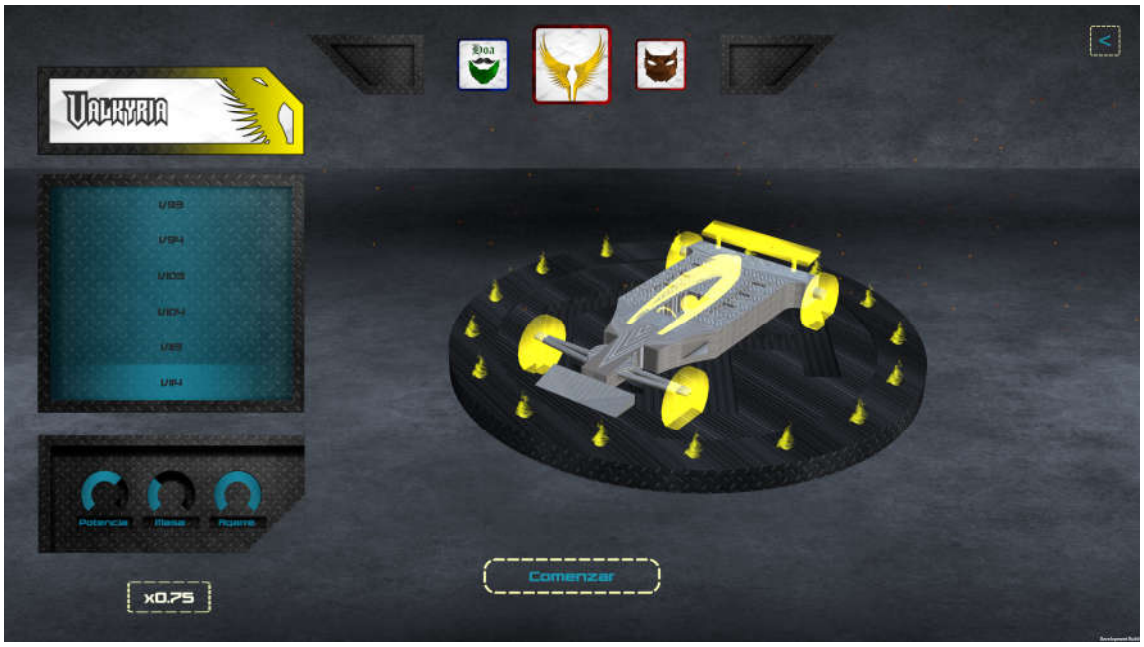
Si pulsamos el botón de carrera rápida se nos abrirá la siguiente escena:



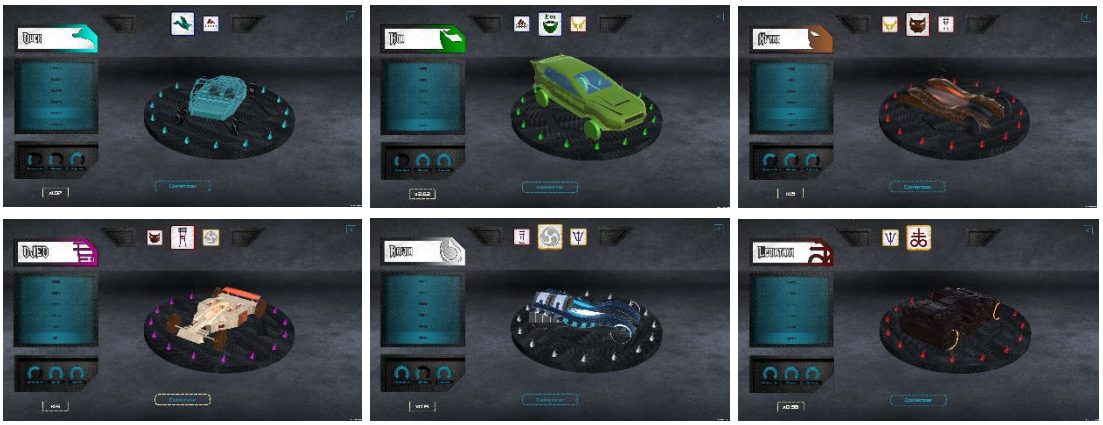
Como vemos, tenemos cuatro mapas a elegir, además del nivel de dificultad y número de vueltas. La dificultad se puede elegir entre 1 y 5, siendo 5 la más difícil. Vemos que a la derecha de la dificultad y el número de vueltas tenemos dos multiplicadores (x0.2 y x1 en este caso). Estos multiplicadores actúan sobre la puntuación final. El valor de los multiplicadores aumenta conforme se elige mayor nivel de dificultad o número de vueltas, lo que se traduce en una recompensa por jugar carreras difíciles o largas. Una vez elegida la dificultad y el número de vueltas, vamos a elegir coche.

Cuando pulsamos el botón de “Elegir Coche”, se nos abre una nueva escena. En esta nueva escena (imagen de abajo) vemos varias cosas. En la parte de arriba vemos en qué marca estamos actualmente. Podemos cambiar de marca con las flechas de izquierda y derecha, o con las letras A y D. Conforme vamos avanzando hacia la derecha, las marcas van teniendo coches más potentes. Las características de cada coche, marcas, etc. lo veremos en el capítulo de Diseño y Desarrollo.

Cada marca tiene seis modelos distintos, los cuales se pueden elegir en la izquierda. Cada uno de los modelos, tiene unas propiedades únicas: Potencia, masa y agarre. Además, tenemos otro multiplicador. Este multiplicador es mayor conforme peor características tiene el coche, de forma que el videojuego nos recompensa por jugar carreras con coches con peor características, y evita que siempre elijamos el mejor coche.



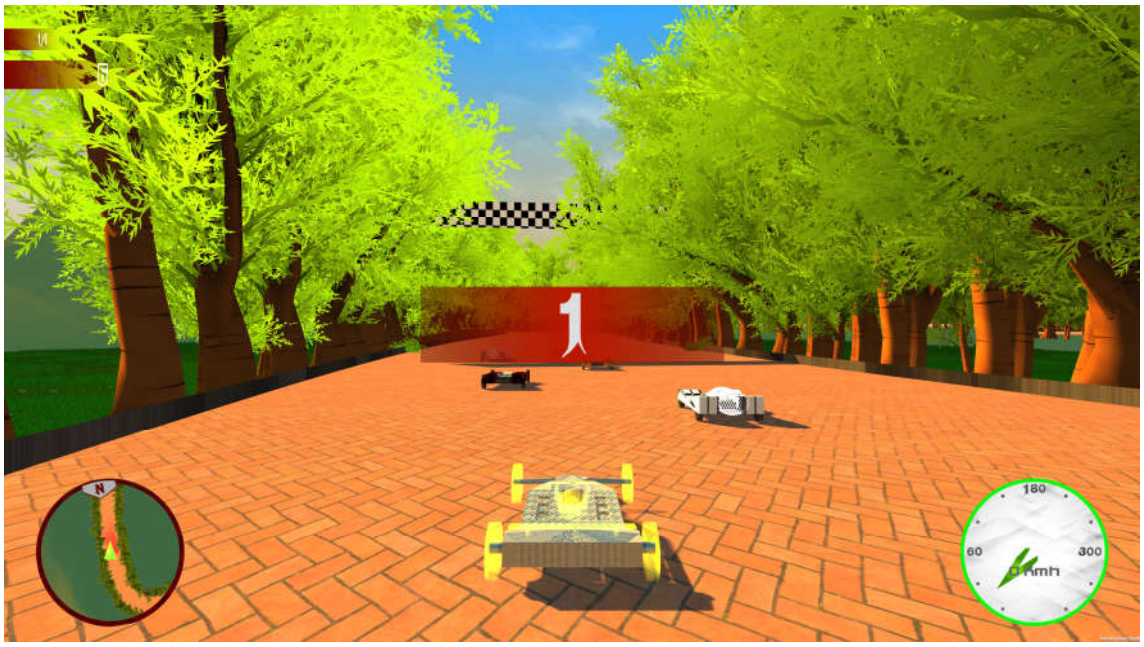
En total tenemos 9 marcas, lo que hacen un total de 54 modelos distintos.



Una vez hemos elegido marca y modelo, podemos darle a “Comenzar” y empezar a correr.

Una vez se nos abre la nueva escena, lo primero que vemos es una cuenta atrás. Cuando esta cuenta atrás llegue a cero, la carrera ha empezado. En la imagen podemos ver varias cosas. Abajo a la derecha tenemos el velocímetro. La circunferencia verde que lo rodea es el nitro/boost, que al empezar está completo. Si pulsamos el botón derecho lo usaremos y se irá gastando. Para evitar que se pulse de forma continuada, si estamos usando el boost y soltamos, y está a menos de la mitad, no podemos volver a usarlo hasta que se vuelva a recargar a la mitad.

Abajo a la izquierda tenemos el minimapa, el cual nos muestra nuestra posición y la de nuestros rivales.



Una vez acabamos la carrera, nos aparece la tabla con los resultados:

En esta tabla podemos ver las posiciones, el modelo de coche usado, y el tiempo que cada coche ha hecho.

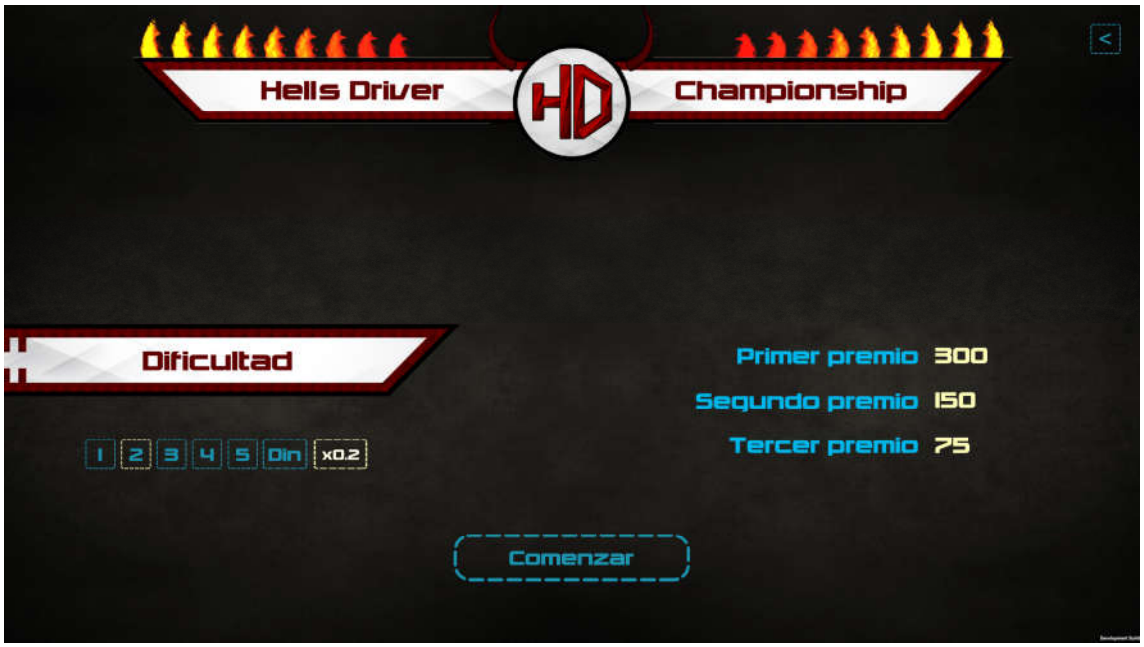
Vamos a calcular puntos:

Como vemos, los multiplicadores hacen efecto en el total de puntos y de monedas.

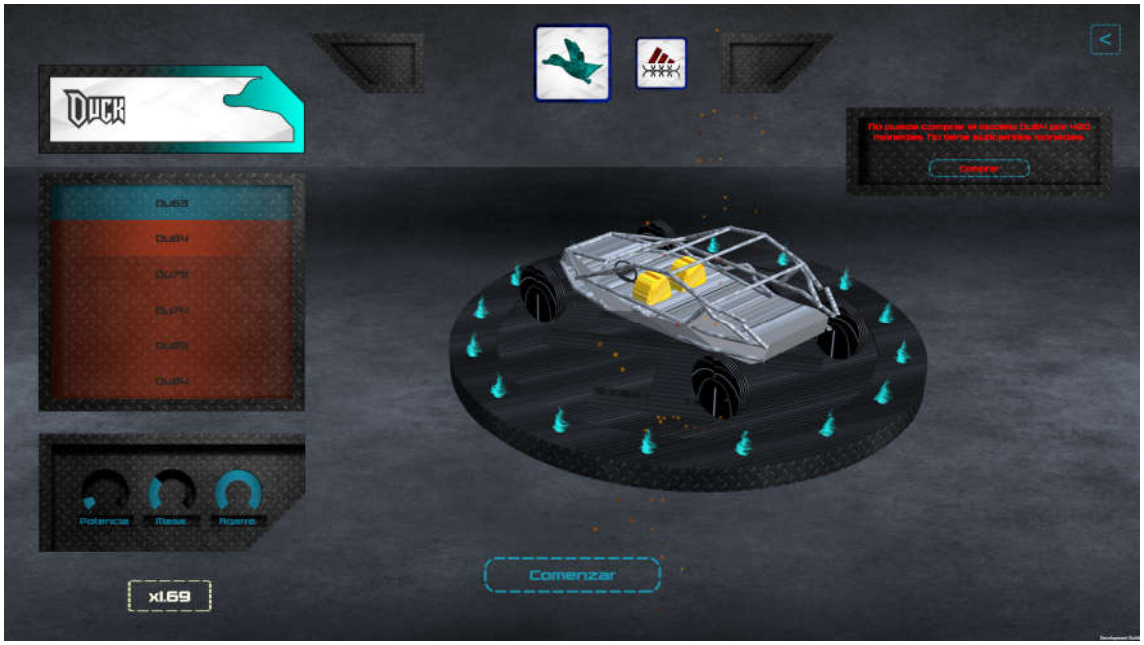
Campeonato

El modo campeonato consiste en cuatro carreras, una en cada mapa.

Como vemos en la imagen, podemos elegir la dificultad deseada para competir. A mayor dificultad, mejores son los premios.

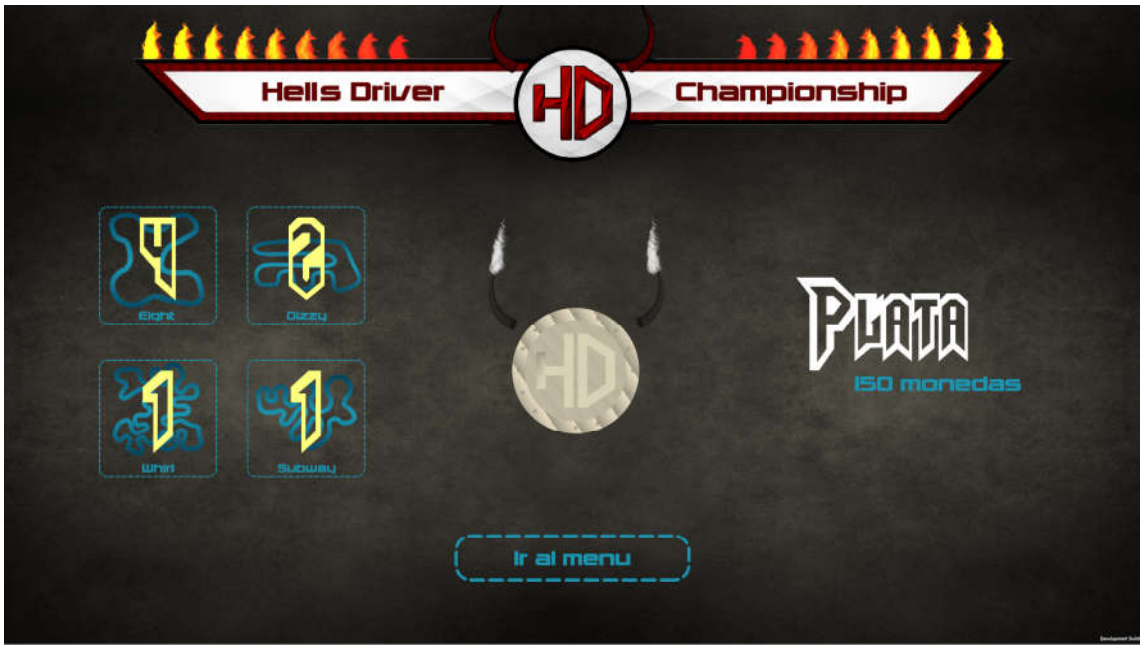


A la hora de correr en cada circuito, podemos elegir modelo de coche. Como vemos, en el modo campeonato tenemos la mayoría de coches bloqueados. Para desbloquearlos, hay que comprarlos como vemos a la derecha. En este caso no tenemos suficientes monedas, por lo que habrá que ahorrar.



Entre carrera y carrera, podemos ver los resultados en anteriores circuitos y en qué circuito tenemos que correr.

Una vez acabadas las cuatro carreras, finaliza el campeonato y se nos entregan los premios.



En cuanto a los modos de juego ya hemos visto todo. Hemos visto tanto el modo de carrera rápida como el modo campeonato. Habiendo enseñado en qué consiste realmente el videojuego, en el capítulo siguiente se procederá a explicar todo el proceso de desarrollo, desde los menús hasta la cómo se maneja el coche.

6. Diseño y desarrollo

Antes de nada, hagamos un resumen de lo que hemos visto en la memoria hasta ahora. Hemos comenzado con una breve introducción, explicando muy por encima en qué consiste este proyecto, así como mi motivación personal y objetivos. Se ha hablado de la historia de los videojuegos, y de la historia de la inteligencia artificial en éstos. Se ha comenzado a hablar del proyecto en sí haciendo un análisis de requisitos, y se ha hecho una planificación del desarrollo del proyecto. Y en el capítulo anterior, se ha enseñado cómo jugar al videojuego, para que veamos el resultado.

En este capítulo, vamos a empezar a hablar de cómo se ha desarrollado el proyecto. En “Diseño y desarrollo”, vamos a englobar todos los módulos antes planificados y explicar con gran nivel de detalle cómo se han llevado a cabo, exceptuando Inteligencia Artificial, la cual dada su importancia merece un capítulo adicional en esta memoria.