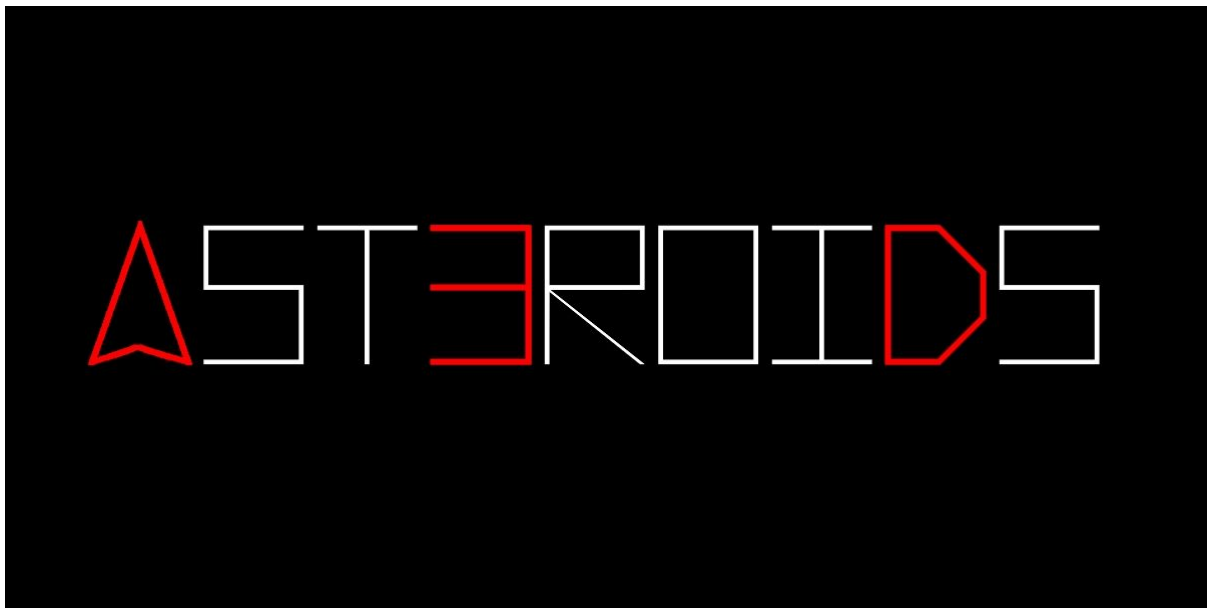
MEMORIA PROYECTO VIDEOJUEGO



Adrián Alejandre Escriche (682237)

Jorge Andrés Galindo (679155)

Alejandro Fernández Poza (679890)

Miguel Jorge Galindo Ramos (679954)

ÍNDICE

5. Arte
    5.1. Estilo
    5.2. 2D
    5.3. Música y FX
    5.4. 3D
        5.4.1. Modelo 3D
        5.4.2. Texturas
        5.4.3. Shaders

6. Inteligencia Artificial
    6.1. Red neuronal simple para el movimiento de los ovnis
    6.2. Red neuronal entrenada con algoritmos genéticos
    6.3. Red de Elman
    6.4. Red neuronal simple para los disparos de los ovnis
    6.5. Inteligencia Artificial basada en reglas
    6.6. IA 3D

7. Tecnologías y herramientas de desarrollo
    7.1. Lenguaje de programación
    7.2. Librerías
    7.3. Control de versiones
    7.4. Modelado 3D

8. Problemas encontrados
    8.1. Nuevo tipo de proyecto
        8.1.1. Diseño adaptativo a los problemas que van surgiendo
        8.1.2. Tiempo real
        8.1.3. Pruebas con aleatoriedad
        8.1.4. Compilación de librerías para un entorno inusual
    8.2. Inteligencia Artificial
    8.3. Ajustes de escala
    8.4. Coordenadas de textura
    8.5. Movimiento libre de la nave

9. Reparto de tareas y cronograma del proyecto

1.INTRODUCCIÓN

1.1 Concepto del juego

La propuesta de videojuego consiste en un clon del mítico videojuego 'Asteroids' lanzado en máquinas recreativas en 1979 por la compañía Atari. El juego consiste en una nave cuyo objetivo es destruir todos los asteroides y ovnis que se encuentre en su camino para poder sobrevivir a la vez que acumula puntos. Cada jugador puede utilizar su estrategia preferida para sobrevivir en el espacio. Algunos jugadores preferirán mover lo mínimo posible su nave y centrarse en atacar aquellos asteroides que más se aproximan mientras que otros recorrerán el espacio buscando posiciones ventajosas para disparar a los asteroides y ovnis.

1.2 Características principales

Ast3roiDs es un juego muy entretenido a la par que sencillo, tanto en jugabilidad como en estética, ya que utiliza gráficos vectoriales. Es un juego cuya duración depende exclusivamente de la habilidad del jugador para hacer que la nave se mantenga intacta mientras la partida y su dificultad avanzan. Mientras no se destruya la nave un número de veces concreto, el jugador debe destruir elementos tales como ovnis y asteroides, dotados con diferentes comportamientos y propiedades.

Género	**Estilo**	**Ambientación**
Arcade	Retro	Espacial

1.3 Público dirigido

A cualquier jugador nostálgico de las máquinas recreativas de videojuegos arcade y a los nuevos jugadores que deseen conocer, aprender y disfrutar de este gran juego. En general a cualquiera que quiera poner a prueba sus habilidades y reflejos al mando de una nave espacial.

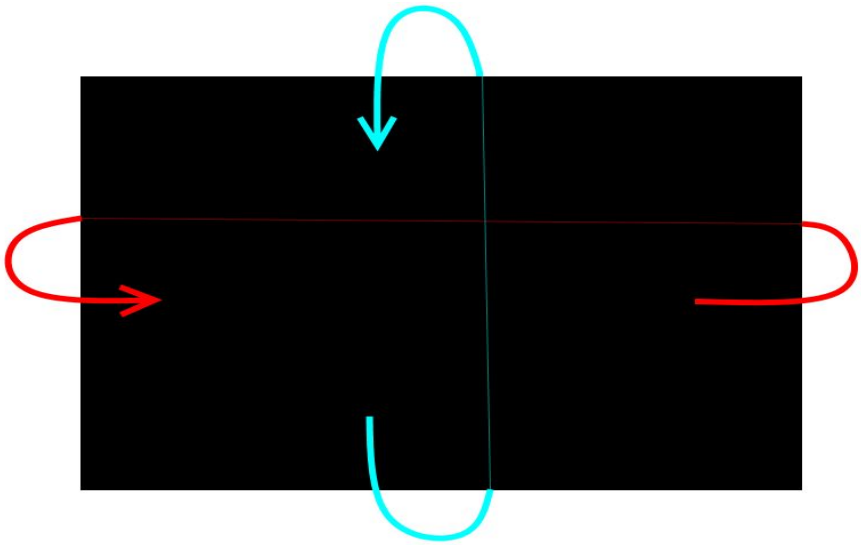
1.4 Plataforma

PC, sistema operativo Windows 7 y posteriores. Arquitectura x86-64.

2.MECÁNICAS DEL JUEGO

2.1 Resumen de la jugabilidad

El jugador controla una nave triangular que puede rotar en ambos sentidos. Para desplazarse por el mapa, puede acelerar y moverse en la dirección a la que esté apuntando dicha nave. El movimiento de la nave se caracteriza por ser inercial, es decir que la nave no frena instantáneamente al dejar de acelerar sino que la velocidad en la dirección de movimiento disminuye paulatinamente. El mapa, es el plano de un espacio esférico, por lo que al atravesar uno de sus límites, se aparece por el límite opuesto.



Una de las habilidades principales de la nave es el disparo. Al pulsar el botón de disparo, desde el morro de la nave se genera un láser que viaja por el espacio en línea recta hasta que recorre una distancia máxima o impacta con algún elemento. La nave solo puede mantener 4 disparos activos simultáneamente. Otra de las habilidades que se posee es la del salto al hiperespacio, lo que teletransporta la nave a cualquier posición del plano, incluyendo al interior de un asteroide. Además, las naves espaciales poseen la valiosa capacidad de reaparecer tras ser destruidas, pero hay un número máximo de veces que esta habilidad puede usarse, por lo que es vital no perder todos los usos ya que deja la nave vulnerable a cualquier amenaza.

La nave no está sola en el espacio, sino que hay asteroides que destrozan cualquier nave con la que impactan, aunque no impactan entre ellos. Estos asteroides se clasifican en tres categorías según su tamaño. Los de mayor tamaño tienen una velocidad más lenta y al ser interceptados por un disparo, se dividen en dos de tamaño medio. Al igual que los de un tamaño mayor, los medianos también se dividen al recibir un impacto en dos de tamaño pequeño. La trayectoria de los asteroides es siempre una línea recta con velocidad constante desde que aparecen por primera vez, ya sea por un cambio de zona o por originarse de un impacto.

Los asteroides no son los únicos elementos no humanos del espacio sino que pueden aparecer platillos volantes u ovnis. Estos alienígenas viajan en ovnis e intentarán destruir la nave con mayor o menor precisión. Al igual que los asteroides, hay ovnis de varias clases, los de mayor tamaño no controlan sus armas de manera precisa lo que disminuye la amenaza. Pero los de menor tamaño son más avanzados y consiguen una gran precisión por lo que pueden destruir la nave en cualquier despiste.

Cuando una zona espacial queda libre de asteroides aparecen más automáticamente, 2 más que en la zona anterior (al comienzo del juego, aparecen 4 asteroides). Al destruir cualquier tipo de amenaza, ya sea un asteroide o un ovni, se obtiene una cantidad de puntos en función de su peligrosidad. Si bien no existe ninguna pantalla final, el número máximo de asteroides que puede aparecer en una pantalla es 12.

En la pantalla de juego puede verse en todo momento la puntuación actual y el número de reapariciones disponibles.

2.2 Elementos

2.2.1 Nave:

Es el elemento controlado por el usuario, puede rotar sobre sí mismo, avanzar en la dirección a la que apunta, disparar hasta 4 proyectiles a la vez y puede saltar al hiperespacio. Si la nave colisiona con un asteroide, un ovni o un disparo enemigo explotará. Tras explotar puede reaparecer en el centro de la pantalla un número limitado de veces. Este límite aumenta en una unidad cada vez que el jugador obtiene 10000 puntos.

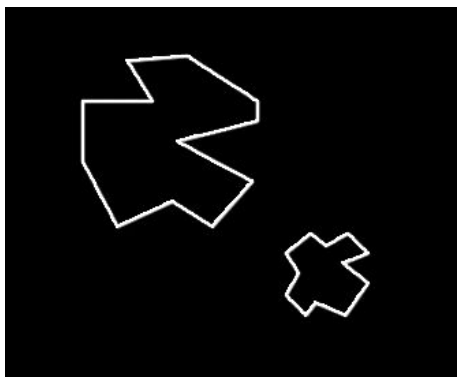
Su movimiento siempre sufre aceleración, ya sea porque el jugador mantiene pulsada la tecla de acelerar o porque está desacelerando automáticamente. Su velocidad tiene un límite máximo, ya que no tendría sentido que la nave fuera más rápida que los proyectiles que lanza y además sería excesivamente complicado de controlar. Mientras se acelera es posible girar la nave, dando completa libertad de movimiento al jugador. Esta puede girar a la vez que se acelera.La aceleración siempre se aplica en la dirección en la que apunta la nave, y sobre la velocidad que ya posee en ese momento. Esto significa que si llevando una cierta velocidad en una dirección la nave gira 180 grados y acelera, la nave no se detendrá instantáneamente, sino que primero reducirá su velocidad a 0 y si se continúa pulsando la tecla de aceleración entonces obtendrá velocidad en la nueva dirección.

Saltar al hiperespacio conlleva el riesgo de que la nave explote. No solo por aparecer en el interior de un asteroide sino que existe la posibilidad de que la nave explote sin motivo aparente. Esto hace que saltar al hiperespacio se utilice sólo como último recurso cuando la nave no tenga ninguna opción de escapatoria segura. Además, tras realizar esta acción debe transcurrir un tiempo mínimo de "recarga" hasta poder repetirla.

Las diferentes acciones disponibles a la nave son independientes entre sí, es decir, el jugador puede acelerar, girar, disparar y saltar al hiperespacio simultáneamente, siempre y cuando la acción no esté en tiempo de recarga.

La nave posee 6 puntos para detectar colisiones, los tres vértices del triángulo base que forma la nave y el punto medio de cada arista, con ello se puede detectar colisiones lo suficientemente precisas con círculos (ovnis y asteroides). Para detectar colisiones con los disparos, se ha calculado si alguno de los extremos de los disparos está en el interior del triángulo base.

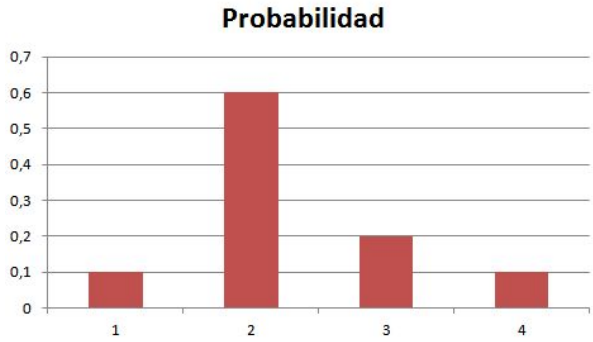
2.2.2 Asteroides:



Elementos que se mueven en línea recta con velocidad aleatoria con un límite máximo inversamente proporcional a su tamaño y se le ha añadido una constante multiplicativa y un componente aleatorio para hacerlo menos predecible. A grandes rasgos, se puede decir que, cuanto mayor es un asteroide, menor su velocidad. Los hay con varias formas, aunque no afecta a sus atributos. Se clasifican en 3 tipos de asteroides según su tamaño: El mayor tiene una velocidad baja por lo que al ser destruido proporciona pocos puntos. El pequeño es más rápido y difícil de alcanzar con los proyectiles por lo que la recompensa de puntos por destruirlos es grande. Hay un tipo intermedio con una velocidad entre las dos anteriores, un tamaño mediano y una puntuación acorde a sus características. Los asteroides grandes y medianos no son destruidos al ser alcanzados por un disparo, en lugar de

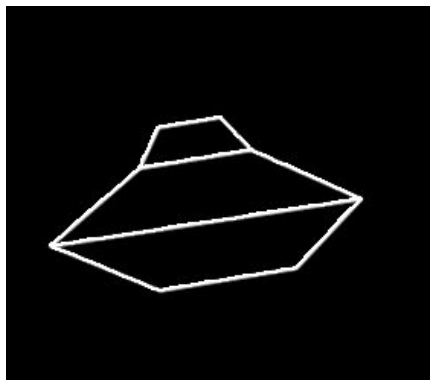
ello, se dividen en dos del tamaño inmediatamente inferior que, una vez alcanzado por la nave, un ovni, o un proyectil, es destruido por completo. Los asteroides no pueden colisionar entre sí.

Al ir pasando niveles, la velocidad de todos los asteroides aumenta con un factor 1+nivel/20 para aumentar la dificultad. El número de asteroides producidos de la destrucción de uno grande o mediano es aleatorio en el rango [1,4] siendo la probabilidad de dicho número la siguiente:



La colisión con los asteroides se calcula si la distancia de alguno de los puntos de detección está dentro del círculo aproximado según el tamaño del asteroide debido a su parecido a la forma circular. Para detectar la colisión con otros elementos circulares, se calcula la distancia entre los dos centros y se comprueba la intersección según los radios. En ningún caso se comprueban colisiones entre asteroides.

2.2.3 Ovnis:



Los ovnis son elementos capaces de cambiar su trayectoria y de disparar. Los hay de dos tipos, los grandes son torpes, apenas varían su trayectoria lineal y disparan en direcciones aleatorias, por ello dan una puntuación menor que los del siguiente tipo. Los ovnis pequeños son más rápidos y dotados de inteligencia, son capaces de evitar asteroides cercanos y disparar a la nave. Por ello, los ovnis de menor tamaño dan una gran cantidad de puntos.

Para destruirlos y obtener puntos el jugador tiene una única opción, conseguir que uno de sus proyectiles intersecte con ellos. También puede esperar a que choquen con un asteroide o colisionar directamente contra ellos pero en ninguno de los casos obtendrá puntos y en el segundo además perderá una vida.

La colisión con los ovnis es idéntica a la de los asteroides ya que también se ha aproximado a un círculo.

2.3 Flujo de juego

2.3.1 Jugando:

El usuario pone a prueba sus habilidades manejando la nave intentando que esta sobreviva a las distintas amenazas que se le pongan delante mientras obtiene la mayor cantidad de puntos posible destruyendolas. Tras eliminar todos los asteroides aparece un mayor número de asteroides de tamaño grande que en la etapa que se acaba de superar.

El jugador se enfrenta ante un número ilimitado de fases, en cada nivel el número de asteroides aumenta en dos hasta un máximo de 12 así como su velocidad. Del mismo modo aumenta la probabilidad de que aparezcan ovnis más difíciles.

2.3.2 Fin de partida:

Una vez que la nave haya sido destruida sin posibilidad de reaparición, al usuario se le mostrará su puntuación final permitiéndole guardarla si muestra ser lo suficientemente buena para acreditar al usuario como uno de los mejores.

2.4 Control de jugador

2.4.1 Interacción con la nave:

El juego cuenta con unos controles para manejar la nave muy sencillos y personalizables que cualquier usuario puede entender independiente de su experiencia previa.

- **Botón ‘Acelerar’ (A)**: El botón de acelerar que hará que la nave avance en la dirección a la que apunta..
- **Botones de ‘Dirección’ (Izquierda/Derecha)**: Los botones de dirección harán que la nave gire sobre sí misma y apunte en una dirección distinta..

- **Botón de ‘Disparar’ (D)**: El botón de disparar generará un proyectil que viajará en línea recta en la dirección en la que apunte la nave. Solo podrá haber como máximo 4 disparos simultáneos.
- **Botón de ‘Hiperespacio’ (Barra espaciadora)**: El botón de hiperespacio provoca que la nave desaparezca momentáneamente para aparecer en un punto aleatorio de la pantalla (incluyendo el interior de un asteroide). Es posible que la nave explote sin colisionar con ningún asteroide.

2.4.2 Interacción con el entorno:

La nave que maneja el usuario puede impactar con cualquier otro elemento de la pantalla como pueden ser los asteroides, los ovnis o disparos provocando el obligatorio uso de la habilidad de reaparición o la destrucción total de la nave si ya no quedan. Además la nave puede disparar y dichos proyectiles interaccionan con el entorno provocando la división o la destrucción de otros elementos.

2.4.3 Interacción con la interfaz:

Para navegar por las diferentes pantallas de menú se usan las teclas de dirección del teclado, Arriba y Abajo para subir y bajar por los menús e Izquierda y Derecha para modificar las distintas opciones. Para confirmar se usa la tecla Intro.

2.5 Puntuaciones:

Los puntos que se obtienen por la destrucción de las amenazas se pueden ver en la tabla que se muestra a continuación:

Elemento	**Puntuación**
Asteroide grande	20
Asteroide mediano	50
Asteroide pequeño	100
Platillo volante grande	200
Platillo volante pequeño	1000

2.6 Opciones:

Algunas de las opciones con las que se podrá personalizar la experiencia de juego se muestran a continuación:

- **Resolución:** Se da a elegir al usuario entre diferentes resoluciones que mantienen el aspecto de 4:3 y soporta su monitor.

- **Pantalla completa:** Es posible jugar en pantalla completa o en ventana flotante.

- **Antialiasing:** Se da a elegir al usuario entre niveles típicos de antialiasing aunque no es seguro que su sistema los soporte. Es recomendable utilizar el máximo posible ya que no se espera ningún decremento en la experiencia de juego en sistemas no muy antiguos.

- **Volumen:** Permitirá modificar el volumen del sonido y música producido por el juego. De esta forma el jugador podrá jugar en silencio si así lo prefiere.

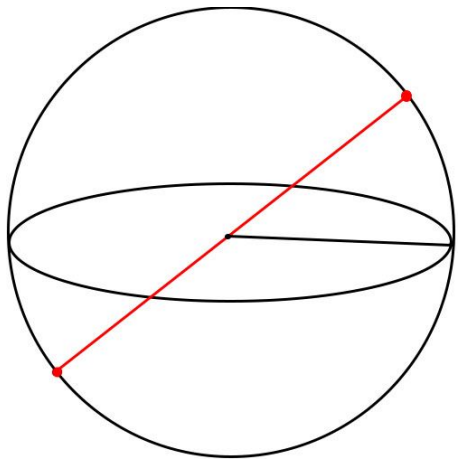
- **Cambio de controles:** Permitirá a cada jugador configurar los controles del juego como él desee.

- **Cambio de color:** Al utilizar el juego gráficos tan simples y contar con únicamente dos colores diferentes se permitirá al jugador elegir entre blanco, rojo, verde y azul para los colores del todos los elementos del juego.

3. MECÁNICAS DEL JUEGO 3D

3.1 Resumen de la jugabilidad

La versión tridimensional del clásico Asteroids expande la jugabilidad del original para permitir el movimiento de la nave en un espacio con volumen. De forma similar a la versión 2D, el jugador controla una nave espacial que se desplaza por un mapa con movimiento inercial. En esta ocasión, la nave puede encontrarse en cualquier rotación y el espacio por el que se mueve es realmente esférico. Si la nave, un ovni, un disparo o un asteroide fuera a escapar del espacio designado para el juego, la esfera de juego, reaparecerá en la antípoda de dicho espacio de juego conservando su inercia y orientación.



Las habilidades de la nave son las mismas que en el espacio 2D, eso sí, adaptadas a un espacio esférico tridimensional. La característica inercia del movimiento se ha mantenido lo más parecida posible a la original, aunque los mismo valores que en 2D no provocan la misma movilidad en el 3D.

Los demás elementos del juego se han mantenido igual que en la versión bidimensional, es decir, hay asteroides de tres tamaños con un movimiento rectilíneo en el espacio y con la capacidad de división. También hay dos tipos de ovnis, uno aleatorio y otro más “inteligente”.

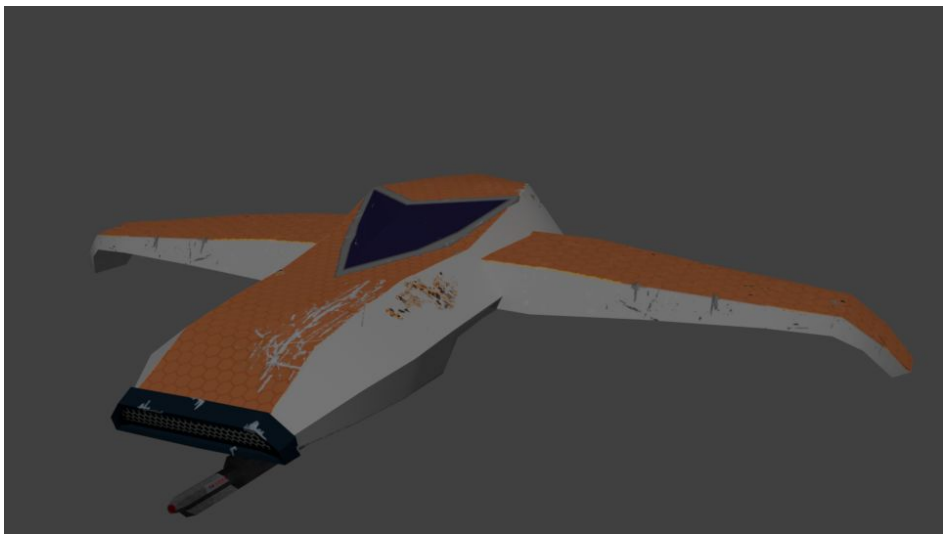
Cuando una zona espacial queda libre de asteroides aparecen más automáticamente, 2 más que en la zona anterior (inicialmente hay 6). Al destruir cualquier tipo de amenaza, ya sea un asteroide o un ovni, se obtiene una cantidad de puntos en función de su peligrosidad. Si bien no existe ninguna pantalla final, el número máximo de asteroides que puede aparecer al comienzo de un nivel es 20.

Mientras se está jugando se puede ver en todo momento la puntuación que se posee, las vidas restantes y dos minimapas (radares) que muestran la posición global (no relativa a la nave) de las amenazas y de la nave.

3.2 Elementos

3.2.1 Nave

La mecánica, habilidades y características de la nave manejada por el usuario son idénticas a la de la nave del juego en 2D pero trasladadas a un espacio tridimensional. Como característica añadida, la nave posee unos radares que indica su posición y la de las amenazas en las coordenadas de la esfera espacial.

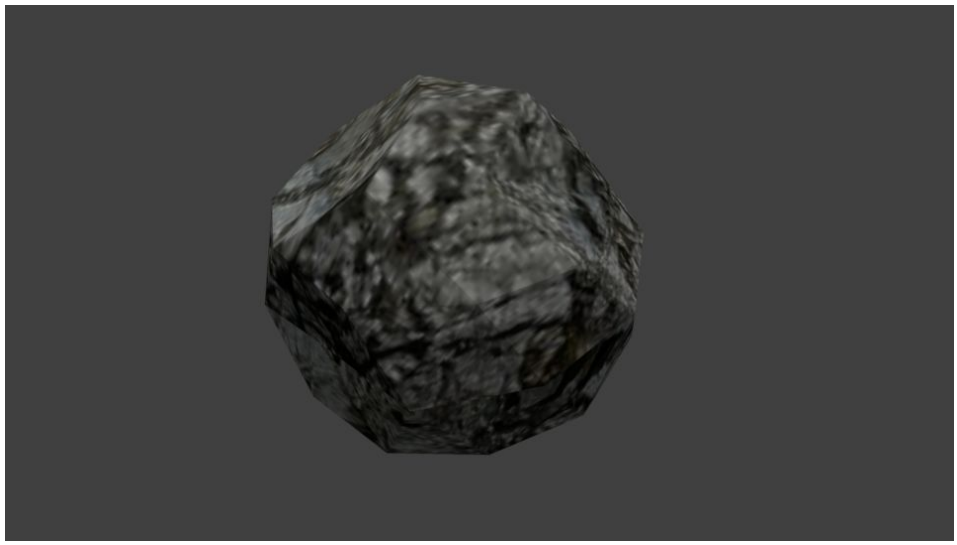


Como el modelo de la nave en 3 dimensiones es más complejo, se ha aproximado con una esfera por lo que la detección de colisiones se hace mediante la distancia de un punto al centro dado el radio o la distancia entre centros dados dos radios. Para ajustar más las colisiones se podrían usar distintas esferas, con el incremento de tiempo en las colisiones. Tras hacer pruebas, el error en la colisión con una esfera apenas se nota.

Los disparos de la nave son alargados, pero se han aproximado con el punto central de estos ya que la precisión es suficiente para que parezca más real, penetran un poco dentro del objeto con el que colisionan antes de destruirlo.

3.2.2 Asteroides

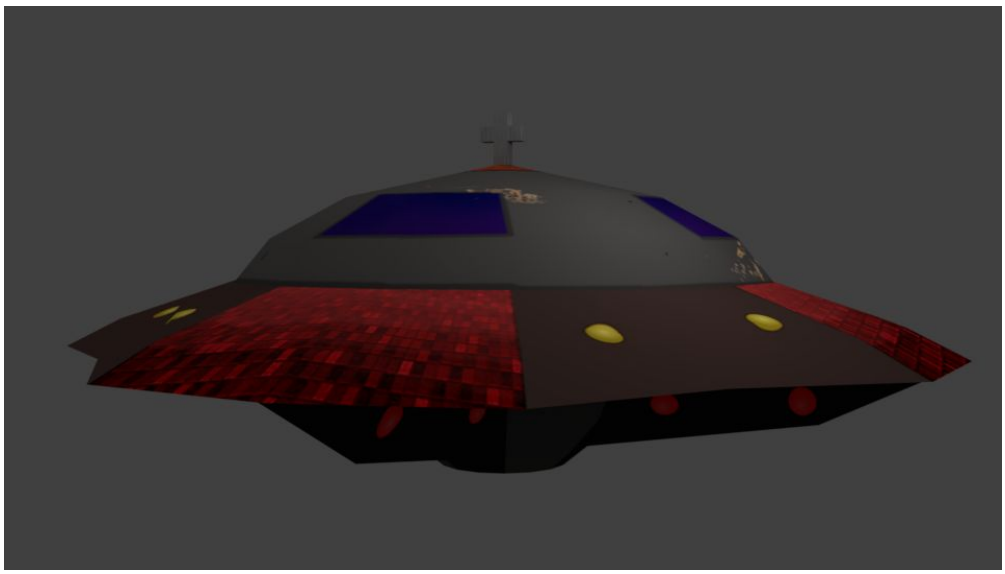
Los asteroides siguen la misma lógica que en el juego 2D, también se dividen entre 1 y 4 asteroides de menor tamaño con las probabilidades mostradas en el apartado 2.2.2. También escala la velocidad con el mismo factor. Como modificación, se ha incrementado el número de asteroides que salen en el nivel 1 a 6 y el número máximo de estos al comenzar un nivel a 20.



Su modelo es bastante esférico por lo que se usa una sola esfera para aproximar las colisiones.

3.2.3 Ovnis

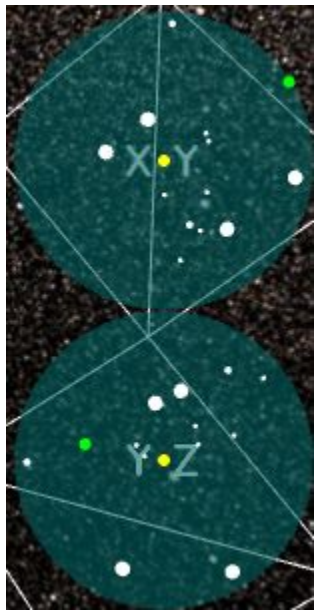
Los ovnis siguen las mismas pautas que en el 2D, se ha realizado un nuevo modelo 3D con su textura y una nueva inteligencia artificial para adecuarlo al espacio tridimensional, así como un rediseño de las reglas de evasión de asteroides.



Se han mantenido los dos tipos de ovnis, el aleatorio y el inteligente, su probabilidad de aparición es la misma que en la versión 2D así como el incremento de esta al pasar niveles.

Los ovnis también se aproximan como una esfera por los tanto todas las colisiones son mediante esferas, menos con los disparos. Los disparos de los ovnis a pesar de ser esféricos, se detectan solo con el punto central.

3.2.4 Minimapas



Para ayudar al usuario a esquivar los asteroides que están detrás de la nave y que no se ven, se han agregado dos radares que muestran las posición de las amenazas dentro de la esfera. Uno de ellos muestra el plano XY de la esfera y otro el YZ, con lo que mirando ambos mapas, se puede saber la posición exacta de todos los elementos. La posición de los planos es fija, por lo que la posición mostrada es absoluta respecto de la esfera espacial. Cabe destacar que los radares no rotan con la nave.

La nave se muestra mediante un círculo amarillo, sus disparos con círculos de menor tamaño y del mismo color. Los asteroides se muestran como círculos blancos de tres tamaños, según el tamaño del asteroide que representan. Por último se muestran los ovnis con un círculo de color verde, pero no sus disparos.

3.3 Control del jugador

El control del videojuego en 3D es más complicado que en 2D y requiere del uso de, como mínimo, el ratón para poder navegar en el espacio 3D con comodidad. Se mantienen los controles ya utilizados en el juego 2D para acelerar, disparar y saltar al hiperespacio pero para girar la nave en el nuevo espacio se incorpora el movimiento del ratón. Moviendo el ratón hacia arriba y hacia abajo la nave se inclina alrededor de su eje X en la misma dirección y moviéndolo de izquierda a derecha la nave gira sobre su eje Y.

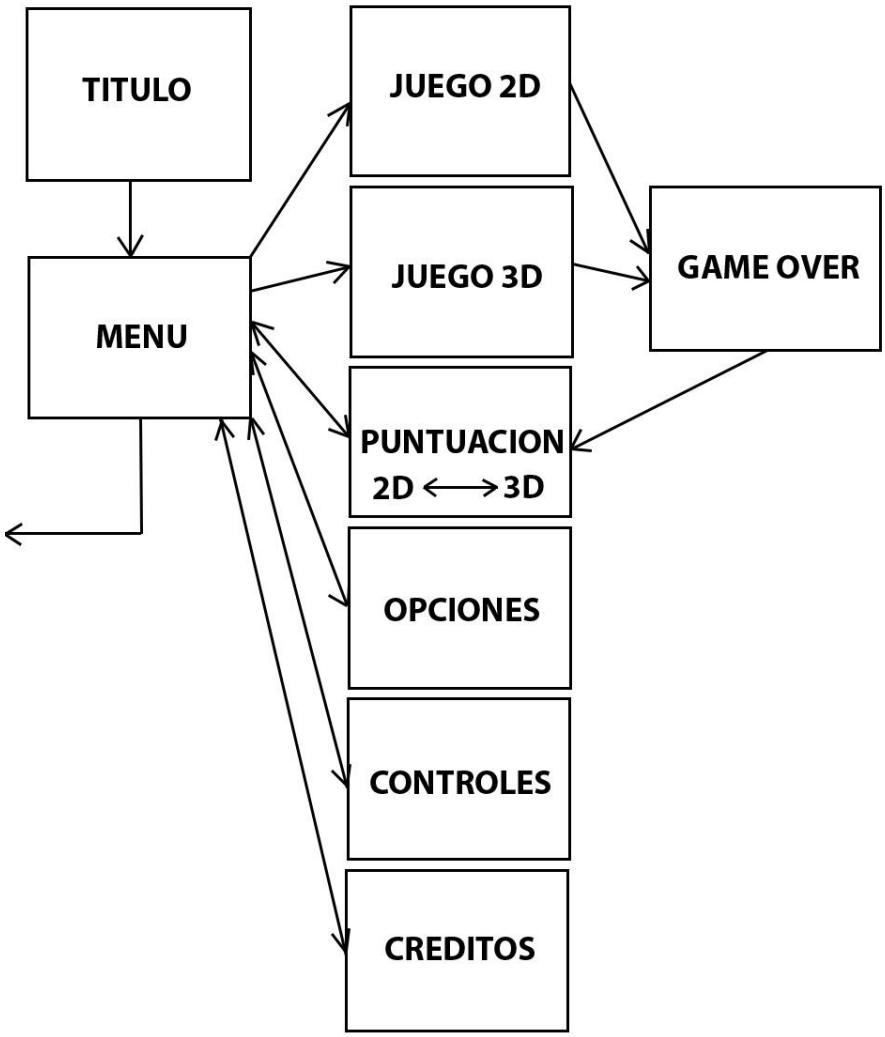
3.4 Control de la cámara

Se incluyen cinco tipos de cámara con los que jugar accesibles con las teclas de función:

- Cámara 3ª persona (F1): es la cámara por defecto. Permite ver el espacio siempre desde detrás de la nave girando solidaria con ella. Al implementarla como una cámara solidaria, la nave siempre aparece en la misma posición en pantalla, perdiendo ligeramente la sensación de movimiento. Para solucionarlo la cámara se aleja ligeramente de la nave cuando esta aumenta de velocidad, una técnica muy sencilla pero que logra el objetivo de darle dinamismo al movimiento.
- Cámara 1º persona (F2): permite ver desde la posición frontal de la nave. Con esta cámara se añade realismo al colocar al jugador en la posición del piloto.
- Cámara libre (F3): la cámara se queda fija en la posición actual cuando se accede a este modo. Manteniendo pulsada la tecla *Alt* es posible girarla con el ratón y avanzar con la tecla *W*.
- Cámara fija con seguimiento (F4): la cámara sigue a la nave sin moverse de su posición. Para cambiar su posición se puede pasar a la cámara libre y luego volver a este modo.
- Cámara *top-down* (F5): la cámara se mantiene por encima de la nave desde lejos. En este modo se simula la perspectiva del juego 2D manteniendo los controles y gráficos del juego 3D.

4.INTERFAZ

4.1 Diagrama de flujo de pantallas:

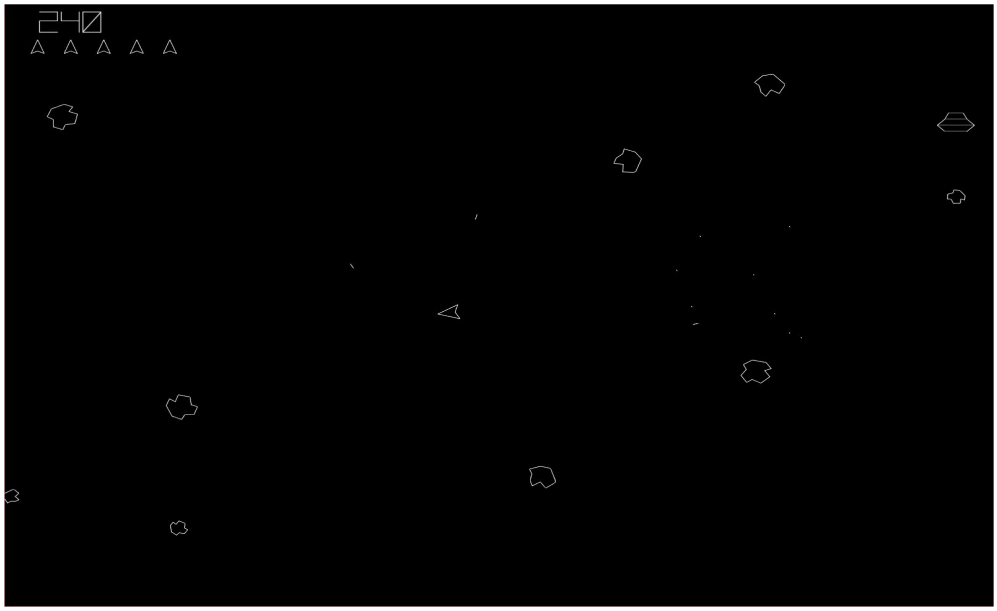


4.2 Título: Muestra una pequeña animación de como es el juego, se avanza al menú principal pulsando la tecla Enter.

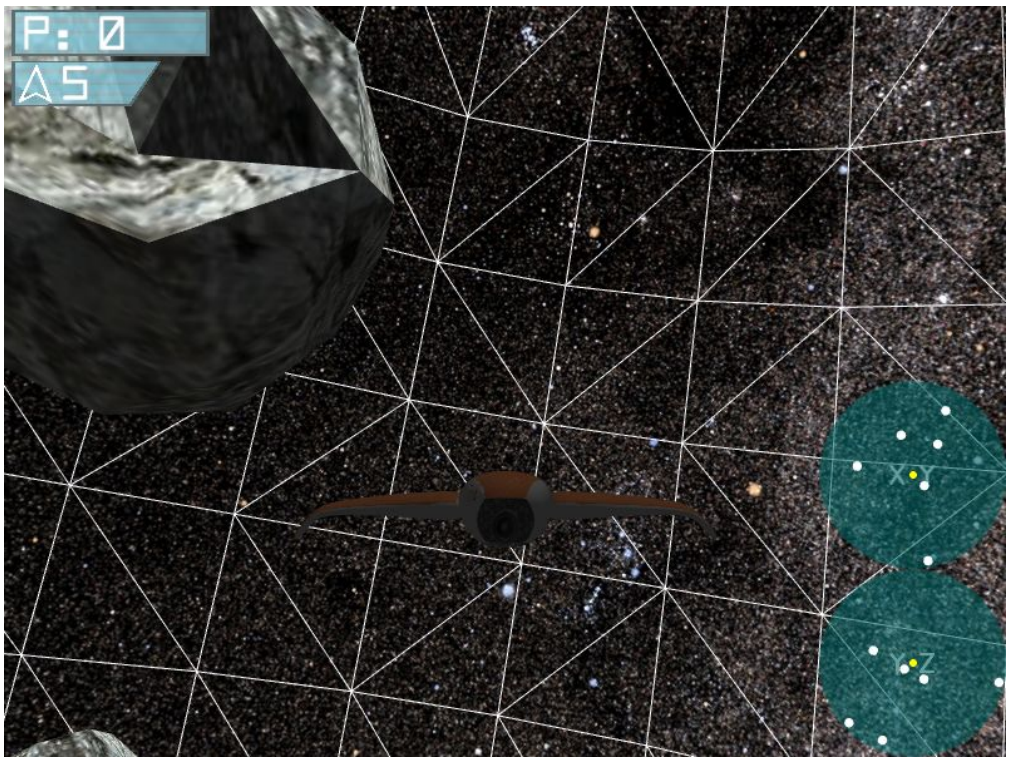
4.3 Menú principal: El menú principal del juego ofrecerá varias opciones básicas al usuario. Estas opciones serán: comenzar una nueva partida, observar las mejores puntuaciones, modificar algunas configuraciones del juego y salir de éste.

4.4 Partida 2D: En esta pantalla se verá tanto los elementos del juego como el número de reapariciones que posee el usuario, la puntuación y otros datos que se consideren de interés para el usuario. Tras acabar el juego, se avanza a la interfaz de fin de juego. En cualquier

momento, se puede pausar el juego pulsando la tecla Esc, desde la cual se puede terminar el juego.



4.5 Partida 3D: En esta pantalla se pueden ver los elementos antes nombrados y manejarlos o interactuar con ellos.

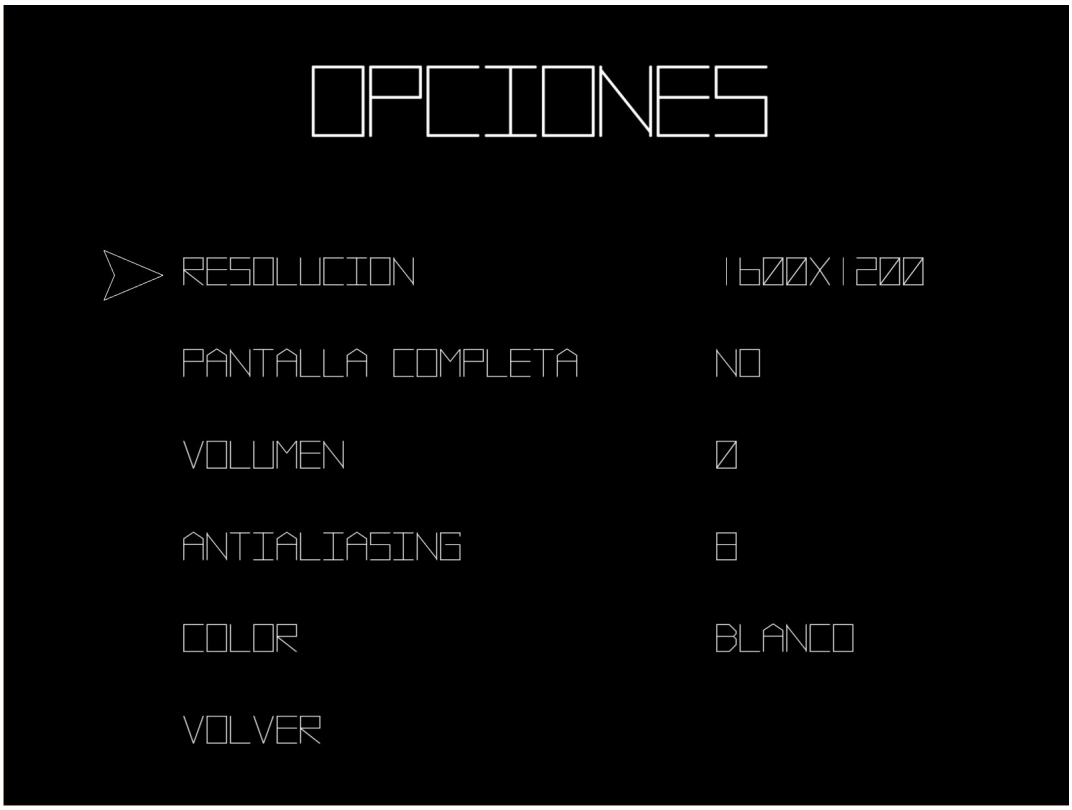


En la esquina superior izquierda se puede ver la puntuación actual y las vidas restantes. En la parte inferior derecha, se pueden ver dos radares con la posición global (respecto de la esfera espacial) de los asteroides, ovnis y nave.

4.6 Fin de partida: En esta pantalla se puede ver la puntuación final del jugador así como estadísticas obtenidas de su juego. El usuario podrá introducir un identificador que se asignará a su puntuación si esta es lo suficientemente alta como para estar entre las mejores. Tras confirmar, se avanza a la pantalla de puntuaciones.

4.7 Puntuaciones: Se mostrarán al usuario las mejores puntuaciones obtenidas en el juego y el identificador de los jugadores que las obtuvieron. Esta pantalla tiene dos opciones, mostrar las puntuaciones conseguidas en el modo 2D y las conseguidas en el modo 3D, para cambiar entre modos, hay que pulsar la tecla TAB. Al confirmar se vuelve al menú principal.

4.8 Opciones: El usuario puede cambiar ciertas configuraciones de juego para hacerlo más de su agrado. Para navegar por su menú y variar las configuraciones se usan las teclas de dirección y la tecla Intro para confirmar. Al terminar se volverá al menú principal.



4.9 Controles: El usuario puede consultar y modificar las teclas de acción del juego, para ello se desplaza hasta la acción que desee, pulsa Enter, después pulsa la tecla que desea usar para dicha acción, de nuevo confirma con Enter y al volver al menú principal, ya tendrá la nueva tecla asignada a la acción.

4.10 Créditos: Muestra los nombres y NIPs de los creadores. Se puede saltar pulsando ESC. Al terminar se volverá al menú principal.

5.ARTE

5.1 Estilo

El juego utiliza el modelo gráfico vectorial imitando al título original, un estilo artístico simple y reconocible. Esto quiere decir que todo elemento está realizado con puntos que se unen mediante líneas calculadas en tiempo de ejecución. Con ello, se logra una gran calidad aunque se lleven a cabo rotaciones y escalados.

5.2 2D

El juego utiliza una vista cenital, permitiendo ver la nave controlada por el jugador desde la parte superior de esta. Al utilizar este tipo de gráficos la nave únicamente se puede mover en un plano XY, ignorando la profundidad. Además, dicho plano, simula un espacio esférico, con lo que al atravesar uno de sus límites, la nave aparece por el límite opuesto.

La nave cuenta con diferentes estados y una apariencia diferente para cada uno de ellos, cuando acelera se ve el fuego del propulsor y al ser destruida explota y sus líneas se dispersan. Los asteroides cuentan con tres formas diferentes que se escalan según el tamaño de este, cuando son alcanzados por algún disparo o por alguna nave explotaran dividiéndose en dos y mostrando una nube de puntos. Los ovnis solo cuentan con un diseño que se escala de acuerdo al tipo al que pertenezca y al ser eliminados desaparecen, de nuevo, en una nube de puntos.

5.3 Música y FX

La melodía principal del juego utiliza dos tonos combinándolos a diferentes tiempos para dar sensación de urgencia según el jugador avanza por el nivel. Además, diferentes enemigos añaden otros sonidos para que el jugador se de cuenta de que una nueva amenaza ha aparecido en la pantalla.

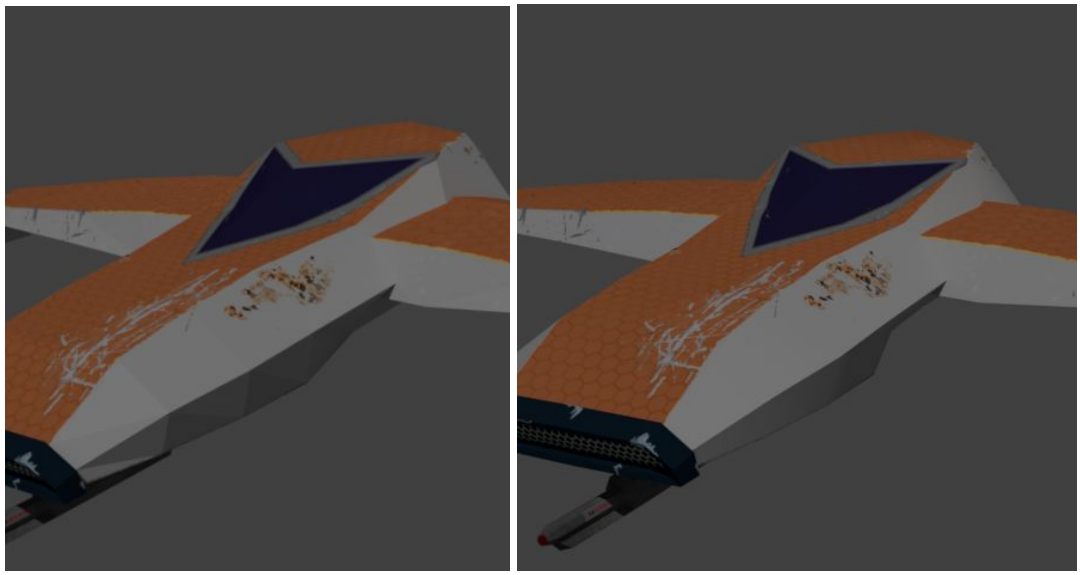
Por otra parte, se cuenta con efectos sonoros para las distintas acciones del jugador como pueden ser acelerar o disparar. También tiene efecto todo aquello que es destruido para aumentar la inmersión del jugador a pesar de que en el espacio los sonidos no se transmiten por el vacío.

5.4 3D

5.4.1 Modelos 3D

Se utilizan 5 modelos 3D diferentes creados por el encargado de arte utilizando el software de modelado 3D Blender. La nave y el ovni son los modelos 3D más “complejos” del videojuego y los que más trabajo han llevado. El disparo y los asteroides son pequeñas variaciones sobre formas geométricas sencillas.

Los modelos tienen pocos polígonos, y para evitar que se vean cortes duros en la iluminación se ha optado por modificar las normales de los modelos con objetivo de suavizarlos. En el siguiente ejemplo se puede apreciar la diferencia:



5.4.2 Texturas

Las texturas son imágenes planas que es posible “pegar” en modelos 3D para añadir detalles de forma muy simple. Las texturas de la nave y del ovni son de creación propia, intentando dar un aspecto realista. La textura de la cúpula celeste se ha obtenido de una fotografía 360º del espacio.

Además de para los modelos 3D, también se han utilizado texturas para la interfaz durante el juego, informando al jugador de sus puntos, vidas y de cuando reaparece tras morir.

5.4.3 Shaders

Los shaders son pequeños programas que ejecuta el procesador gráfico para determinar de qué color dibujar cada pixel de la ventana. En el juego 3D se utilizan dos shaders muy básicos. El primero simula una luz de área (en el infinito) con iluminación ambiente. Se utiliza en los modelos en los que la iluminación tiene más sentido, la nave, los ovnis y los asteroides. El segundo muestra los modelos ignorando la orientación de forma que no se ven iluminados. Se utiliza en los disparos de la nave y el ovni y en las esferas que rodean el espacio de juego.

6.IA

En este videojuego los elementos sobre los que se puede aplicar la Inteligencia Artificial son los ovnis enemigos cuya principal misión es destruir la nave del jugador. Las principales cualidades de los ovnis sobre los que se puede aplicar Inteligencia Artificial, son su capacidad de esquivar asteroides, que les permite aguantar durante más tiempo en la partida, y su puntería al disparar hacia la nave del jugador.

Debido a que hay dos tipos de ovnis diferenciados, cada uno debe contar con una IA distinta. El ovni grande disparara en cualquier dirección posible sin tener que apuntar en la dirección del jugador, además este ovni no siempre esquivara los asteroides y tendrá altas posibilidades de estrellarse. Por otro lado el ovni pequeño contará con una IA más avanzada, que además de esquivar con éxito a la mayoría de los asteroides apuntará directamente contra la nave del jugador.

Para implementar las redes neuronales se planteó el uso de varias librerías. Finalmente se utilizo la libreria neural, encontrada en github, que permitía la creación de redes neuronales con diferentes capas de forma sencilla.

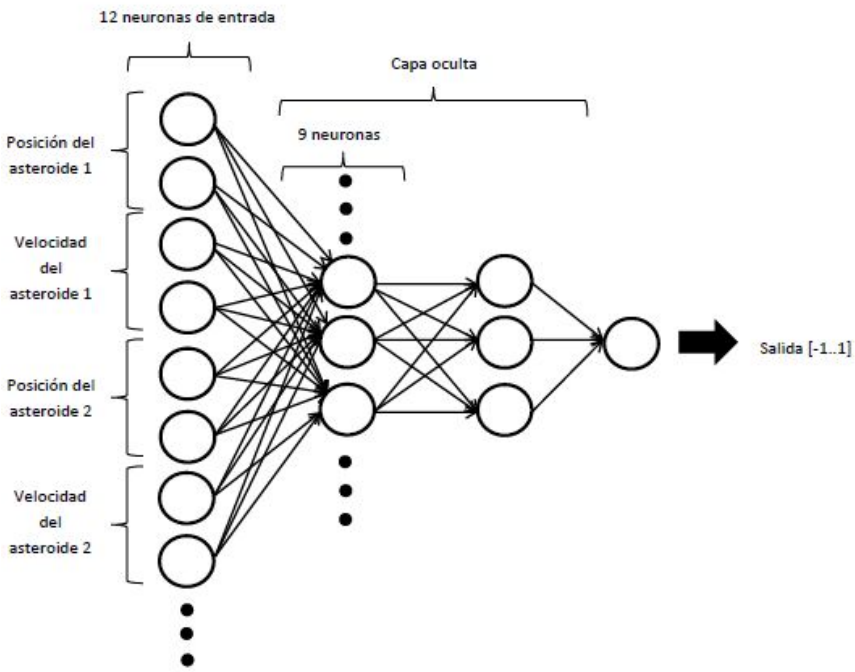
6.1 Red neuronal simple para el movimiento de los ovnis

En primer lugar, para aplicar esta IA, se probó con una red neuronal y aprendizaje supervisado. La red recibía como entrada la posición y velocidad de 3 asteroides, siempre que la nave chocaba contra uno, la red era entrenada introduciendo las direcciones que debía haber tomado. Para simplificar el aprendizaje, se limitó las posibles direcciones a 8.

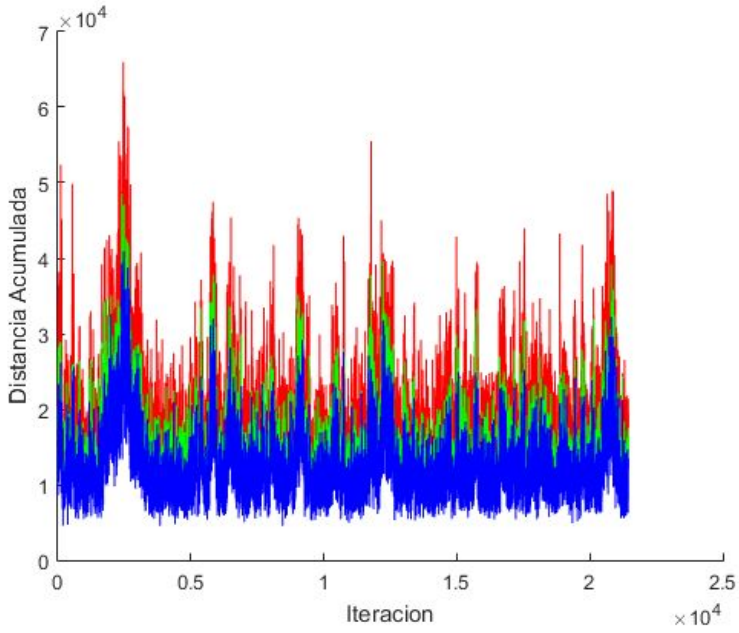
Éste red fue finalmente desechada debido a que apenas se producía ninguna mejora tras horas de entrenamiento, tras las cuales el ovni seguía chocando contra los asteroides de forma continua.

6.2 Red neuronal entrenada con algoritmos genéticos

Más adelante se probó con una red neuronal cuyos pesos se calculaban mediante un algoritmo genético. Dicho algoritmo mantenía una población de 8 individuos (pesos) en todo momento. Tras ejecutar diez simulaciones con cada uno de los individuos y medir la distancia recorrida por el ovni, se calculaban los individuos de la siguiente generación. Esto se hacía de la siguiente manera: se mantenían los tres que más distancia consiguieron recorrer, después, se generaban otros tres mediante combinaciones de dichos tres mejores. Por último, se completaba con dos individuos aleatorios. Con los mejores y sus combinaciones, se conseguía que el algoritmo se acercase progresivamente a los máximos de la distancia, mientras que los aleatorios se encargan de que no se atascase al encontrar un máximo local. La red neuronal utilizada tenía la siguiente forma:



Tiene 12 neuronas de entrada correspondientes a la posición (x e y) y la velocidad (x e y) de los tres asteroides más cercanos. Además tiene dos capas ocultas, la primera con 9 neuronas y la segunda con 3. Por último tiene una salida que se mapea en una dirección en la que se mueve el ovni. Tras un entrenamiento exhaustivo de muchas horas, se comprobó que no convergía lo suficientemente rápido, como puede verse en la siguiente imágen:

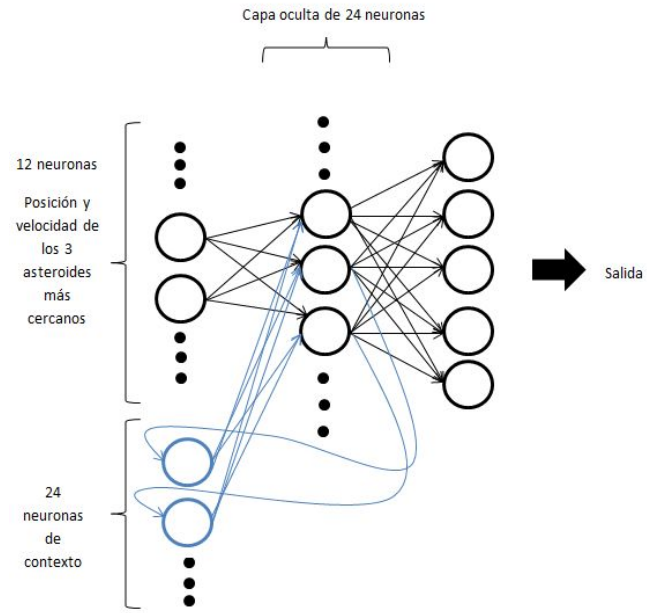


En la gráfica se pueden ver representados los tres mejores pesos (rojo el mejor, verde el segundo y azul el tercero) en cada iteración (eje X). El eje Y representa la distancia acumulada tras diez ejecuciones para cada conjunto de pesos. Como se observa, hay varios picos en la distancia, aunque ninguno con la suficiente altura como para funcionar bien en todos los posibles casos que pueden surgir durante una partida. Llevándolo al extremo, unos pesos óptimos lograrían una distancia infinita, de modo que 65.000 no es una marca muy notable.

El máximo obtenido durante varios entrenamientos fue 120.000. Se probó la red con los mejores pesos encontrados y se consideró que no era lo suficientemente buena esquivando. Con un tiempo de cálculo mucho mayor, se podría haber llegado a encontrar una solución válida que se podría utilizar en el ovni.

6.3 Red de Elman

Otro tipo de red que se implementó para que el ovni esquivase asteroides fue una red de Elman. Este tipo de redes se caracterizan por tener solo una capa oculta, cuya salida en cada iteración es enviada a la entrada de la red en la siguiente iteración. Las neuronas que reciben esta salida son llamadas neuronas de contexto. La red implementada fue muy similar a la usada en otros tipos de red con 12 entradas con la posición y velocidad de los asteroides, 24 neuronas en la capa oculta (y por tanto 24 neuronas de contexto) y 5 salidas, cada una de las cuales puede tomar los valores 0 y 1, dando un total de 32 direcciones posibles en las que podría moverse el ovni.

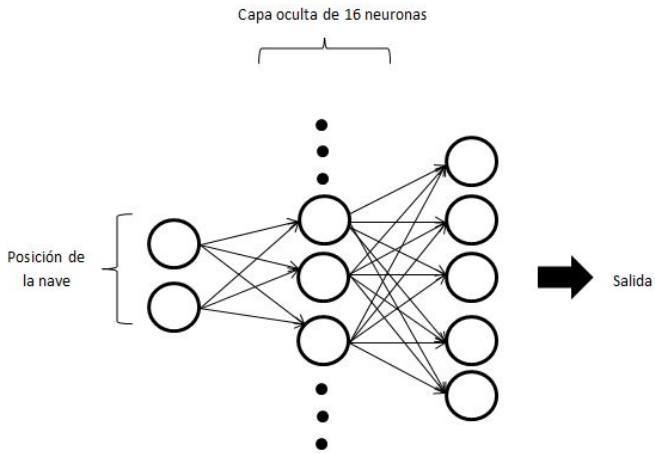


Los resultados obtenidos con esta red fueron muy similares a los anteriores con lo que también se desecho para ser utilizada en la versión final.

6.4 Red neuronal simple para los disparos de los ovnis

Para que la nave aprendiese a disparar a la nave del jugador, se diseñó una red neuronal para que dada la posición de la nave del jugador determinase hacia qué dirección debía disparar el ovni para poder destruir dicha nave.

En la imagen se puede observar el diseño de la red. Consta de dos entradas que indican la posición de la nave (relativa a la posición del ovni), una capa oculta de 16 neuronas y 5 salidas binarias que proporcionan un total de 32 direcciones posibles en las que realizar el disparo.

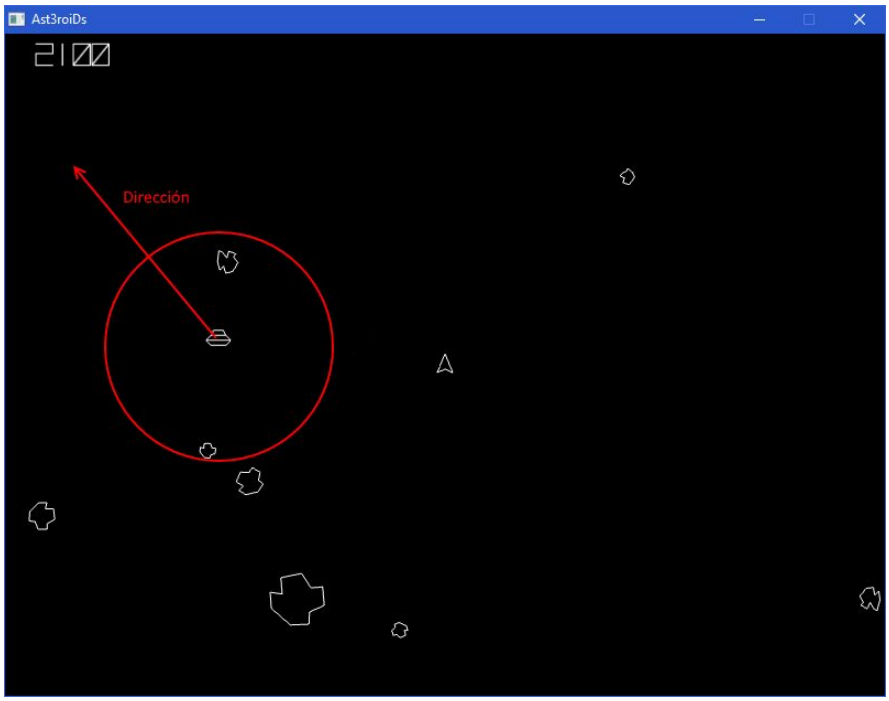


Para su entrenamiento se coloco a la nave y el ovni en una posición aleatoria del mapa y se ajustaban los pesos siempre que el disparo fallase. Los resultados obtenidos con esta red fueron bastante buenos por lo que es la red utilizada en la versión final del juego.

6.5 Inteligencia Artificial basada en reglas

Ante los pobres resultados que habían proporcionado las redes neuronales para esquivar asteroides, se decidió emplear un inteligencia artificial basada en reglas. Con este tipo de Inteligencia Artificial se han logrado cumplir los objetivos esperados obteniendo unos resultados aceptables a la hora de esquivar asteroides.

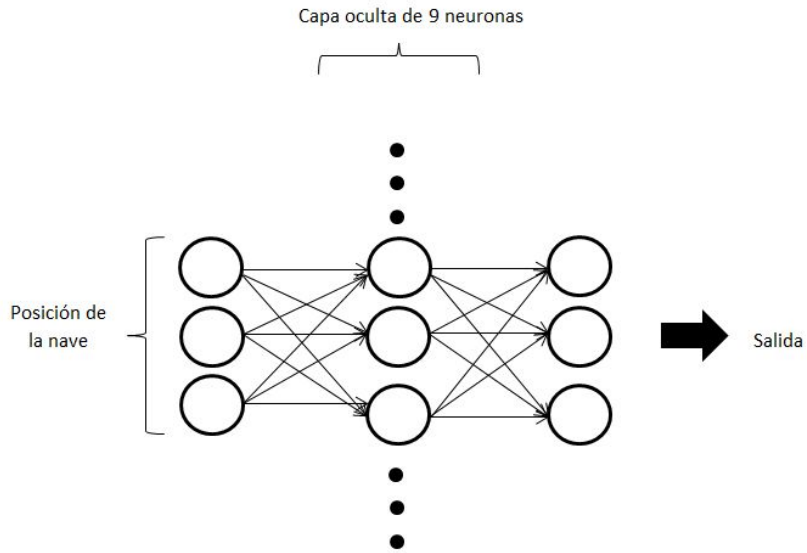
Para que el ovni esquivase los asteroides de la forma deseada, se han diseñado una serie de reglas consistentes en que cada vez que el ovni va a realizar un movimiento comprueba en qué posición se va a encontrar cada asteroide. Si algún asteroide se encuentra dentro de un determinado radio, el ovni tomará la dirección adecuada para poder esquivarlos a todos. El ovni además tiene preferencia por la dirección que tiene actualmente, variando así su trayectoria lo mínimo posible.



6.6 IA 3D

Para la versión 3D del videojuego se ha implementado, al igual que en la versión 2D, una inteligencia artificial que permite a los ovnis enemigos esquivar los asteroides y disparar a la nave controlada por el jugador. Para ello se han utilizado las mismas técnicas empleadas en la versión 2D.

Para los disparos del ovni se ha implementado una red neuronal. Esta red, al igual que la de la versión 2D, obtiene la posición de la nave como entrada y devuelve como salida la dirección en la que se debe disparar. La red cuenta con 9 neuronas en una capa oculta y una salida formada por 3 valores que representan los ejes XYZ.



Para que el ovni esquivase los asteroides se decidió utilizar el mismo algoritmo empleado en la versión 2D adaptado a las 3 dimensiones. Con este método se han obtenido unos resultados similares a los de la versión 2D, en la que el ovni inteligente solo se chocara si está completamente rodeado por los asteroides.

7. TECNOLOGÍAS Y HERRAMIENTAS DE DESARROLLO

7.1 Lenguaje de programación

Se decide realizar el proyecto enteramente en C++. La decisión se debe a que es uno de los lenguajes de programación más utilizado en proyectos de este estilo. Debido a la escasa experiencia con este tipo de proyectos, existía el riesgo de no conseguir una solución lo suficientemente eficiente para cumplir exigencias básicas para poder disfrutar de una experiencia de juego satisfactoria con lenguajes de programación de más alto nivel. Además, en las últimas etapas de la carrera se ha utilizado C++ bajo recomendación del profesorado para conseguir una eficiencia mayor al ejecutar programas con cálculos de cómputo intenso.

Aunque el estándar en entornos Windows es utilizar Visual Studio, preferimos utilizar el entorno CLion de JetBrains y MinGW para compilar el código ya que es más ligero y tiene herramientas similares a otras utilizadas en la carrera como make y gcc.

7.2 Librerías

Otro motivo igualmente importante es la cantidad de librerías de juegos disponibles para C++. Son una parte fundamental del desarrollo (casi) todo videojuego al permitir

mostrar gráficos por pantalla. Incluso crear una simple ventana no es un problema trivial si no se utilizan librerías externas con este tipo de funcionalidades.

Tras investigar varias librerías de juegos, se decide utilizar SFML porque unifica en una sola librería la posibilidad de crear ventanas, dibujar gráficos 2D y 3D (con OpenGL), reproducir sonidos, recibir inputs del usuario y realizar operaciones matemáticas sencillas con datos vectoriales.

Para crear el juego en su versión 3D, se ha tenido que usar OpenGL y GLEW que tras algunos días de trabajo, se ha integrado con las demás librerías anteriores. Además para manejo de matrices y otros cálculos, se ha utilizado la librería glm.

Para crear redes neuronales se optó por buscar una librería con una interfaz sencilla con la que poder crear y entrenar redes neuronales con tamaños fijos. La mejor candidata fue neural ya que permite crear, entrenar, guardar y cargar redes neuronales de forma muy sencilla. Con unas pequeñas modificaciones, como añadir un acceso directo a los pesos de las neuronas, fue posible utilizar algoritmos genéticos para entrenar redes neuronales con la misma librería.

7.3 Control de versiones

El control de versiones del proyecto se ha realizado con un repositorio Git almacenado en GitHub. Ha demostrado ser de gran utilidad para gestionar el trabajo simultáneo sobre diferentes secciones del videojuego. Además, revisando el repositorio podemos ver qué cambios se lograron en cada momento de la vida del proyecto desde sus inicios.

7.4 Modelado 3D

Todos los modelos 3D se han realizado mediante el programa Blender, se han texturizado mediante la técnica de UV mapping, creando los mapas difusos con Photoshop. Todos los modelos son de bajo poligonaje, y para disimular los polígonos se han modificado desde Blender las normales de los objetos.

8. Problemas encontrados

A lo largo del desarrollo se han encontrado multitud de problemas de mayor y menor magnitud. Éstos son los más importantes.

8.1 Nuevo tipo de proyecto

En la carrera de ingeniería informática nunca ha sido necesario tratar con muchos de los problemas inherentes al desarrollo de un videojuego.

8.1.1 Diseño adaptativo a los problemas que van surgiendo

El primer problema, y tal vez el más importante, es el de enfrentarse al diseño de un programa diferente a los vistos en la carrera. Los diseños iniciales fueron inapropiados para la librería y las necesidades del programa y se modificaron constantemente conforme se avanzó el desarrollo. Es indudable que el código final no posee cualidades deseables de la programación orientada a objetos ni ningún paradigma. La multitud de problemas que surgen, unida a la falta de tiempo, complican el mantenimiento de un código limpio, claro y bien estructurado. En retrospectiva, podría realizarse un mejor trabajo de ingeniería del software pero como una primera aproximación a la programación de videojuegos el equipo queda satisfecho con el trabajo realizado.

8.1.2 Tiempo real

Un videojuego como AsteroiDs tiene unos requisitos básicos para suponer una experiencia de juego agradable.

Al tener movimientos fluidos y requerir de precisión al apuntar a los asteroides, es importante mantener el tiempo de cálculo de cada fotogramas lo más

bajo posible. Si no fuera posible mostrar más de 30 fotogramas por segundo la experiencia en un monitor de ordenador convencional no sería óptima e incluso podría resultar molesto para algunas personas.

En verdad esto no ha llegado a ser nunca un problema puesto que el juego es muy sencillo y cada fotograma tarda decenas de milisegundos en calcularse en un computador moderno.

8.1.3 Pruebas con aleatoriedad

La mayoría de elementos con los que se interacciona en el videojuego son aleatorios. Esto presenta un reto para encontrar errores de programación dentro del juego. Por ejemplo, cuando la nave reaparece está forzada a esperar a que no haya ningún asteroide en la zona central de la pantalla ya que no sería justo para el jugador que fuera destruida nada más aparecer. Esto implica realizar varias pruebas hasta que se dé la situación en la que la nave pueda aparecer en el momento en el que aparecen asteroides por el centro de la pantalla.

8.1.4 Compilación de librerías para un entorno inusual

Los binarios de la librería SFML no están disponibles para la versión de MinGW 64 que se ha utilizado en este proyecto, por lo que resultó necesario compilarlo manualmente para obtener los ficheros .dll necesarios para ejecutar el videojuego.

8.2 Inteligencia Artificial

En lo referente a la Inteligencia Artificial uno de los primeros problemas a resolver fue cómo implementar las redes neuronales que se planeaban utilizar ya que implementar todos los métodos necesarios para crearlas de cero era demasiado costoso. Por ello se decidió utilizar una librería obtenida en GitHub que tenía todo lo necesario para la red neuronal que se tenía en mente.

8.3 Ajustes de escala

Otro de los grandes problemas encontrados ha sido la multitud de variantes de resolución que los distintos equipos pueden permitir. En primera instancia, se fijó una resolución de 800x600 y se hicieron todos los ajustes de velocidad, tamaño y velocidad de los elementos. Después se realizó un método que ajustaba dichos valores según la relación de las medidas de la pantalla. Cómo estos ajustes no funcionaban correctamente en resoluciones con una proporción distinta de 4:3, se permitió al usuario que cambiase la resolución a cualquiera que su ordenador soportase siempre que fuese 4:3. Finalmente, se ha añadido la posibilidad de usar la pantalla completa, pero como una pantalla 4:3 y bandas negras en los laterales.

8.4 Coordenadas de textura

Un gran problema que surgió durante el desarrollo en 3D fue la aplicación de las texturas a los modelos en el videojuego. Aunque en el programa de edición el resultado tenía el aspecto deseado, en nuestro espacio 3D se veían errores en las costuras por las que se había dividido la malla a la hora de aplicarle la textura.

El problema residía en que algunos vértices tienen múltiples coordenadas del mapa UV pero solo llegaba a la GPU una coordenada de textura por vértice.

Para solucionarlo, se modificó la forma en la que se cargan los modelos 3D de los ficheros wavefront (*.obj*) para que cada combinación de vértice, normal y coordenada de textura sea única y por tanto se dibuje con los valores correctos. Esto tiene el efecto adverso de ocupar más memoria en la GPU pero como los modelos son muy sencillos y tan solo hay 5 diferentes el problema es mínimo.

8.5 Movimiento libre de la nave

La rotación de cualquier elemento en un espacio 3D puede dar lugar a problemas dependiendo de la codificación que se utilice. En un principio, la codificación de las rotaciones de cada elemento utilizaban ángulos de Euler, es decir, una tripla de valores indicando la rotación alrededor de cada eje. Esta forma de representar rotaciones tiene el problema del *gimbal lock*, que impide rotar un elemento en algún eje si alguno de ellos se ha alineado.

La solución a este problema se ha encontrado con el uso de cuaterniones, una extensión de los números reales que permite representar cualquier rotación evitando el problema del *gimbal lock.* Intuitivamente, un cuaternión representa un vector y la rotación que se realiza alrededor del mismo. Es una forma diferente de representar rotaciones menos intuitiva pero igualmente válida.

9. REPARTO DE TAREAS Y CRONOGRAMA

A pesar de tener asignados unos roles, todos los integrantes del grupo hemos trabajado en todas y cada una de las partes del videojuego, ya sea aportando ideas, llevándolas a cabo o probando que todo funciona correctamente. No hemos llevado un recuento de horas invertidas de forma individual ya que eso implica un trabajo extra, pero si uno de horas invertidas globalmente. Dicho recuento, hasta la fecha de 2/5/2017 es de unas 420 horas.

En el siguiente cronograma, se pueden ver las tareas relativas al proyecto realizadas cada semana:

TIPO	TAREA	1 (13F-19F)	2 (20F-26F)	3 (27F-5M)	4 (6M-12M)	5 (13M-19M)	6 (20M-26M)	7 (27M-2A)	8(3A-9A)	9(10A-16A)	10(17A-23A)	11(24A-30A)	12(1My-7My)
DISEÑO	Extracción de características principales del videojuego original (diseño de niveles, movimientos, elementos del juego...)												
	Boceto en alto nivel del videojuego a desarrollar.												
	Elección de una biblioteca para manipular elementos gráficos, crear ventanas y reproducir sonidos.												
	Diseño completo del videojuego a desarrollar (2D), así como mapa de navegación y lista de características.												
	Modificación y/o ampliación del diseño para cubrir otras características deseables.												
	Diseño videojuego 3D												

ARTE	Extracción de las características gráficas del videojuego original (gráficos vectoriales, primeros bocetos...)												
	Obtención de los sonidos originales del videojuego.												
	Lectura de tutoriales y documentación de la biblioteca gráfica sobre como realizar gráficos vectoriales, música...												
	Dibujo, geometricamente, de la nave y un tipo de asteroide												
	2 diseños nuevos de asteroides, nuevo diseño de la nave y primer dibujo de un ovni.												
	Nueva forma de definir las lineas que forma cada objeto y dibujos finales de los mismos.												
	Primera versión estética del título y menú principal.												
	Versión final del título y menú, primera versión de las puntuaciones, game over y controles.												
	Versión final de todas las pantallas y efectos en los créditos.												
	Creación modelos 3D												
	Creación de UI												

IMPLEMENTACIÓN	Implementación del mapa de navegación de las partes del videojuego (titulo, menu, juego, puntuaciones...)												
	Primera implementación de los asteroides que se mueven por la pantalla.												
	Primera implementación de un triangulo que se mueve, rota y dispara en la pantalla de juego.												
	Primeras colisiones con circulos.												
	Añadidas más características a la nave y asteroides. Primera implementación de los ovnis.												
	Nuevos métodos de colisión para ajustar mas a la forma de los objetos.												
	Sistema de puntuaciones y vidas.												
	Sistema de opciones y cambio de atributos para adecuarse a la resolución. Destrucción según colisiones.												
	Cambio de controles y creditos.												
	Inclusión de bibliotecas: glew, glm y tinyobjloader												

	Creación de las clases y elementos necesarios para mostrar modelos 3D												
	Jugabilidad 3D												
	Rotaciones con cuaterniones												
	Añadir detalles obtenidos de las críticas de los compañeros												
IA	Extracción de las características de los NPC del videojuego original.												
	Inclusión de biblioteca para redes neuronales.												
	Entrenamiento del ovni mediante redes neuronales para esquivar asteroides.												
	Uso de redes neuronales con algoritmos geneticos.												
	Implementación de IA basada en reglas.												
	Implementación y entrenamiento de red neuronal de disparos												
	Implementacion de IA para el videojuego 3D												
PRUEBAS	Pruebas del mapa de navegación.												
	Pruebas de la nave (aceleración, deriva, disparo, rotación...)												
	Pruebas de colisiones básicas y de los ovnis.												

	Pruebas con redes neuronales												
	Prueba de las nuevas colisiones												
	Pruebas generales y finales del videojuego y corrección de errores.												
	Pruebas del videojuego 3D												
DOCUMENTACIÓN	Anotación de las características extraídas.												
	Añadidas nuevas características.												
	Conversión del documento de características al GDD y añadidas nuevas secciónes sugeridas.												
	Finalización del GDD y entrega del mismo.												
	Corrección del GDD.												
	Realización de la memoria del videojuego 2D+IA.												
	Ampliación de la memoria para incluir mejoras al juego 2D												
	Memoria final videojuego 2D+IA+3D												