POLITÉCNICA

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES
UNIVERSIDAD POLITÉCNICA DE MADRID

José Gutiérrez Abascal, 2. 28006 Madrid
Tel.: 91 336 3060
info.industriales@upm.es

www.industriales.upm.es

Javier Díaz Camacho

05 TRABAJO FIN DE GRADO

INDUSTRIALES

TRABAJO FIN DE GRADO

ESTRATEGIAS DE ATAQUE Y DEFENSA DE ENJAMBRES DE DRONES HETEROGÉNEOS

TRABAJO FIN DE GRADO PARA LA OBTENCIÓN DEL TÍTULO DE GRADUADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

SEPTIEMBRE **2018**

Javier Díaz Camacho

DIRECTOR DEL TRABAJO FIN DE GRADO:
Antonio Barrientos

INDUSTRIALES
ETSII | UPM

POLITÉCNICA

Agradecimientos

En primer lugar, me gustaría agradecerle a Antonio Barrientos darme la oportunidad de realizar este trabajo, y especialmente a Pablo García por toda la ayuda que me ha ofrecido a lo largo de la realización del proyecto. A los tutores del grupo Swarm City, agradecer su buena disposición para ayudarme en cualquier duda, aunque no fuese su proyecto. A los profesores de la especialidad, por despertarme inquietudes en ciertos temas de automática y electrónica, y por todo lo que me han enseñado durante estos años.

Por último, a mi familia, amigos y a ti Aída por vuestra ayuda, paciencia y tiempo invertido.

Resumen

La investigación y desarrollo de vehículos aéreos no tripulados (drones) ha tenido un gran auge en las últimas décadas y actualmente se ha demostrado que el empleo de enjambres de drones presenta mayores ventajas en multitud de ámbitos.

El primer objetivo de este proyecto ha sido el modelaje y manejo de dos inteligencias de enjambres de drones heterogéneos controladas por una red comportamientos de tipo iB2C en Python. Para ello se enfrentarán estas dos inteligencias entre sí en dos escenarios diferentes, para determinar cuál es la más eficiente. La primera inteligencia, llamada individualista, se basa en que los drones no tienen un objetivo predeterminado, sino que en función de los datos que van recibiendo por el radar, realizan la acción más eficiente, mientras que la segunda, denominada por roles, tiene las tareas a realizar definidas por su rol. El primer escenario es un campo 3D donde nos encontramos las bases de los dos equipos, obstáculos a esquivar y 3 zonas de recursos a recolectar. Por último, el segundo escenario tiene los mismos elementos que el primero, pero añade una zona central, llamada colina, en la que los equipos deberán tener más drones que el equipo contrario. La manera de comprobar que inteligencia es más eficiente será mediante un sistema de puntos que se conseguirán realizando 3 acciones diferentes. Dichas acciones son llegar a la zona de recursos y volver a la base sin ser abatidos, disparar y acertar con proyectiles a los drones del equipo rival y mantener mayoría de drones en la colina. El primer escenario cuenta con la posibilidad de obtener puntos solo en las dos primeras tareas, mientras que en el segundo escenario se puede puntuar de las tres formas.

El segundo objetivo consistió en la optimización de dichas inteligencias mediante un gradiente descenso, ya que los valores con los que empezaremos a realizar las simulaciones son aleatorios. Este método consiste en realizar simulaciones modificando las variables dentro de unos márgenes hasta que se encuentren una combinación de parámetros óptimos. Dicha combinación puede dar lugar a dos situaciones, que se supere a la otra inteligencia o que se obtenga los parámetros que dan lugar a la mayor puntuación. Tras la optimización podremos afirmar que los resultados obtenidos no son fruto del azar y serán válidos. Además, realizaremos paquetes de 10 simulaciones para comprobar que las puntuaciones no están dispersas.

Al realizar las primeras simulaciones en el primer escenario, el equipo 1 ganó, por lo que optimizamos el otro equipo. Tras el proceso de optimización, el equipo 2 no consiguió ninguna combinación de parámetros con los que ganar al equipo 1. En el escenario 2, antes de las optimizaciones, el equipo 1 también ganó, pero al optimizar el equipo 2, este sí consiguió ganar. El siguiente paso fue optimizar el equipo 1. Tras el proceso de optimización, volvió a ganar el equipo 1. Por último, al hacer la segunda optimización del equipo 2, este no consiguió superar al otro equipo.

De esta forma podemos concluir que, la inteligencia que se adapta a la información de sus sensores para determinar su próximo objetivo, es más eficiente que la inteligencia cuyos roles están ya asignados.

Es destacable el hecho de que en el primer escenario la diferencia de puntos obtenidos es bastante mayor que en el segundo. Esto se debe a que, al no tener en cuenta el objetivo de mantener la mayoría de drones en la colina, las posibilidades de optimización del segundo equipo son menores y tiene menos flexibilidad, dejando al equipo con la inteligencia individualista con un mejor rendimiento.

Palabras clave

UAV, dron, enjambre, control basado en comportamientos, iB2C, Python, optimización, gradiente de descenso

Código UNESCO

331101 Tecnología de la Automatización

331102 Ingeniería de Control

120302 Lenguajes algorítmicos

120304 Inteligencia artificial

120326 Simulación

330710 Radar

ÍNDICE

1. INTRODUCCIÓN

1.1. Antecedentes

En los últimos años los UAV (*Unmanned Aerial Vehicle*) o drones han dejado de ser herramientas exclusivas en el ámbito militar y han pasado a ser un aspecto más de la vida cotidiana [1]. Estas aeronaves tienen una utilidad inmensa, desde tareas de mantenimiento o exploración en zonas peligrosas hasta el ocio. Todo esto se debe a que son autónomos, es decir, que tienen la capacidad de autogestión y no están pilotados por personas, y que además son manejables y seguros. Pero un solo dron tiene varias limitaciones, como que solo puede realizar una tarea cada vez y el limitado tiempo de vuelo o capacidad de carga.

Para mejorar estos aspectos, los enjambres de robots autónomos o drones están en auge, ya que incrementa enormemente el amplio abanico de usos que tiene un dron. El beneficio del uso de enjambres reside tanto en la adaptación a cambios inesperados en el ambiente, como en la gestión de varias actividades simultaneas o en la eliminación de las limitaciones que tiene cada dron por separado. Un ejemplo de una tarea de estos enjambres es el levantamiento topográfico de zonas extensas de terreno. Esta tarea sería impensable que la hiciese un solo dron, debido al tiempo que tardaría en completarla, sin embargo un enjambre tardaría mucho menos en llevarla a cabo.

A la hora de controlar tanto un único dron como un enjambre nos encontramos con varias opciones perfectamente aceptables, pero de todas estas, resalta el control basado en comportamientos, y en especial, el iB2C [7]. Se basa en una red de comportamientos simples que se estimulan e inhiben entre ellos para dar lugar a una inteligencia compleja, flexible y robusta. Gracias a estas interacciones entre los diferentes comportamientos, se consigue que las instrucciones que reciben los actuadores (motores y válvulas) sean las más adecuadas a la información que reciben los sensores.

Por último, este trabajo ha tenido lugar en la Escuela Técnica Superior de Ingenieros Industriales de la Politécnica de Madrid dentro del departamento de Automática, Electrónica e Informática Industrial durante el curso 2017/2018. Asimismo, el presente trabajo fin de grado se engloba dentro del grupo Swarm City. Dicho proyecto tiene como objetivo el estudio de enjambres de drones para diferentes tareas, que van desde el reconocimiento y búsqueda en zonas urbanas hasta la creación de un equipo de fútbol que podría competir en la RoboCup, torneo internacional donde se reúnen multitud de equipos de fútbol formados por robots.

1.2. Objetivos

El objetivo principal de este trabajo es determinar si en enjambres de drones heterogéneos una inteligencia individualista es más eficiente que una inteligencia por roles. Para ello, nos propusimos tres acciones que los enjambres deberían completar:

- Recolectar recursos: los drones han de ir a varios puntos donde se aprovisionan y volver a la base sin ser derribados.
- Abatir drones enemigos: los drones tienen la capacidad de disparar proyectiles que deben impactar en los drones del equipo rival.
- Mantener posición: consiste en tener más drones que el equipo rival en un área acotada y fija.

Se integrarán las dos inteligencias en dos escenarios diferentes. En el primero solo se valorarán los dos primeros objetivos, mientras que en el segundo los dos equipos deberán competir en obtener la mayor puntuación en las 3 tareas. Usaremos dos escenarios ya que queremos comprobar si la inteligencia que ha resultado más eficiente en el primero, lo va a ser también en el otro. De esta forma podríamos concluir si es la mejor entre las dos.

Por último, se intentarán optimizar las inteligencias y se comparará si dicha evolución ha sido eficiente o no.

2. ESTADO DEL ARTE

2.1. Dron

En el siglo XX, el término UAV, vehículo aéreo no tripulado o dron ha estado relacionado con el ámbito militar [1] porque los drones suponen una ventaja significativa en el campo de batalla, ya que ofrecen la posibilidad de realizar ataques sin poner en riesgo vidas humanas. Pero debido al avance tecnológico y a la reducción de su precio, los UAVs han empezado a ser utilizados por grupos de aficionados al aeromodelismo. De acuerdo a un informe de Tractica [2], se espera que las compras de drones sigan creciendo exponencialmente en los próximos años. El total de unidades vendidas en el mundo se estima que se incrementará de 6,4 millones en 2015 a un total de 67,9 millones en 2021 [2], lo que provocará una bajada en los precios, favoreciendo que esta tecnología sea asequible para todo el mundo [8]. El potencial de los UAVs va más allá del ámbito del entretenimiento, y sus aplicaciones están aumentando en gran medida. Hoy en día, es muy común el uso de drones con cámaras para hacer fotografías sobre centros urbanos. También es usual verlos ayudando a grabar películas o en industrias como petróleo o gas para realizar misiones de mantenimiento en zonas de difícil acceso [8].

La industria está actualmente floreciendo con múltiples diseños de UAV, donde cada uno tiene sus propias fortalezas y debilidades. Los factores más importantes son el precio, el tiempo de vuelo, la seguridad, el despegue y aterrizaje y la capacidad de carga.

Los dos tipos de drones que definiremos en este proyecto van a ser el cuadricóptero y el ala fija.

Los cuadricópteros son los más comunes en el mercado debido a su bajo precio en comparación con los otros modelos. Estos drones son perfectos para llevar pequeñas cámaras en el aire durante un tiempo de vuelo reducido. Y debido a sus 4 motores, el piloto tiene un gran control de la aeronave, además, gracias a su estabilidad y maniobrabilidad, lo puede pilotar con facilidad gente sin experiencia [8].

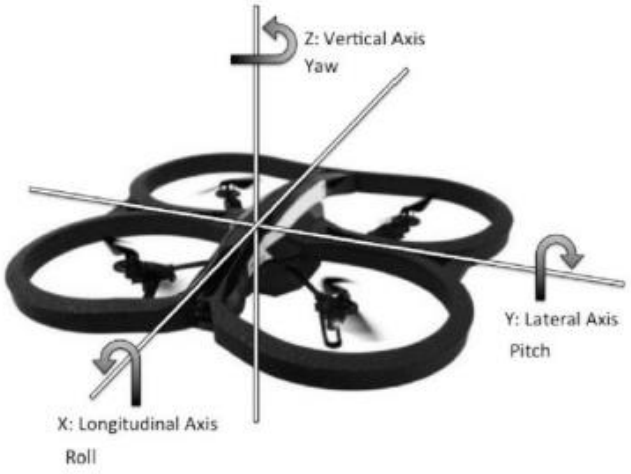

Figura 2.1. Cuadricóptero

En cambio, los drones de ala fija son más parecidos a una aeronave convencional. Debido a que las alas son las que se encargan de mantenerlo en el aire, este diseño permite que los motores ejerzan más fuerza de empuje y se consiga más potencia que los cuadricópteros. También permite el uso de motores de combustión, con una menor relación peso-tiempo de

vuelo que los motores eléctricos. Algunos drones de ala fija pueden volar durante periodos de tiempo prolongados, siendo ideales para trazar mapas de áreas extensas [8].

Figura 2.2. Dron de ala fija

2.2. Enjambre de robots

El término robot celular fue acuñado por Toshio Fukuda en Japón en los años 80 para referirse a grupos de robots que se asemejaban a las células y que, juntos, creaban partes más complejas. Pero el tema perdió el interés hasta que en el 2002 se publicó el libro "*A New Kind of Science*" de Wolfram Science [4]. Enjambres, el término que se usa actualmente para definir a estos robots, fue utilizado por primera vez por Gerardo Beni [3], dado la similitud que tienen con los enjambres de insectos en la naturaleza.

Las características que los definen son [6]:

-Son robustos: el enjambre debe de ser capaz de continuar con su trabajo, aunque fallen algunos de sus integrantes. Esto se debe a la redundancia del sistema, ya que, al estar formado por multitud de unidades muy simples, la tarea que realizaba un robot que deja de funcionar, puede ser cubierta por otro. Además, como el diseño de cada individuo es muy básico, tiene menos probabilidad de que cada unidad falle. Y, por último, al tener multiplicidad de señales recibidas por los sensores, el error que podría tener un sensor individual se reduce.

-Son flexibles: es decir, que son capaces de realizar un gran número de tareas y se adaptan mediante estrategias coordinadas a cambios inesperados en el ambiente.

-Son escalables: los enjambres tienen la capacidad de funcionar con un amplio rango de número de robots, o lo que es similar, que la funcionalidad de grupo no se verá alterada por un cambio en la cantidad de individuos que lo forman.

Un enjambre de drones está influenciado por 3 fuerzas básicas: separación, alineamiento y cohesión. La fuerza de separación provoca que los drones se alejen individualmente de cada vecino, para evitar colisiones. El alineamiento hace que cada uno de los individuos se muevan con la misma dirección y sentido que el grupo. Y la fuerza de cohesión les obliga a acercarse al centro de masas del grupo [9].

El modelo más usado para el control de la velocidad de los drones es un modelo de inercia simple con una k y tau. La k es un parámetro que aumenta o disminuye, de forma directa, la

velocidad máxima alcanzada por el dron, mientras que la tau indica lo rápido o lento que se va a llegar dicha la velocidad máxima [5].

$$Vel = k * vel\_max * \left(1 - e^{-t/\tau}\right) + Vel * \ e^{-t/\tau}$$

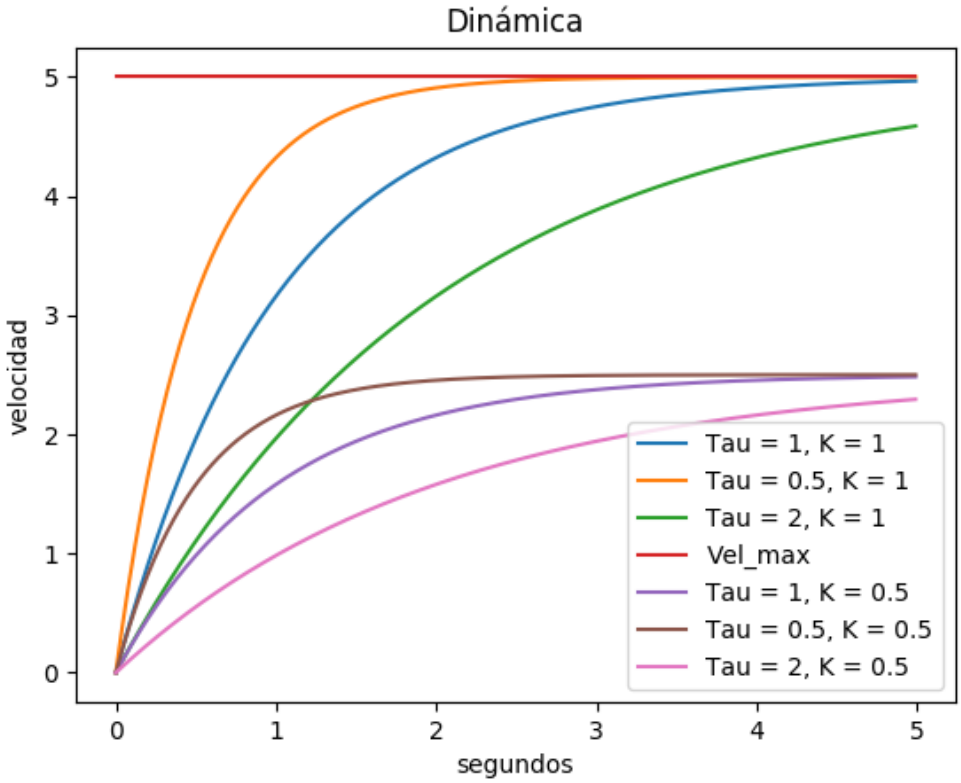

Figura 2.3. Modelo dinámico

2.3. Control basado en comportamientos

Crear una inteligencia con la que se va a controlar los robots es uno de los problemas más complejos cuando se desarrolla el sistema de mando para cualquier tipo de aplicación. La complejidad de esas tareas depende de las características del robot, tales como el número de sensores y actuadores (motores, válvulas o cilindros), del entorno y como se relacionan entre sí.

El proceso de desarrollar este complejo sistema de control debería ser soportado por una metodología adecuada para ayudar a superar dichas dificultades. Hay varias formas de controlar este proceso. Las más importantes son el control reactivo, de rápida respuesta pero poca eficiencia y el control deliberativo, de algoritmos más complejos y soluciones más óptimas.

La inteligencia basada en comportamientos pertenece al primer grupo, y ha manifestado que es capaz de superar problemas como la dependencia de las órdenes de un núcleo central, demostrando que no disminuye su funcionalidad frente a tareas que van incrementando en complejidad, y muestran robustez frente a información desconocida captada por los radares.

Aunque la inteligencia basada en comportamientos solventa algunos problemas, crea otros nuevos, como la coordinación de diferentes conductas que intentan controlar el mismo actuador a la vez. La inteligencia iB2C [7] solventa este problema, seleccionando el mejor comportamiento y coordinando sus acciones. Se basa en la creación de un modelo de red de comportamientos estandarizado, formado por diferentes módulos simples. La interfaz permite la reutilización y la adición de dichos módulos de una manera sencilla, y el mecanismo de coordinación del comportamiento hace posible la separación de los datos de control con los datos de flujo.

Se ha demostrado que la arquitectura propuesta es suficientemente potente para soportar la implementación de mecanismos clave de otros modelos basados en comportamientos. Además, los módulos implementados son tan sencillos que da la posibilidad de crear un sistema robusto y consistente. Por último, el marco de programación admite la aplicación de las herramientas de implementación, depuración e inspección [7].

A continuación, se explican los dos tipos fundamentales de módulos que forman dicha red. El modo de traspaso de información entre módulos se realiza mediante la conexión de las salidas de unos a las entradas de otros, aunque se pueden quedar entradas o salidas abiertas.

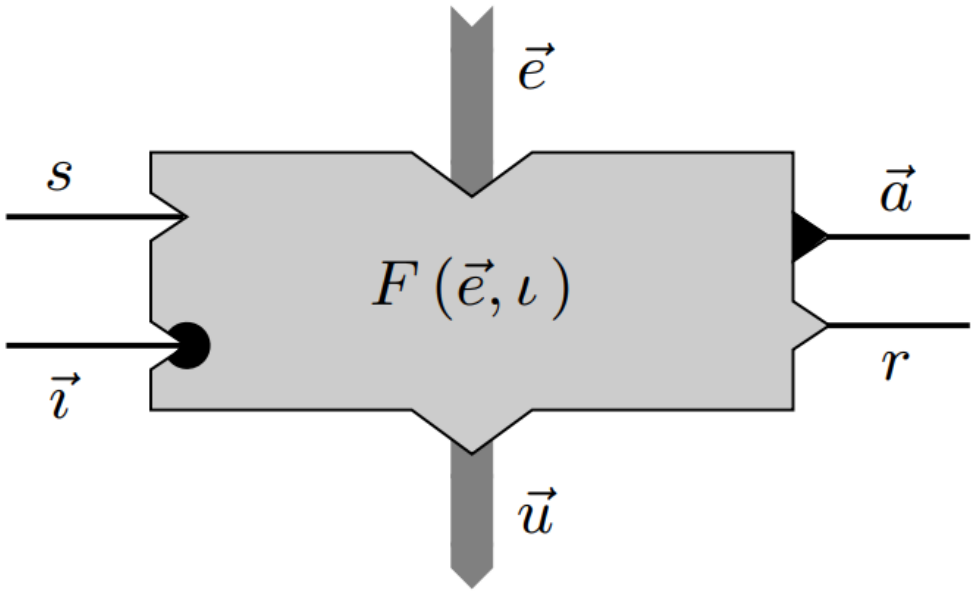

Figura 2.4. Módulo comportamiento

Los módulos de comportamiento son las unidades básicas de la red. Cada bloque realiza una acción, afecta y es afectado por otros módulos y, en consecuencia, el conjunto realiza una acción compleja.

Las entradas son el vector de datos de entrada, estimulación e inhibición.

-Vector de datos de entrada: $\bar{e} \in R^m$. Son los datos obtenidos por los sensores o por otros comportamientos.

-Estimulación: $s \in [0, 1]$. S=0 implica que no está estimulado, S=1 que está plenamente estimulado y valores intermedios, que está parcialmente estimulado.

-Inhibición: $\bar{\imath} \in [0, 1]$. La inhibición i es el máximo del vector inhibición $\bar{\imath}$. De igual manera que la estimulación, i=0 implica que no está inhibido, i=1 que está completamente inhibido y valores intermedios, que está parcialmente inhibido.

-Activación: $\iota \in [0, 1]$. Es una variable que se calcula en función de la estimulación e inhibición con la siguiente fórmula y representa si el módulo realizará alguna acción o no.

$$\iota = s * (1 - i)$$

Como se puede apreciar, si la inhibición vale 1 y el comportamiento está inhibido valga lo que valga la estimulación, la activación valdrá 0 y el módulo no afectará a la red, mientras que, si se quiere que el módulo funcione al 100%, la estimulación deberá ser 1 y la inhibición 0.

Las salidas son el vector de datos de salida, la actividad y el ratio.

-Vector de datos de salida: $\bar{u} \in R^n$. Son los datos generados por el comportamiento. Para calcularlo, se realiza la función de transferencia F, que variará dependiendo de los objetivos de cada módulo y depende únicamente del vector de entrada y la activación, cuya fórmula es la siguiente.

$$\bar{u} = F(\bar{e}, \iota)$$

-Actividad: $\bar{a} = (a_0, a_1, \ldots, a_{q-1})^T$. A=1 significa que todas las salidas van a tener un impacto muy significativo, a=0, que es un comportamiento inactivo, y valores intermedios, que está parcialmente activo.

-Ratio: $r \in [0, 1]$. Es un indicador de la satisfacción del comportamiento, siendo 0 satisfecho, 1 totalmente en desacuerdo y valores intermedios, parcialmente satisfecho.

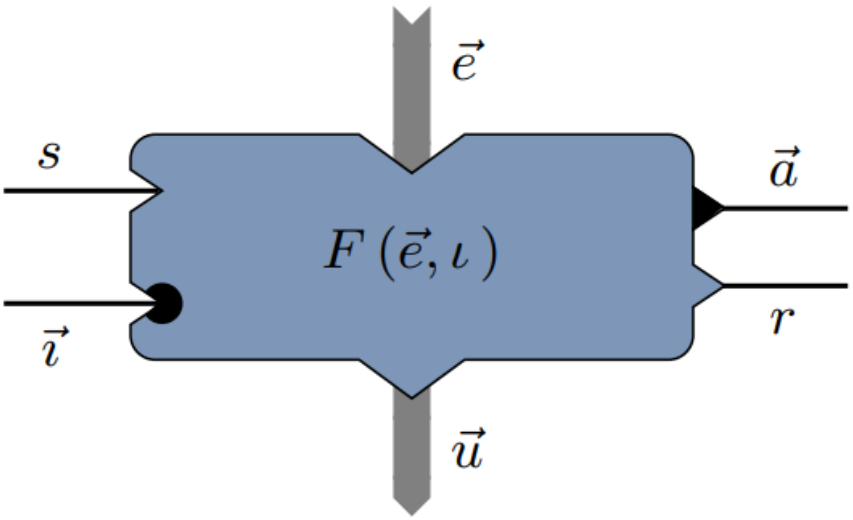

Figura 2.5. Módulo fusión

Respecto al módulo de fusión, hay tres tipos. Fusión máxima, fusión ponderada y fusión de suma ponderada. La función de estos módulos será la suma, parcial, total o ponderada de diferentes módulos de comportamiento.

-Fusión máxima: solo se tiene en cuenta el comportamiento más activo, el resto son ignorados. Su vector de datos de salida, actividad y ratio se calculan con las siguientes fórmulas.

$$\bar{u} = \bar{u}_s, s = argmax_c(a_c)$$

$$a = max_c(a_c)$$

$$r = r_s, s = argmax_c(a_c)$$

-Fusión ponderada: cada comportamiento tiene una influencia proporcional a su actividad. Su vector de datos de salida, actividad y ratio se calculan con las siguientes fórmulas.

$$\bar{u} = \frac{\sum_{j=0}^{p=1} a_j * \bar{u}_j}{\sum_{k=0}^{p=1} a_k}$$

$$a = \frac{\sum_{j=0}^{p=1} a_j^2}{\sum_{k=0}^{p=1} a_k}$$

$$r = \frac{\sum_{j=0}^{p=1} a_j * r_j}{\sum_{k=0}^{p=1} a_k}$$

-Fusión de suma ponderada: cada comportamiento tiene una influencia proporcional a su actividad e inversamente proporcional a la actividad máxima. Su vector de datos de salida, actividad y ratio se calculan con las siguientes fórmulas.

$$\bar{u} = \sum_{j=0}^{p=1} \frac{a_j * \bar{u}_j}{max_c(a_c)}$$

$$a = \min\left(1, \sum_{j=0}^{p=1} \frac{a_j^2}{max_c(a_c)}\right)$$

$$r = \frac{\sum_{j=0}^{p=1} a_j * r_j}{\sum_{k=0}^{p=1} a_k}$$

Por último, hay una serie de principios que se deben de cumplir.

-Principio 1: la actividad siempre será menor o igual que la activación.

-Principio 2: la actividad se mantiene constante siempre que el ratio sea igual a 0 y la activación constante.

-Principio 3: el ratio no está influenciado por la activación.

-Principio 4: en la fusión de comportamientos, tanto la actividad como el ratio están acotados por las siguientes fórmulas.

$$min_c(a_c) * \iota \leq a \leq \min\left(1, \sum_{j=0}^{p=1} a_j\right) * \iota$$

$$min_c(r_c) \leq r \leq max_c(r_c)$$

-Principio 5: la estimulación o inhibición solo podrán provenir de la actividad de uno o varios comportamientos.

-Principio 6: un comportamiento no se puede estimular o inhibir a sí mismo.

3. METOLOGÍA

3.1. Hardware

Para la realización de este proyecto se ha usado un portátil Asus© X555LJ. Tiene como procesador un Intel© Core™ i5 5200U, como memoria una DDR3L 1600 MHz SDRAM y 8 GB de RAM y por tarjeta gráfica, la NVIDIA© GeForce©920M con 2GB DDR3 VRAM.

3.2. Software

3.2.1. Lenguaje y sistema operativo

Se ha utilizado como lenguaje de programación Python 3.6.2, en el entorno de programación PyCharm Community Edition 2017.2.4 y se ha trabajado en dos sistemas operativos, Windows 10 Home de 64 bits y Ubuntu 16.04.

3.2.2. Vispy

Vispy es una librería de Python que hemos usado para la simulación y visualización de nuestro escenario e interacciones entre los equipos. Además, nos permite interactuar con la simulación mediante entradas por teclado o ratón. En este proyecto hemos utilizado estas funcionalidades para mover la cámara, hacer zoom y pausar, reanudar, acelerar y ralentizar la simulación.

Para la visualización, se superponen varias capas, de las que hay que destacar las llamadas "*vertex shader*" y la "*fragment shader*". La primera da información de la posición de cada punto, mientras que la segunda hace referencia al color.

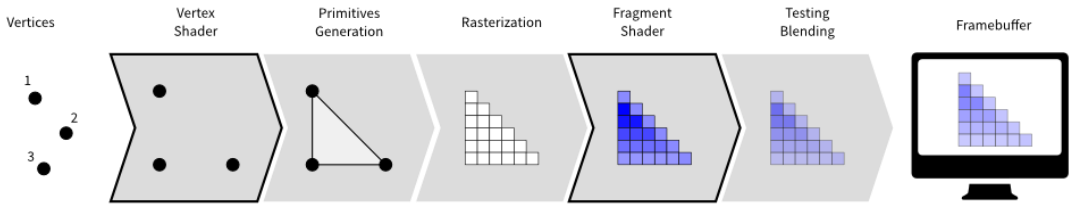

Figura 3.1. Estructura de capas en Vispy

3.3. Inteligencia individualista

La base de esta inteligencia es que los drones no tengan un objetivo predeterminado, sino que en función de los datos que van recibiendo por el radar, realicen la acción más eficiente. Está compuesto por 7 módulos de comportamiento y un módulo de fusión de suma ponderada. Esta es la inteligencia que usaremos para el equipo 1.

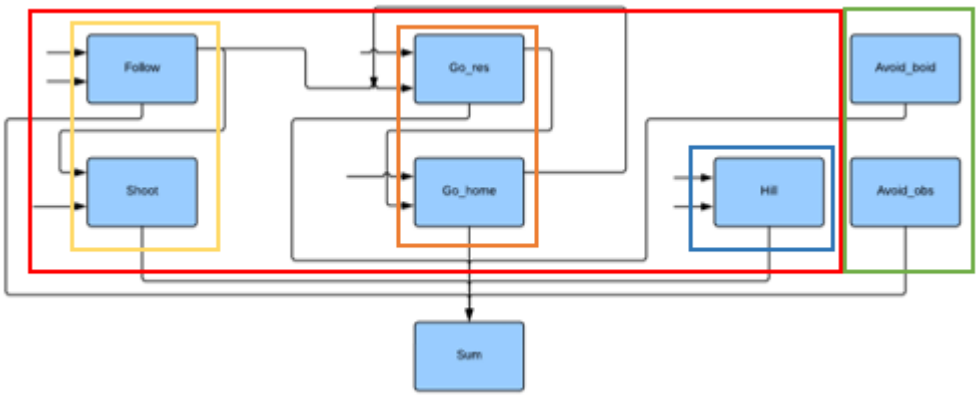

Figura 3.2. Estructura de la inteligencia individualista

A su vez, esta inteligencia está dividida en dos grandes bloques, el bloque pasivo (en verde) y el bloque activo (en rojo).

El bloque pasivo está compuesto por los comportamientos "*Avoid_boid*" y "*Avoid_obs*" y su función es la de esquivar los obstáculos y drones de su mismo equipo que se encuentran en su radar. Se denomina bloque pasivo porque no está afectado por ninguno de los otros comportamientos y está siempre plenamente estimulado.

El bloque agresivo está formado a su vez por 3 bloques más pequeños, el bloque de ataque (amarillo), de recursos (naranja) y de la colina (azul).

La función del bloque de ataque es la de perseguir y disparar a los drones del equipo enemigo, está estimulado por la identificación de enemigos y está inhibido por la carga del dron y su muerte. Dentro del bloque, el comportamiento que se encarga de disparar está estimulado por el de perseguir.

En el bloque de recursos, están los comportamientos de "*Go_res*" y "*Go_home*". Su objetivo es el de ir a las zonas de recursos y una vez lo obtienen, volver a la base para puntuar. El comportamiento de ir a por los recursos está inhibido por el comportamiento de perseguir drones enemigos, por la muerte y la carga. En cambio, el comportamiento de volver a la base está estimulado por la carga e inhibido por la búsqueda de recursos.

Por último, en el bloque colina nos encontramos con el comportamiento "*Hill*", que se encarga de ir y mantener la mayoría en el área central. Este comportamiento está completamente inhibido en el primer escenario, ya que no hay área por la que luchar, mientras que en el segundo escenario solo está inhibido por la muerte.

3.4. Inteligencia por roles

En contraste con la inteligencia anteriormente nombrada, la inteligencia por roles, como su nombre indica, se basa en que cada dron tiene una función predeterminada, o va a recolectar recursos o va a intentar perseguir drones enemigos o va a luchar por mantener la colina, aunque siempre que encuentren un dron enemigo, intentarán dispararle. Estructuralmente es casi igual que la anterior, solo que se le añade el comportamiento "*Rol*". Esta es la inteligencia que usaremos para el equipo 2.

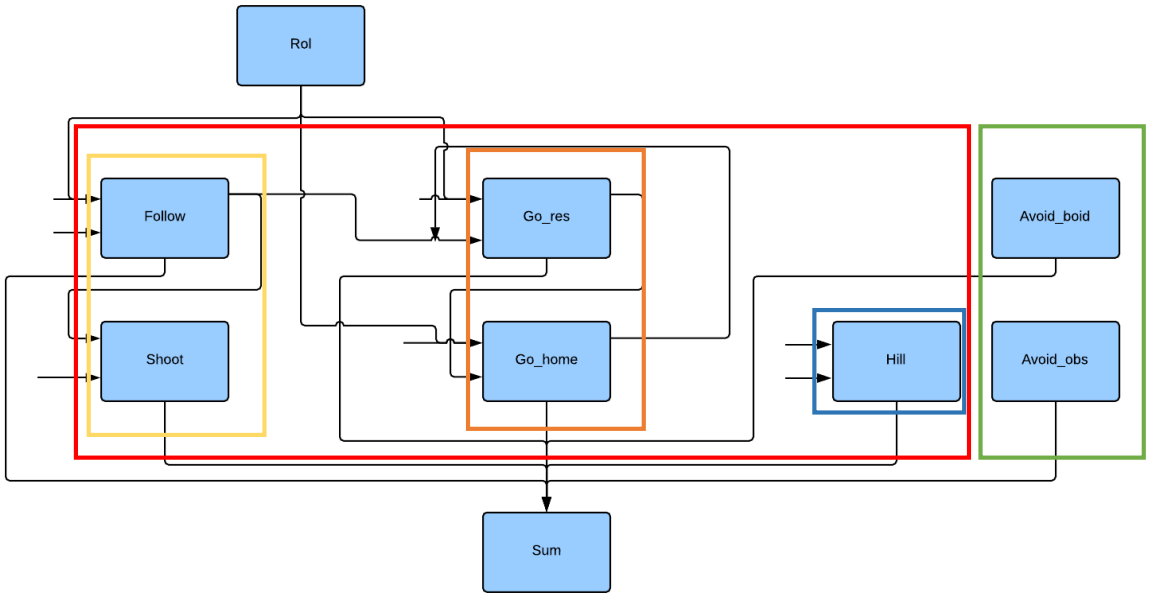

Figura 3.3. Estructura de la inteligencia por roles

De igual forma que la inteligencia individualista, la inteligencia por roles está separada por el bloque pasivo y activo, siendo el bloque pasivo exactamente igual. En cambio, en el bloque activo tiene la diferencia del comportamiento "*Rol*", que estimula e inhibe el bloque de ataque, de recursos y el de la colina.

3.5. Modelado de los drones

Como se ha mencionado más arriba, se han utilizado dos modelos diferentes de drones, el cuadricóptero y el dron de ala fija. En el modelo dinámico que se representa en la siguiente fórmula, solo se diferencian en la tau. Se ha considerado que para el ala fija tomase un valor de 1, mientras que en el cuadricóptero vale 2.

$$Vel = k * vel\_max * \left(1 - e^{-t/\tau}\right) + Vel * \ e^{-t/\tau}$$

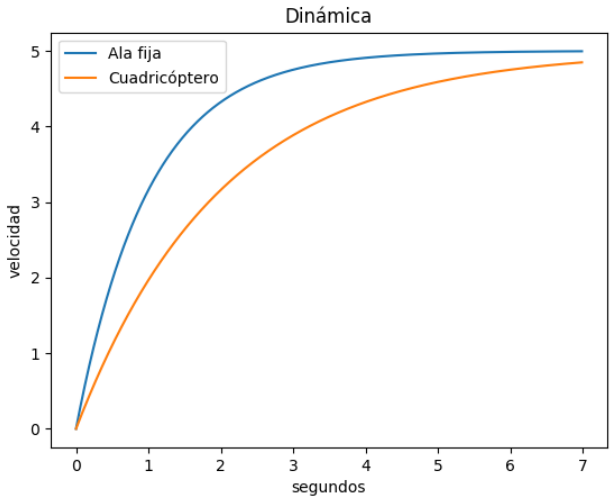

Figura 3.4. Representación del modelo dinámico de los dos drones

Para representarlos, se ha decidido que los cuadricópteros sean más grandes que los de ala fija.

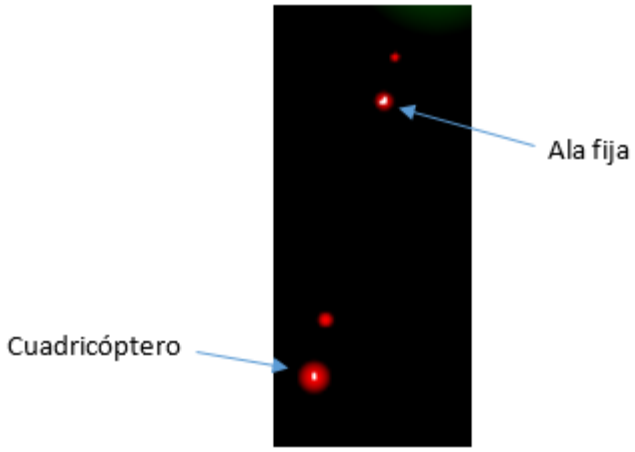

Figura 3.5. Visualización de los drones en el programa

También se puede apreciar que en la imagen hay dos puntos que acompañan a los drones. Esos puntos muestran la dirección en la que se mueve el dron. Además, la distancia del punto con el dron muestra su aceleración, siendo mayor cuanto más separados estén.

3.6. Radar

Esta función es la representación del radar físico que tendrían los drones reales.

Detectan drones, tanto aliados como enemigos, obstáculos y los impactos de las balas con los drones. Además, se encarga de gestionar las variables de la carga y la vida cuando un dron ha sido impactado y de calcular la variable "*battle*", que sirve para determinar qué equipo tiene mayoría de individuos en la zona central. Y, por último, rellena el vector "*priority*", cuyo propósito es determinar el orden de cercanía de los drones del equipo 1 respecto a la zona central.

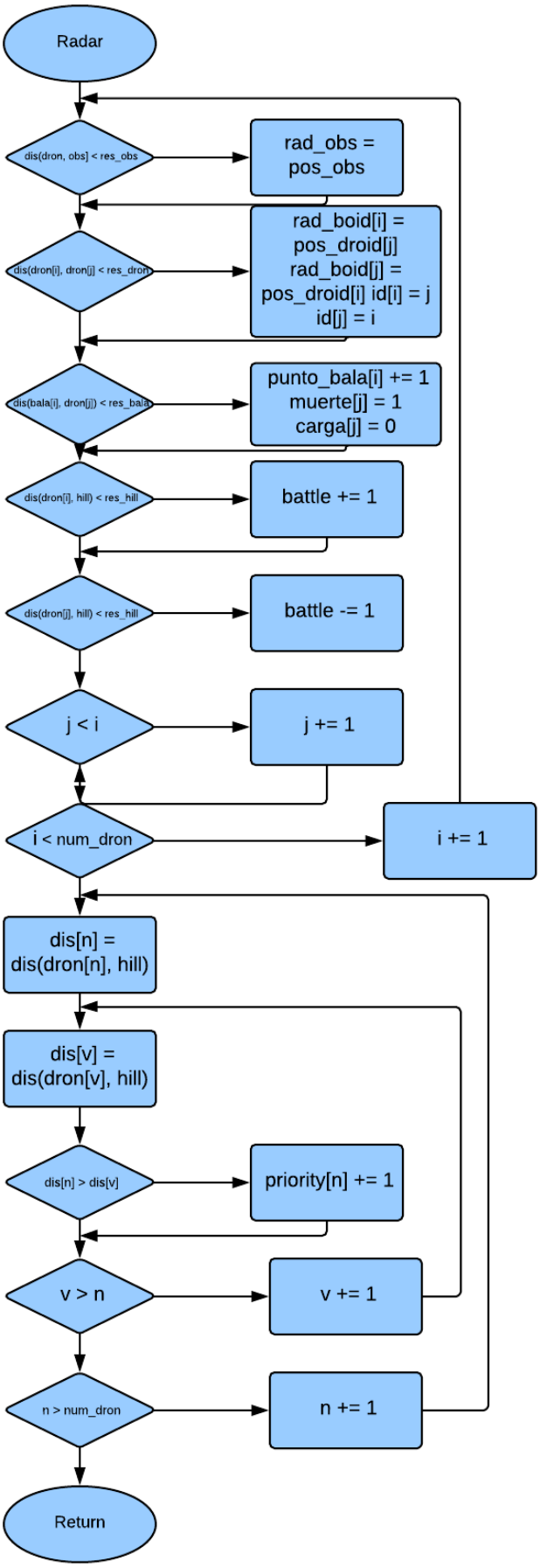

Figura 3.6. Flujograma de la función radar

3.7. Programa principal

El programa principal es el nexo entre todos los apartados anteriormente nombrados y el que se encarga de que el proyecto funcione correctamente. El programa empieza visualizando una ventana con una serie de instrucciones escritas que explican qué teclas sirven para interactuar con el programa, además del tiempo de simulación. A continuación, inicializa todas las variables y las funciones y da paso a un bucle que finaliza cuando ha transcurrido el tiempo de simulación y se pulsa la tecla 'e' del teclado. En dicho bucle se refresca y visualiza la ventana con todos los elementos del escenario y se calcula la siguiente iteración. El Δt del bucle, el tiempo que tarda en ejecutarse esta parte del código, es de 0,04 segundos. Cuando ha transcurrido el tiempo de simulación y se pulsa la 'e', el programa crea varias ventanas con unas gráficas de la evolución temporal de los puntos conseguidos por los dos equipos. Estas gráficas dependen del escenario en el que estemos. En el escenario 1 son solo dos ventanas, dándonos información sobre los puntos de los recursos e impactos obtenidos, pero en el escenario 2, se sacan dos ventanas más con la información de los puntos de la lucha por la colina.

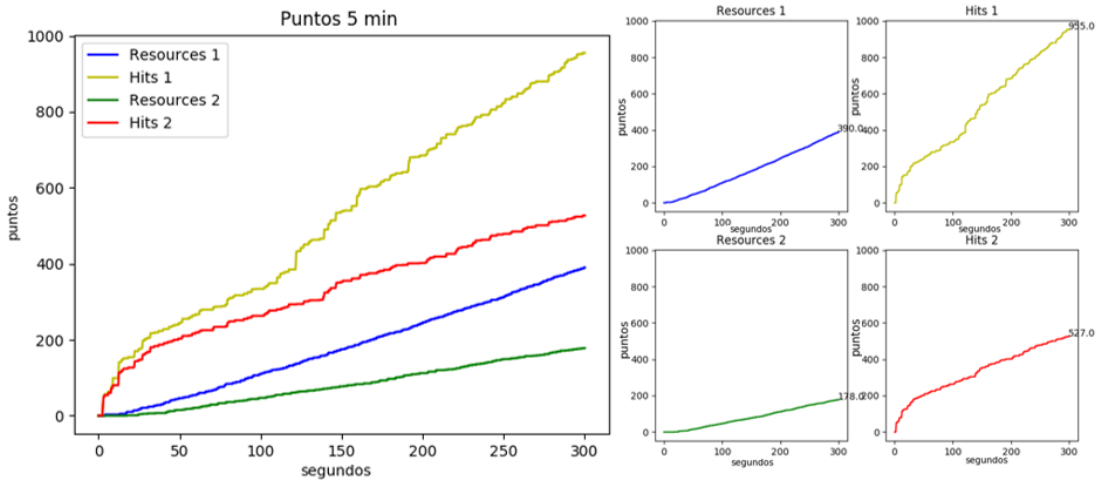

Figura 3.7. Gráficas del escenario 1, simulación 6, después de la optimización

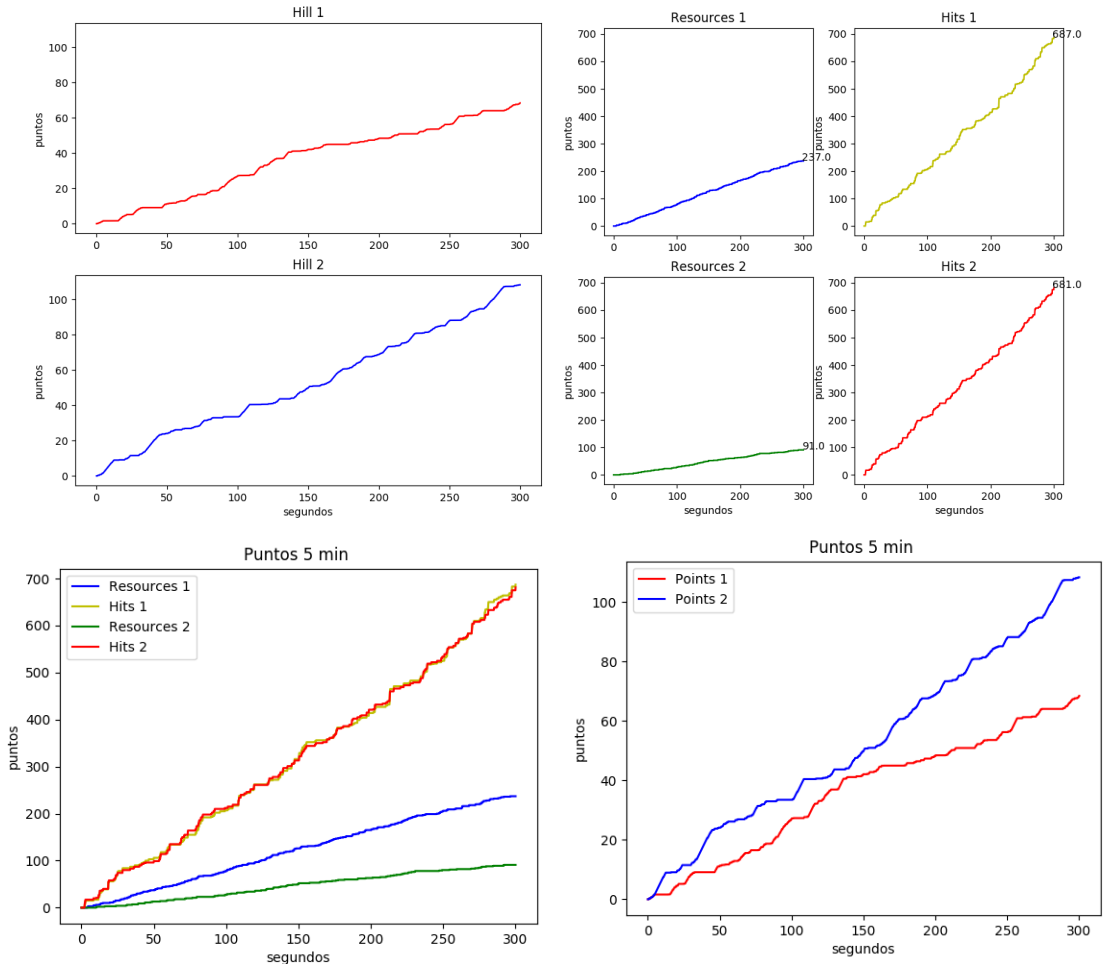

Figura 3.8. Gráficas del escenario 2, simulación 2, antes de la optimización

En cada iteración, el programa principal obtiene los datos recibidos por el sensor, a continuación, calcula la fuerza resultante de las dos inteligencias para cada dron. En el siguiente paso calcula la velocidad con esas fuerzas. En la siguiente instrucción calcula las velocidades y posiciones de las balas, si los drones abatidos ya han pasado el tiempo de revivir y calcula los puntos de la lucha por el área central y los recursos.

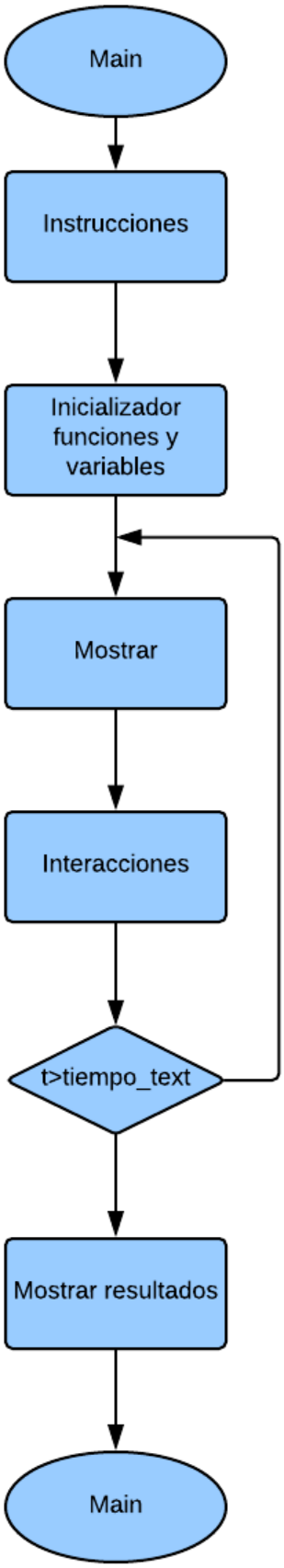

Figura 3.9. Flujograma del programa principal

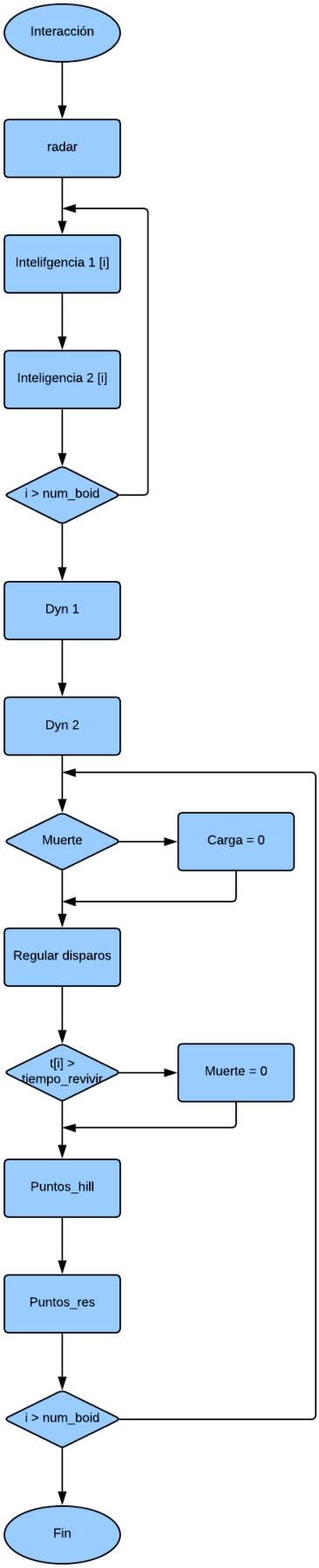

Figura 3.10. Flujograma de la función iteración del programa principal

3.8. Escenario 1

El primer escenario es un campo en 3D que tiene como elementos la base del equipo en rojo semitransparente, la base del equipo 2 en azul semitransparente, 6 obstáculos en blanco y 3 áreas de recursos en verde semitransparente. Los obstáculos están localizados en el punto medio entre las bases y los recursos para que tengan que esquivarlos, y las zonas de recursos están a la misma distancia de cada base, por lo que habrá interacción entre los drones de cada equipo. Además, como elementos auxiliares, están implementados un contador del tiempo que queda para visualizar los datos, un contador de velocidad de ejecución y unos ejes de referencia xyz.

En este escenario, los objetivos por los que luchan los equipos es, en primer lugar, obtener el mayor número de recursos posibles, por lo que deberán de ir a las zonas de recursos y volver a su base sin que les disparen los drones enemigos, y, en segundo lugar, disparar a los drones del otro equipo y acertar con la bala.

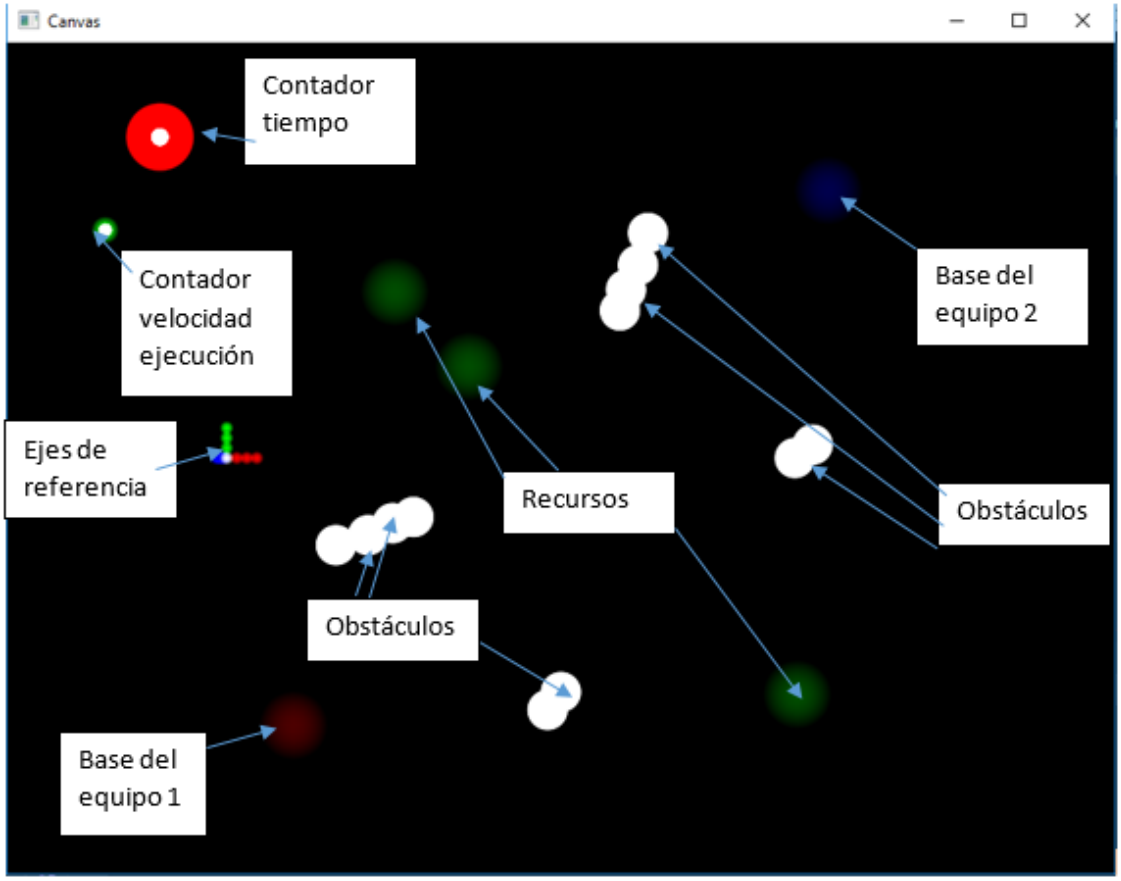

Figura 3.11. Visualización del escenario 1

3.9. Escenario 2

El segundo escenario tiene los mismos elementos, más un área central de color marrón semitransparente.

Además de los dos objetivos que tiene el escenario 1, en este caso se añade la lucha por mantener el control del área central. Dicho control se obtiene cuando uno de los dos equipos consigue tener más unidades en la zona que el otro equipo.

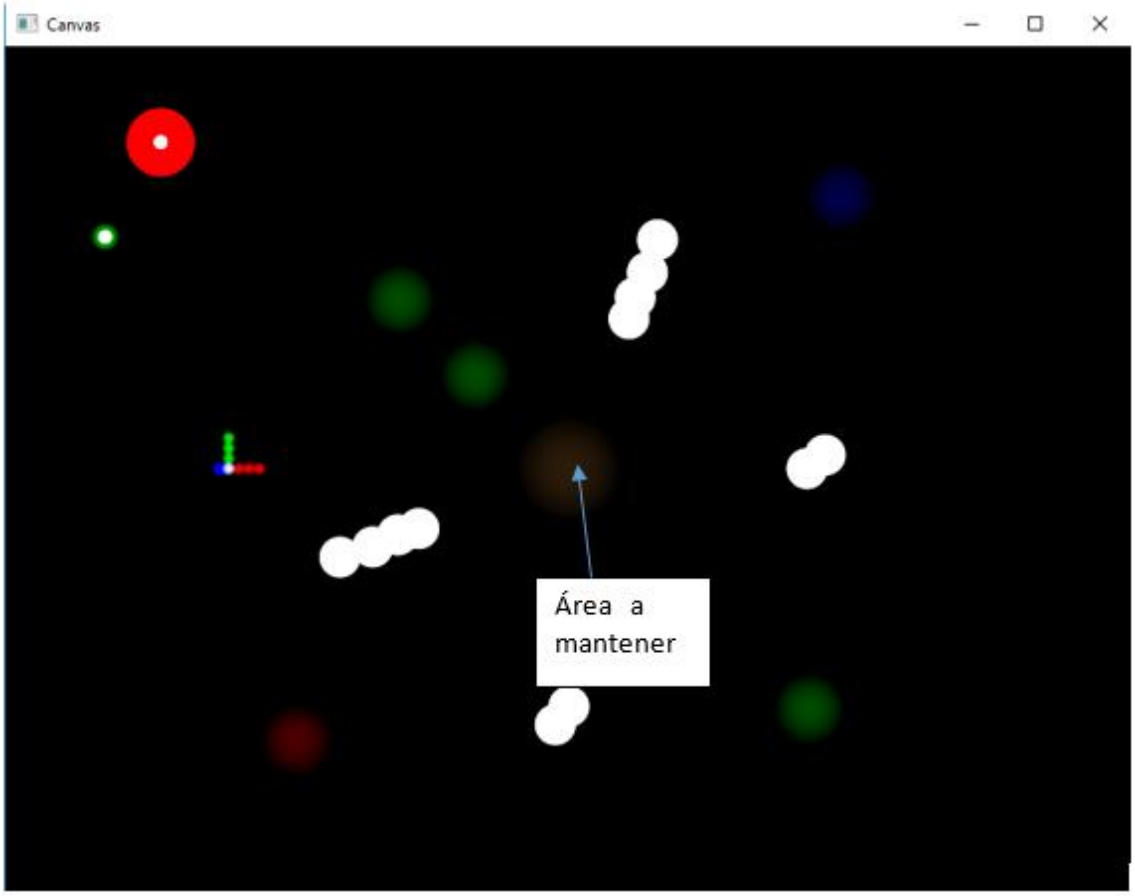

Figura 3.12. Visualización del escenario 2

3.10. Optimización

Al crear las inteligencias, los parámetros que más adelante se iban a optimizar se escogieron de forma aleatoria, de forma que a continuación se realizaría una mejora de la eficiencia de cada inteligencia. Dichos parámetros para el equipo con inteligencia por roles son los números de drones destinados a cada tarea, recolectar recursos, atacar drones enemigos, o en el caso del escenario 2, luchar por el control del área central. Dichos números deben sumar 10, solo aceptándose los números enteros. Y para el equipo de inteligencia individualista, varían los parámetros que afectan a las fuerzas que les indican la tarea que deben hacer. El rango de optimización para estas 2 o 3 variables, dependiendo del escenario, va de 1 a 5, con intervalos de 0,1.

El proceso de optimización que se ha utilizado ha sido un gradiente de descenso. Este sistema consiste en una serie de iteraciones donde se van variando los parámetros anteriormente nombrados hasta que encuentren una configuración que gane al otro equipo o se hayan probado todas las combinaciones entre los rangos impuestos.

3.11. Tratamiento de resultados

Para la obtención de los resultados, se van a realizar 10 simulaciones en cada escenario y antes y después de optimizar, con un tiempo de simulación de 5 minutos. A continuación, calcularemos el valor medio de las puntuaciones y la desviación típica con las siguientes fórmulas estadísticas. De esta forma, si la desviación es menor de un 10% respecto a la media, podremos afirmar que la muestra es suficientemente amplia y no necesitaremos realizar más simulaciones. Pero en el caso de ser mayor, simularemos hasta que la desviación sea menor al 10%. Si la diferencia entre las medias de las puntuaciones de los equipos en cada bloque de simulaciones es mayor que la suma de las desviaciones típicas, se podrá concluir que los resultados son aceptables.

$$Valor\ medio = \frac{\sum_{i=1}^{i=N} puntos_i}{N}$$

$$Desviación\ típica = \sqrt{\frac{\sum_{i=1}^{i=N} (puntos_i - valor\ medio)^2}{N}}$$

4. RESULTADOS Y DISCUSIÓN

En este apartado se van a comentar los resultados de todas las simulaciones que se han ido realizando, separadas en bloques de 10, antes y después de cada optimización. Los datos se van a representar en una única gráfica, con el eje 'y' representando los puntos y en el eje 'x' los segundos. Los puntos se visualizan en una evolución temporal durante los 5 minutos de cada simulación, con una transparencia del 50% para poder diferenciar todas las líneas. En color rojo están representados los puntos del equipo 1, y en verde los del equipo 2. Además, debajo de cada gráfica se expone una tabla con la media, desviación típica en valor absoluto y en valor relativo respecto a la media de cada equipo.

4.1. Escenario1

Antes de la optimización:

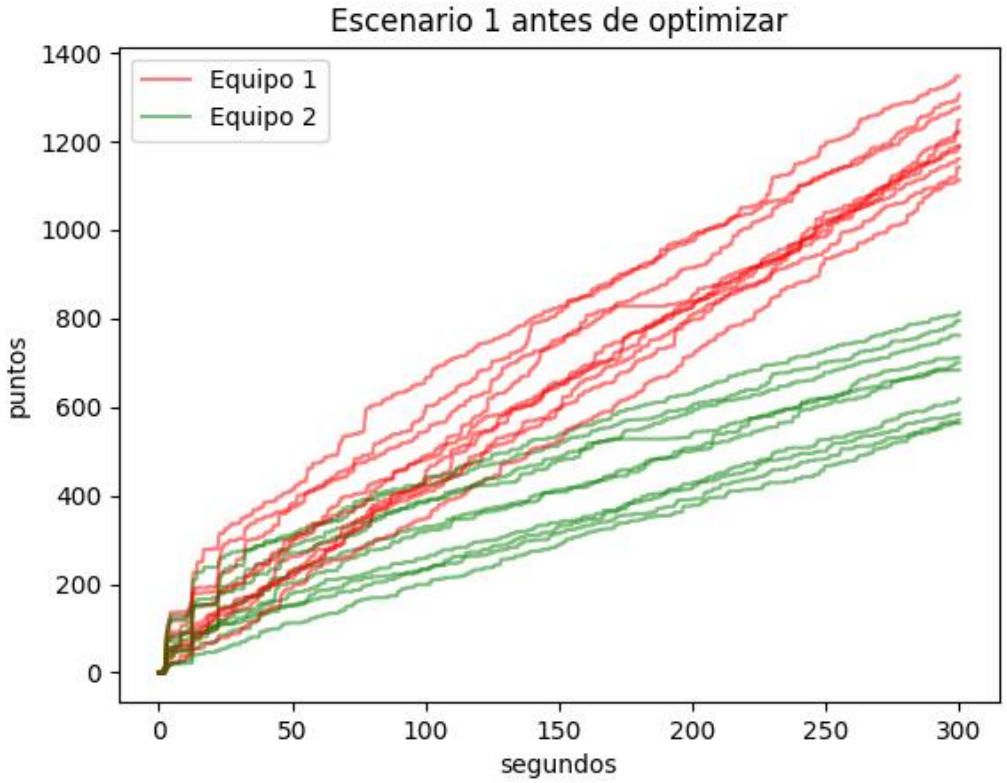

Figura 5.1. Puntuación del escenario 1 antes de optimizar

Media	1222,4	680,5
Desviación	24,85871186	30,99631599
Desviación %	2,033598811	4,554932548

Tabla 5.1. Media, desviación típica y desviación proporcional de escenario 1 antes de optimizar

Como se puede apreciar, el equipo 1 salió victorioso, por lo que, a continuación, se realizó la optimización al equipo 2.

Después de la optimización al equipo 2:

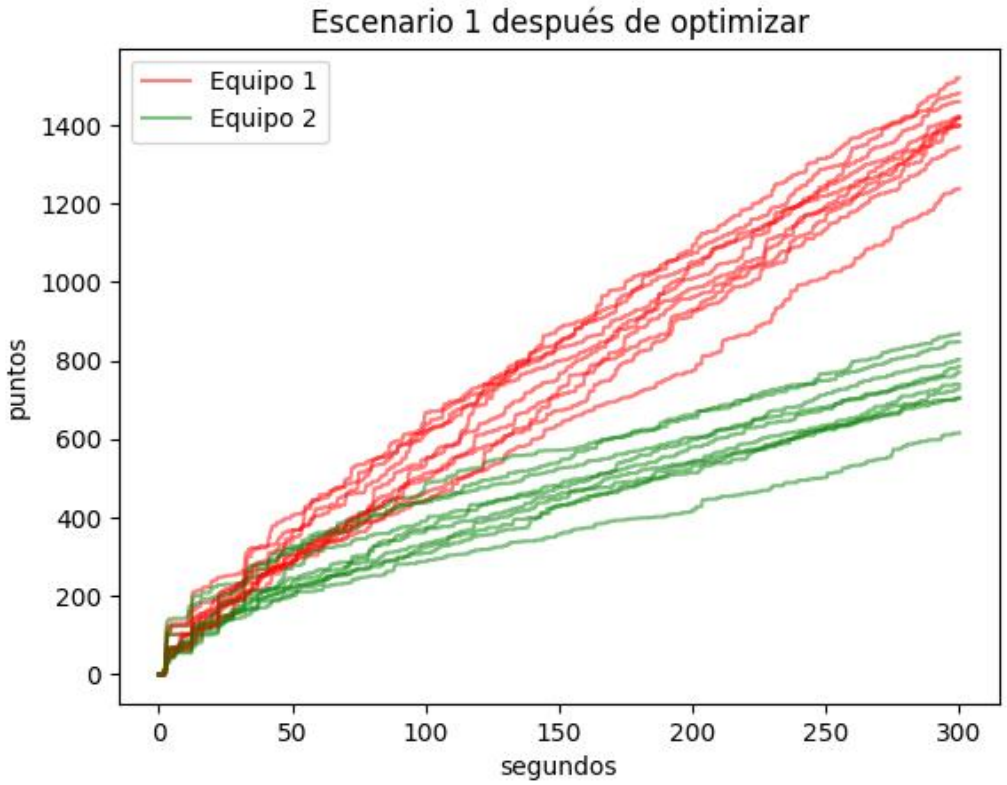

Figura 5.2. Puntuación del escenario 1 tras optimizar el equipo 2

Media	1410,7	757
Desviación	25,81513098	24,90525256
Desviación %	1,829951867	3,289993733

Tabla 5.2. Media, desviación típica y desviación proporcional de escenario 1 después de optimizar el equipo 2

Tras las 11 iteraciones, el equipo 2 no consiguió superar al equipo 1, tan solo aumentó su puntuación media.

4.2. Escenario 2

Antes de la optimización:

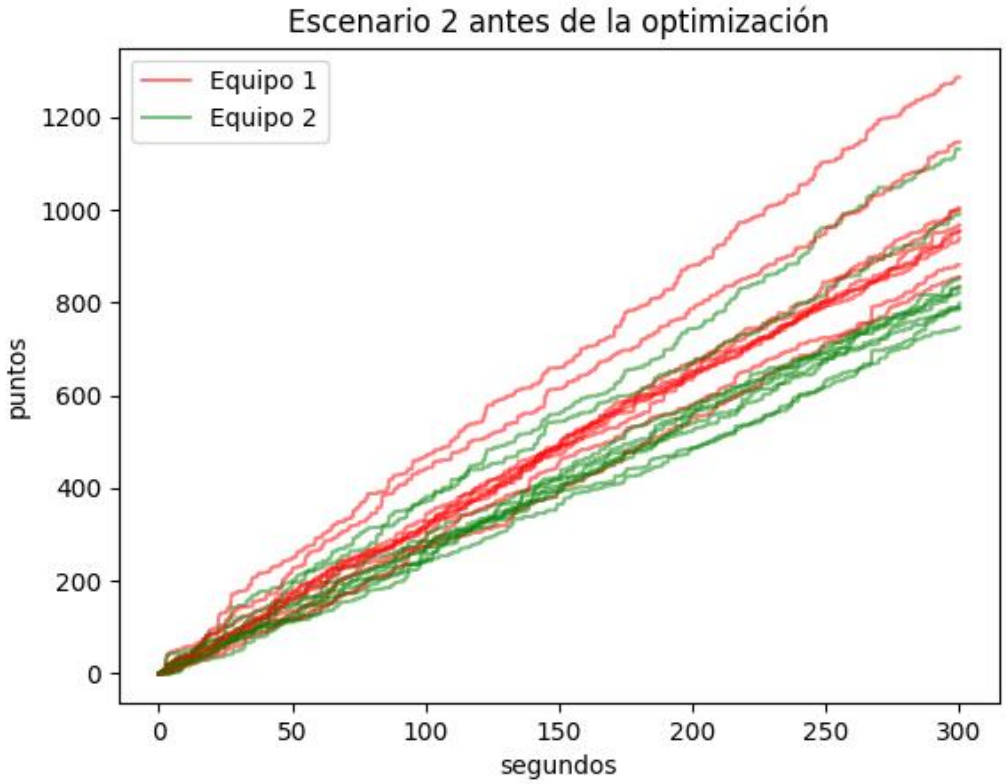

Figura 5.3. Puntuación del escenario 2 antes de optimizar

Media	998,793	858,717
Desviación	42,60703005	38,62700789
Desviación %	4,265851889	4,498223267

Tabla 5.3. Media, desviación típica y desviación proporcional de escenario 2 antes de optimizar

Como se puede apreciar, el equipo 1 salió victorioso, por lo que, a continuación, se realizó la optimización al equipo 2.

Después de la primera optimización del equipo 2:

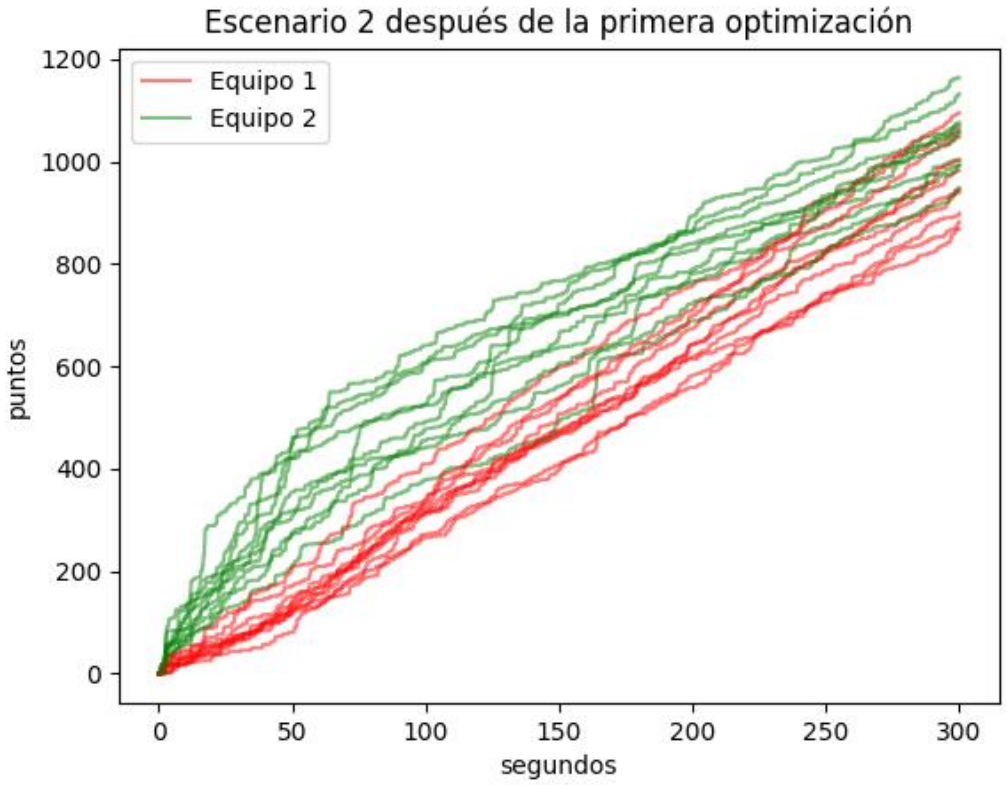

Figura 5.4. Puntuación del escenario 2 tras la primera optimización del equipo 2

Media	973,116	1049,815
Desviación	26,40617334	22,17443018
Desviación %	2,713568921	2,112222647

Tabla 5.4. Media, desviación típica y desviación proporcional de escenario 2 tras la primera optimización del equipo 2

Tras 42 iteraciones, el equipo 2 consiguió una combinación de parámetros que ganó al equipo 1, por lo que tuvimos que realizar una segunda optimización, solo que en este caso al equipo 1.

Después de la primera optimización del equipo 1:

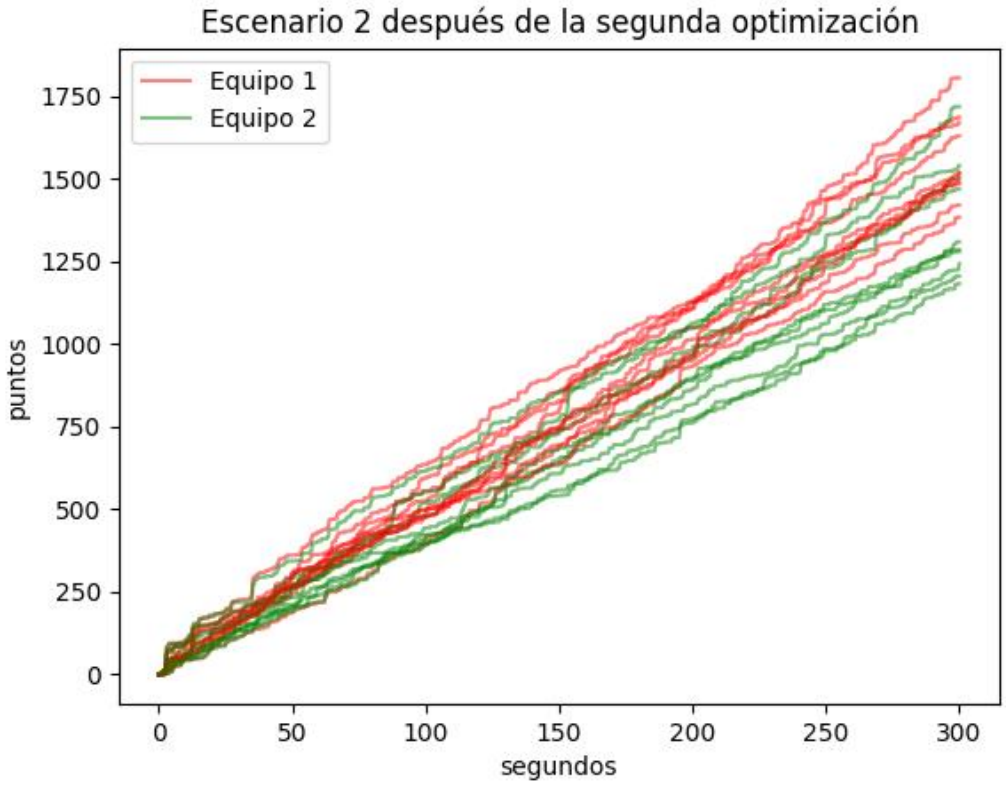

Figura 5.5. Puntuación del escenario 2 tras la primera optimización del equipo 1

Media	1560,501	1373,1
Desviación	44,15346012	58,23085678
Desviación %	2,829441322	4,24083146

Tabla 5.5. Media, desviación típica y desviación proporcional de escenario 2 tras la primera optimización del equipo 1

Tras 489 iteraciones, el equipo 1 volvió a ganar. Y en el último caso, se intentó optimizar el equipo 2.

Después de la segunda optimización del equipo 2:

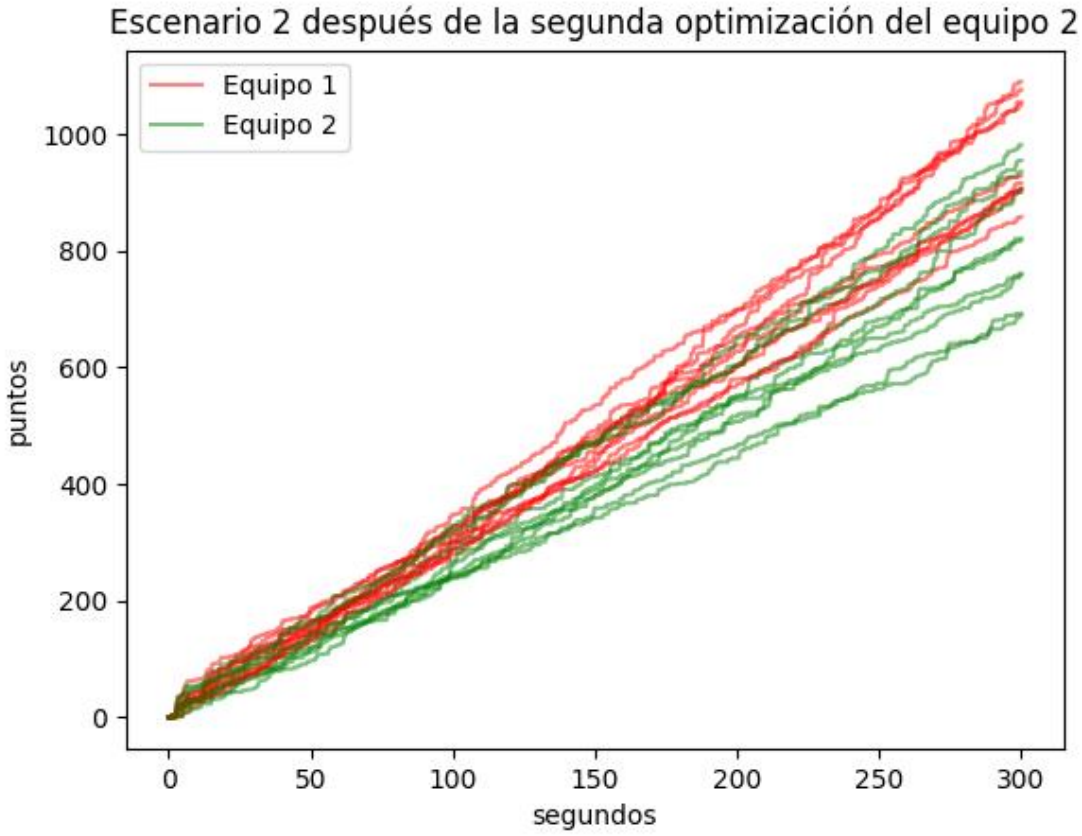

Figura 5.6. Puntuación del escenario 2 tras la segunda optimización del equipo 2

Media	968,994	832,061
Desviación	29,372964	35,946087
Desviación %	3,0312844	4,3201264

Tabla 5.6. Media, desviación típica y desviación proporcional de escenario 2 tras la segunda optimización del equipo 2

Tras todo el proceso de optimización, el equipo 2 no ha conseguido ganar al equipo 1.

5. IMPACTOS DEL TRABAJO

Como en cualquier investigación, tras concluir todos los objetivos establecidos, se considera esencial realizar una reflexión global sobre la repercusión que pudiera tener el presente trabajo en diferentes ámbitos, dejando al margen el aspecto técnico y valorándolo desde un punto de vista social y medioambiental. El impacto medioambiental de este trabajo es nulo, ya que no presenta ningún tipo de contaminación, ya sea acústica, energética, etc. Sin embargo, sí que podría presentar un considerable impacto social tanto en estudiantes como en cualquier interesado en el sector. Tras la finalización del trabajo, tanto el código como la memoria, estarán disponibles en repositorios virtuales de la universidad para permitir su libre acceso. Esto ofrecerá una base o referencia para aquellos que deseen iniciarse en el campo de los enjambres de robots y el control por comportamientos en Python. A continuación, se valoran los posibles beneficios de este trabajo, así como posibles aplicaciones. Adicionalmente, se plantean diferentes vías para continuar esta línea de investigación, tanto para mejorar la propia técnica como su implementación en otras tareas.

5.1. Aplicaciones y beneficios

Este proyecto reúne un conjunto de puntos de interés para futuros proyectos relacionados con este contexto. Las aportaciones que se han conseguido desarrollar y que por lo tanto pueden servir de punto de partida en un futuro son las siguientes:

-Implementación de inteligencias mediante control basado en comportamientos iB2C en Python.

-Modelado de enjambres de robots heterogéneos en Python.

-Uso de técnicas de optimización con un gradiente de descenso.

-Base de referencia para comprobar la eficiencia de una inteligencia nueva.

5.2. Futuras líneas de investigación

Después de haber analizado los resultados obtenidos en este proyecto, se pueden definir varias líneas futuras de trabajo como son:

-La búsqueda de un mayor número de parámetros a optimizar, utilizando algoritmos genéticos para facilitar su optimización.

-La implementación de los escenarios en simuladores como Gazebo o Unity. Dicha implementación sería directa, ya que no habría que cambiar el lenguaje de programación y las inteligencias se encuentran en una estructura de archivos independientes de los escenarios, dinámicas y la función radar.

-La implementación en el mundo real con drones físicos mediante ROS (*Robot Operating System*).

-La optimización de cada inteligencia se podría realizar enfrentándose a sí misma, para lograr una optimización más detallada de los parámetros con técnicas de "*machine learning"*.

-Profundizar en algunos comportamientos, añadiendo más detalles como el cálculo de la trayectoria de los drones para aumentar la precisión de los disparos.

-La adición de un mayor número de sensores para aumentar la eficiencia de las inteligencias.

-El uso de las inteligencias como referencia para la creación de nuevos modelos, tanto gobernadas por comportamientos como con otras técnicas.

Todo este desarrollo se ha realizado en vistas a poder tener una base sólida y de garantías para continuar con investigaciones y trabajos futuros relacionados con enjambres de drones y control mediante comportamientos. Esto quiere decir que las técnicas utilizadas o desarrolladas no solo sean aplicables a nuestro caso, sino que puedan servir como base de trabajo en futuros proyectos.

6. CONCLUSIONES

En este proyecto se ha abordado la cuestión de qué inteligencia, si una que decida sus acciones según la información recibida por los sensores, o una con sus tareas predeterminadas, es más eficiente dentro de dos escenarios y con los objetivos de recolectar recursos, derribar drones enemigos y mantener más drones que el equipo enemigo en un área acotada.

Para obtener una puntuación medible, se han realizado paquetes de 10 simulaciones. Tras la realización de todas estas simulaciones, podemos concluir que el equipo 1, el enjambre con inteligencia individualista, es más eficiente que el enjambre con inteligencia por roles, ya que, tras todas las optimizaciones, ha salido victoriosa en los dos escenarios y bajo las condiciones establecidas.

Además, como la desviación típica ha resultado en todos los casos menor del 10% de la media, se puede afirmar que el programa es estable y no tiene errores que provoquen resultados con puntuaciones aleatorias.

Es destacable el hecho de que en el primer escenario la diferencia es bastante mayor que en el segundo. Esto se debe a que, al no tener en cuenta el objetivo de mantener la mayoría de drones en el área central, las posibilidades de optimización del segundo equipo son menores y tiene menos flexibilidad, dejando al equipo con la inteligencia individualista con un mejor rendimiento. En futuros trabajos, el incremento en el número de parámetros a optimizar en el equipo 2 podría ayudar a solucionar este problema.

.

7. BIBLIOGRAFÍA

1. Nex, Francesco & Remondino, Fabio. (2014). UAV for 3D mapping applications: A review. Applied Geomatics. 6. 10.1007/s12518-013-0120-x.

2. Tractica, 2016. Tractica. [Internet] Available at: https://www.tractica.com/newsroom/press-releases/consumer-drone-sales-to- 92 increase-tenfold-to-67-7-million-units-annually-by-2021/ (Accessed May 2018).

3. Beni, G.: From swarm intelligence to swarm robotics. In S¸ahin, E., Spears, W., eds.: Swarm Robotics: State-of-the-art Survey. Lecture Notes in Computer Science 3342, Springer-Verlag (2005)

4. Wolfram, S. “A New Kind of Science”, Wolfram Media (2002)

5. http://ec.europa.eu/enterprise/sectors/aerospace/uas/ (accessed on June 2018).

6. Şahin E. (2005) Swarm Robotics: From Sources of Inspiration to Domains of Application. In: Şahin E., Spears W.M. (eds) Swarm Robotics. SR 2004. Lecture Notes in Computer Science, vol 3342. Springer, Berlin, Heidelberg

7. Proetzsch M., Luksch T., Berns K. (2007) The Behaviour-Based Control Architecture iB2C for Complex Robotic Systems. In: Hertzberg J., Beetz M., Englert R. (eds) KI 2007: Advances in Artificial Intelligence. KI 2007. Lecture Notes in Computer Science, vol 4667. Springer, Berlin, Heidelberg

8. Simon K. S. (2017) Feasibility study of Unmanned Aerial Vehicles (UAV) application for ultrasonic non-Destructive Testing (NDT) of Wind Turbine Rotor Blades (Master thesis) Retrieved from https://munin.uit.no/bitstream/handle/10037/11350/thesis.pdf

9. I. C. Price and G. B. Lamont, "GA Directed Self-Organized Search and Attack UAV Swarms," Proceedings of the 2006 Winter Simulation Conference, Monterey, CA, 2006, pp. 1307-1315.

ANEXO I: PLANIFICACIÓN TEMPORAL Y PRESUPUESTO

Planificación temporal

En este anexo se va a realizar un estudio de la distribución temporal de las horas dedicadas a este TFG. La tabla 1 muestra un desglose de las tareas con la fecha de inicio, duración en horas y fecha de finalización. En la figura 1, se muestra la misma información que en la tabla 1 en forma de diagrama de Gantt. Por último, en la figura 2 se representa un diagrama de barras con las tareas más importantes y las horas dedicadas.

Tarea	Fecha inicio	Duración (horas)	Fecha final	Duración (días)
Documentación	15-sep	30	30-sep	15
Familiarizarme con Python	10-oct	10	17-oct	7
Matplotlib	17-oct	20	02-nov	16
Exámenes	18-dic	---	23-ene	36
Vispy	02-nov	20	16-nov	14
Pygame	01-dic	15	18-dic	17
Escenario 1	26-ene	6	31-ene	5
Int 1. Estructura por funciones	28-ene	39	08-feb	11
Int 1. PyiB2C	09-feb	21	15-feb	6
Int 1. Bloque pasivo	16-feb	85	20-mar	32
Int 1. Corrección de errores acumulados	21-mar	31	31-mar	10
Int 1. Bloque activo	01-abr	79	25-abr	24
Int 1. Optimización	25-abr	18	01-may	6
Escenario 2	02-may	5	04-may	2
Int 2. Bloque pasivo	02-may	19	10-may	8
Int 2. Bloque activo	10-may	32	18-may	8
Int 2. Optimización	18-may	12	22-may	4
Visualización de datos	22-may	8	27-may	5
Memoria	28-may	127	10-jul	43
Comentar código	28-may	5	30-may	2
Documentación adicional	29-may	12	06-jun	8
Presentación 1	16-feb	4	21-feb	5
Presentación 2	30-mar	5	04-abr	5
Presentación 3	04-may	8	09-may	5
Horas totales		611		

Tabla Anexo I.1. Desglose de tareas de planificación temporal

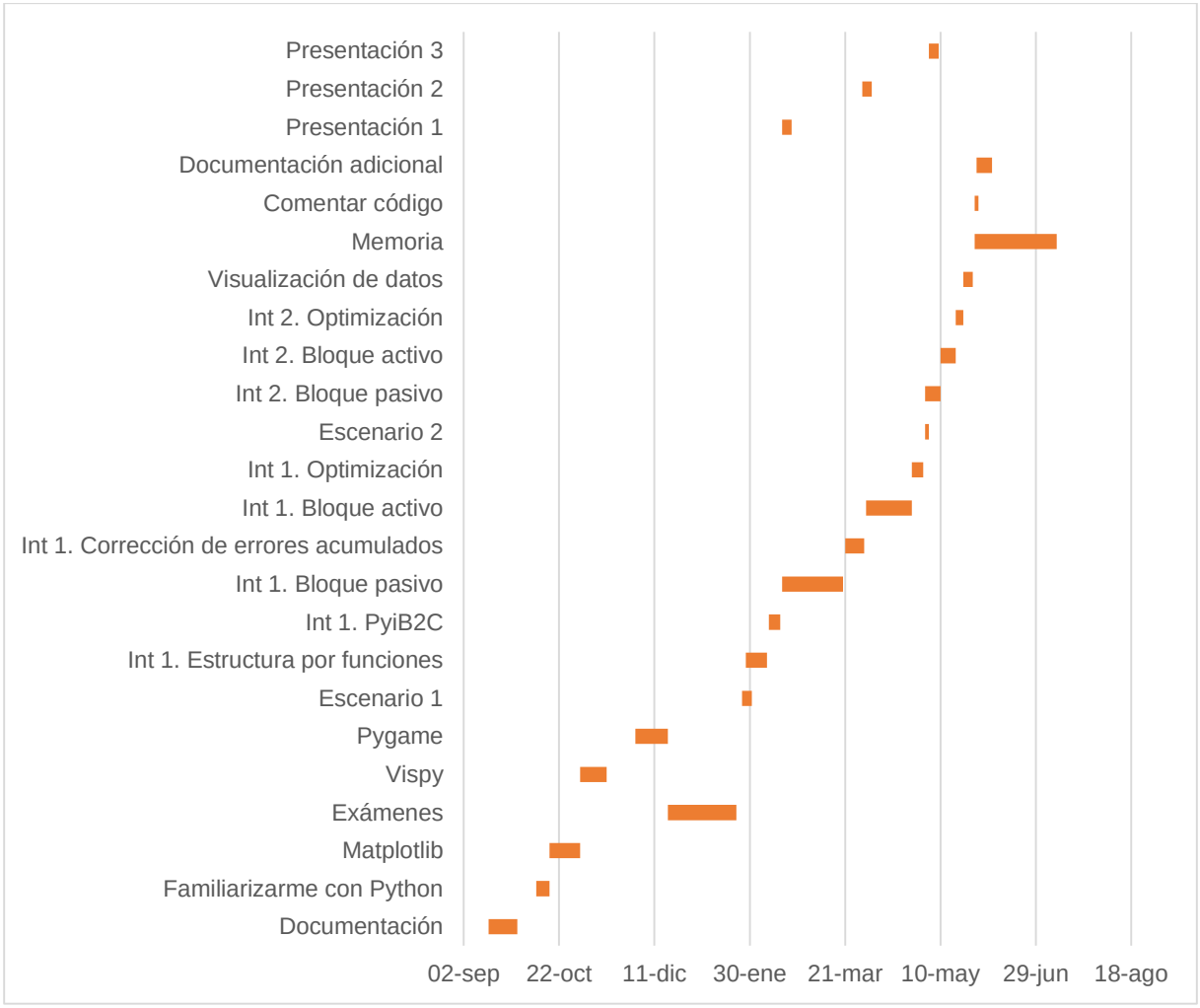

Figura Anexo I.1. Diagrama de Gantt

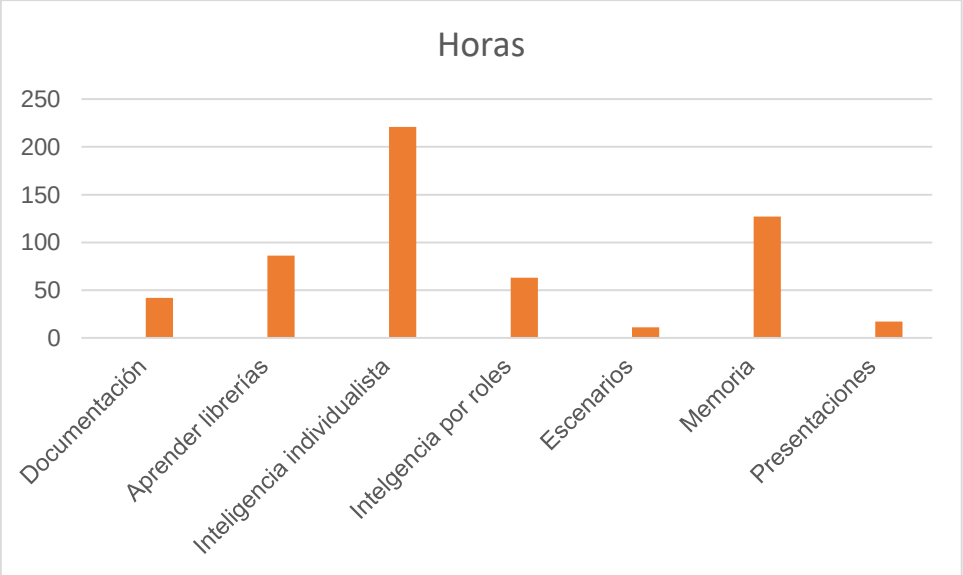

Figura Anexo I.2. Diagrama de barras de horas dedicadas a las partes más destacables

Estudio económico

Para el estudio económico, se van a analizar los costes de personal y los costes materiales. Los costes personales son únicamente las horas dedicadas al proyecto tanto por el tutor como el alumno tal y como se muestran en la tabla 2. Respecto a los costes materiales, en la tabla 3 se va a realizar un desglose de todos los gastos requeridos por el proyecto. Por último, en el gráfico circular se muestran los costes totales en porcentajes.

	Horas	€/hora	€
Alumno	611	20	12220
Tutor	50	40	2000
			14220

Tabla Anexo I.2. Costes de personal

Herramienta	Precio
Portátil	699 €
Linux	0 €
PyCharm	0 €
Microsoft Office	149 €
Windows 10 Home	145 €
	993 €

Tabla Anexo I.3. Costes materiales

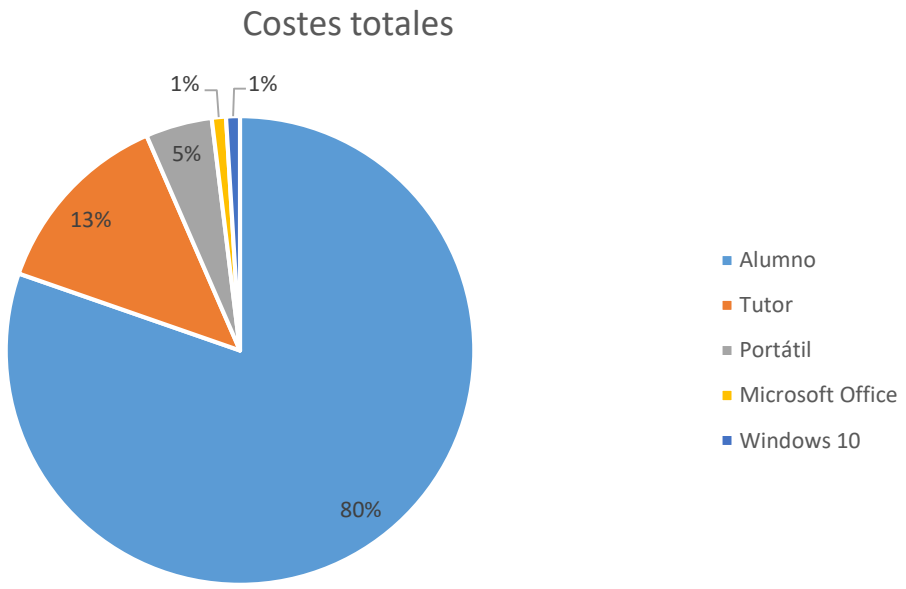

Figura Anexo I.3. Costes totales

Estructura de Descomposición del Proyecto

En la figura 4 se presenta la Estructuración de Descomposición del Proyecto (EDP). Se han ordenado jerárquicamente los paquetes de trabajo necesarios para la consecución del objetivo general de este Trabajo Fin de Grado.

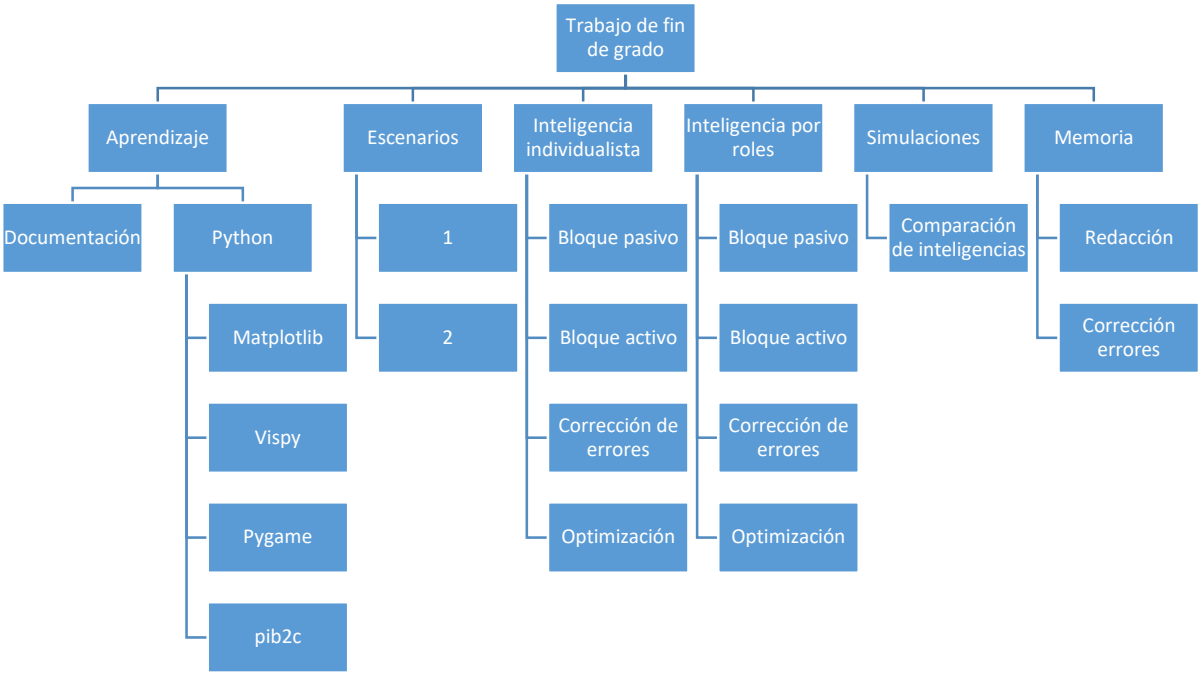

Figura Anexo 1.4. Estructuración de Descomposición del Proyecto

ANEXO II: ÍNDICE DE FIGURAS Y TABLAS

Índice de figuras:

Índice de tablas:

ANEXO III: GLOSARIO Y ABREVIATURAS

Glosario

ē	Vector de datos de entrada en iB2C
s	Estimulación
ī	Inhibición
ι	Activación
ū	Vector de datos de salida
ā	Actividad
r	Ratio
τ	Constante de tiempo
k	Constante de regulación de la velocidad máxima

Abreviaturas

UAV	Unmanned aerial vehicle
EDP	Estructura de Descomposición del Proyecto
ROS	Robot Operating System
TFG	Trabajo Final de Grado