Simulación y Optimización de Entregas de Última Milla con Drones en Entornos Urbanos
Simulation and Optimization of Last Mile Deliveries with Drones in Urban Environments

Trabajo de Fin de Grado
Curso 2024–2025

Autores

Francisco Mollá Astrar (Calificación: 8,5)
Jorge Rubio Carrizo (Calificación: 8,5)

Directores

Sergio Bernabé García
Sandra Catalán Pallarés

Grado en Desarrollo de Videojuegos
Grado en Ingeniería Informática
Facultad de Informática
Universidad Complutense de Madrid

Simulación y Optimización de Entregas de Última Milla con Drones en Entornos Urbanos
Simulation and Optimization of Last Mile Deliveries with Drones in Urban Environments

Trabajo de Fin de Grado en Desarrollo de Videojuegos e Ingeniería Informática

Autores
Francisco Mollá Astrar (Calificación: 8,5)
Jorge Rubio Carrizo (Calificación: 8,5)

Directores
Sergio Bernabé García
Sandra Catalán Pallarés

Convocatoria: *Junio 2025*

Grado en Desarrollo de Videojuegos
Grado en Ingeniería Informática
Facultad de Informática
Universidad Complutense de Madrid

26 de mayo de 2025

Dedicatoria

A nuestros padres, por aguantar tantas quejas sobre drones, espacios aéreos y conversaciones técnicas sin tener ni idea del tema.

A nuestros familiares y amigos en general, ya que nuestras quedadas se convertían solo en escucharnos hablar sobre el funcionamiento de drones.

Y, sobre todo, el uno al otro: por todo el tiempo compartido, las ideas que se pulieron en llamada y la confianza mutua que hizo que los obstáculos fuesen menos difíciles.

Agradecimientos

Queremos expresar nuestro reconocimiento a todos los profesores de la Universidad. Cada asignatura, ha dejado huella en este trabajo. Gracias por las clases que despertaron curiosidad, los proyectos que desafiaron nuestros límites y, sobre todo, la cercanía con la que habéis sabido acompañarnos durante la carrera.

Un agradecimiento especial a Sandra Catalán, por sus ideas frescas y su dedicación constante. Sus sugerencias técnicas y su forma de plantear problemas han sido clave para dar forma a la parte más creativa del proyecto.

También queremos reconocer la labor de Sergio Bernabé. Su exigencia académica nos empujaron a ir más allá, afinando cada detalle y asegurando la solidez de los resultados.

Por último, un guiño a la informática en general: a la comunidad que comparte conocimiento, a los foros y al ecosistema de herramientas abiertas que han hecho posible este Trabajo de Fin de Grado, en especial a los foros relacionados con toda la tecnología emergente de los drones.

Resumen

Simulación y Optimización de Entregas de Última Milla con Drones en Entornos Urbanos

En los últimos años, el uso de drones ha cobrado protagonismo en múltiples sectores. Uno de los ámbitos con mayor proyección es la logística de última milla, donde los drones se perfilan como una solución innovadora para realizar entregas rápidas, eficientes y sostenibles en zonas urbanas. Grandes empresas ya investigan cómo incorporar estos vehículos no tripulados para reducir tiempos de entrega, minimizar costes y disminuir la huella ambiental.

El presente Trabajo de Fin de Grado (TFG) se centra en el estudio y simulación de un sistema de entregas de última milla utilizando drones en entornos urbanos. Para ello, se ha llevado a cabo una serie de fases que incluyen la selección de modelos de drones adecuados, la utilización del software pertinente para llevar a cabo las simulaciones, y la conexión con un dron virtual. Se han diseñado rutas de entrega de distintas longitudes y se han ajustado los parámetros técnicos del dron para simular un comportamiento realista.

Este estudio proporciona una base experimental útil para el análisis de la viabilidad de implementar el reparto de paquetes a través de drones en la logística urbana en el futuro.

Palabras clave

ArduPilot, Dron, QGroundControl, Reparto, Simulación, SITL, Última Milla, Ruta

Abstract

Simulation and Optimization of Last Mile Deliveries with Drones in Urban Environments

In recent years, the use of drones has gained prominence in multiple sectors. One of the areas with the greatest potential is the logistics of the last mile, where drones are emerging as an innovative solution for fast, efficient, and sustainable deliveries in urban areas. Large companies are already investigating how to incorporate these unmanned vehicles to reduce delivery times, minimize costs, and reduce their environmental footprint.

This work focuses on the study and simulation of a last-mile delivery system using drones in urban environments. To this end, a series of phases were carried out, including the selection of suitable drone models, the use of relevant software to carry out the simulations, and the connection to a virtual drone. Delivery routes of varying lengths were designed, and the technical parameters of the drone were adjusted to simulate realistic behavior.

This study provides a useful experimental basis for analyzing the feasibility of implementing package delivery via drones in urban logistics in the future.

Keywords

ArduPilot, Drone, QGroundControl, Delivery, Simulation, SITL, Last Mile, Mission Plan

Índice

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

"La ciencia siempre vale la pena, porque sus descubrimientos, tarde o temprano, siempre se aplican."
— Severo Ochoa

Los servicios postales y de mensajería tradicionalmente se encargaban del envío de cartas y otros documentos. Permitiendo a la gente mantener correspondencia con personas que podían vivir a grandes distancias. Con la irrupción en la década de los 70, a manos de Ray Tomlinson del correo electrónico y su popularización en las décadas posteriores, poco a poco se fue reduciendo la carga de trabajo que afrontaban estas empresas en esa línea. Debido a la inmediatez y comodidad de la nueva alternativa, pasó de estar apenas presente en el ámbito académico a convertirse en un estándar de nuestra sociedad. Además, con carácter más reciente se están tratando de reducir las comunicaciones a papel con una motivación más medioambiental, lo que acentúa esta tendencia.

Sin embargo, en la última década, los servicios de mensajería y los servicios postales han vivido un importante resurgimiento. Principalmente a causa del éxito y normalización de las compras online y otros servicios disponibles en línea a manos de la población general. Popularizándose las entregas a domicilio o en puntos de recogida y, en consecuencia, generando nuevas preguntas y cuestiones relacionadas con cómo entregar dichos paquetes y cómo afrontar todos los desafíos de logística derivados de ello.

En este contexto global, la posibilidad de disponer de alternativas a los métodos de entrega tradicionales cobra un interés especial. Destacando, debido a su versatilidad y gran potencial, los drones como una de las opciones que pueden suponer una mayor revolución de los servicios de mensajería tal y como los conocemos.

La de los drones es una industria que aún está iniciándose, encontrándose en una etapa muy temprana en cuanto a legislación y en cuanto a investigación en muchos aspectos. No obstante, gracias a proyectos piloto e investigación de aplicaciones para distintos usos en distintos tipos de contextos (entregas de paquetería, reparto de comida, exploración y vigilancia de áreas, espectáculos recreativos...) poco a poco va posicionándose en muchos casos como la mejor opción, prueba de ello es como paulatinamente se va normalizando su presencia y su uso en muchas facetas de nuestras vidas.

1.1. Motivación

La motivación que respalda este trabajo viene dada por varios motivos. Por un lado, las entregas de paquetes están a la orden del día, con plazos de entrega aproximados, exigencias de disponibilidad para poder recoger el paquete y los propios repartos sujetos a la situación del tráfico y otros inconvenientes que limitan la eficiencia del sistema. Por otro lado, los drones debido a su tamaño, potencia y flexibilidad en cuanto a diseño y uso presentan una más que interesante vía de investigación para poder afrontar y solucionar todos estos desafíos de manera elegante y eficiente.

A pesar de que aún estemos algo lejos de que la entrega de paquetes mediante drones sea una realidad global y cotidiana Zieher et al. (2024), ya se están comenzando a presentar modelos sólidos que de manera individual van afrontando y proponiendo distintas soluciones a problemas de la cuestión. También se están llevando a cabo proyectos piloto que ponen a prueba algunos de estos modelos acercándonos a un futuro en el que todos estos trabajos culminen en una realidad que los ponga en práctica.

Por todo lo antes nombrado, consideramos interesante el estudio de la cuestión y cómo abordarla tratando de proponer una solución razonable e interesante que aborde los problemas principales de la misma. Además, la simulación es un vehículo muy interesante para enfrentar todas estas cuestiones de una manera sencilla, práctica y que permite abstraer el problema principal de cara a solucionar un problema complejo de una manera sencilla.

1.2. Objetivos

El objetivo principal del trabajo es desarrollar simulaciones de entrega de paquetes con drones que permitan analizar la viabilidad del uso de esta tecnología en entornos urbanos.

De manera específica, los objetivos son:

- Diseñar un modelo de simulación que represente de forma realista un entorno urbano, obstáculos, etc.
- Implementar algoritmos de optimización que minimicen el tiempo total, el consumo energético o la distancia recorrida.
- Analizar el impacto de distintas variables como el número de puntos de ruta, pesos de los pedidos, etc.
- Evaluar la viabilidad técnica y logística del reparto.

Este trabajo busca aportar una base que sirva de apoyo en la toma de decisiones para futuras aplicaciones basadas en el reparto de paquetes con drones.

1.3. Plan de trabajo

El desarrollo del trabajo se llevará a cabo a través de diferentes fases:

- Investigación y lectura de literatura científica: para conocer el estado de la cuestión y plantear objetivos razonables y de interés. Conocer otras aplicaciones relacionadas a través de la lectura de artículos y otras fuentes para extraer información interesante que pudiese tener aplicación en nuestros modelos.

- Búsqueda de modelos de drones adecuados: en la fase inicial se llevará a cabo una revisión de los modelos de drones disponibles en el mercado actual para tomarlos de ejemplo a la hora de hacer nuestras simulaciones.

- Selección de la herramienta de simulación: una vez definidos los modelos, se evaluarán distintas plataformas de simulación y se escogerá la que mejor se ajuste a los requisitos del proyecto, en nuestro caso, el programa QGroundControl.

- Conexión con el dron y ajuste de parámetros: se configurará el dron virtual en el software seleccionado y se ajustarán sus parámetros para que reproduzcan con fidelidad las especificaciones técnicas de los drones reales analizados.

- Diseño de rutas: una vez calibrado el modelo simulado, se diseñarán diversas rutas de entrega que servirán como casos de estudio para las pruebas de eficiencia y viabilidad.

- Obtención e interpretación de resultados: finalmente, se recopilarán los datos generados por las simulaciones y se realizará un análisis para evaluar la eficiencia de las entregas y la viabilidad de las rutas propuestas.

- Desarrollo de una interfaz: se desarrollará una interfaz que facilite la creación de rutas para usuarios no especializados. Simplificando la tarea de generar rutas, permitiendo definir de manera natural en un mapa los distintos puntos y los parámetros más importantes para la simulación.

Todo el código, *scripts* y documentación necesarios para reproducir las simulaciones se encuentran disponibles en GitHub en `https://github.com/jrubioczo/UltimaMillaTFG.git`

A continuación, como puede verse en la Figura 1.1 hemos realizado un diagrama de Gantt que refleja como ha sido el avance temporal del Trabajo de Fin de Grado (TFG) y como se han ido alcanzado los distintos objetivos.

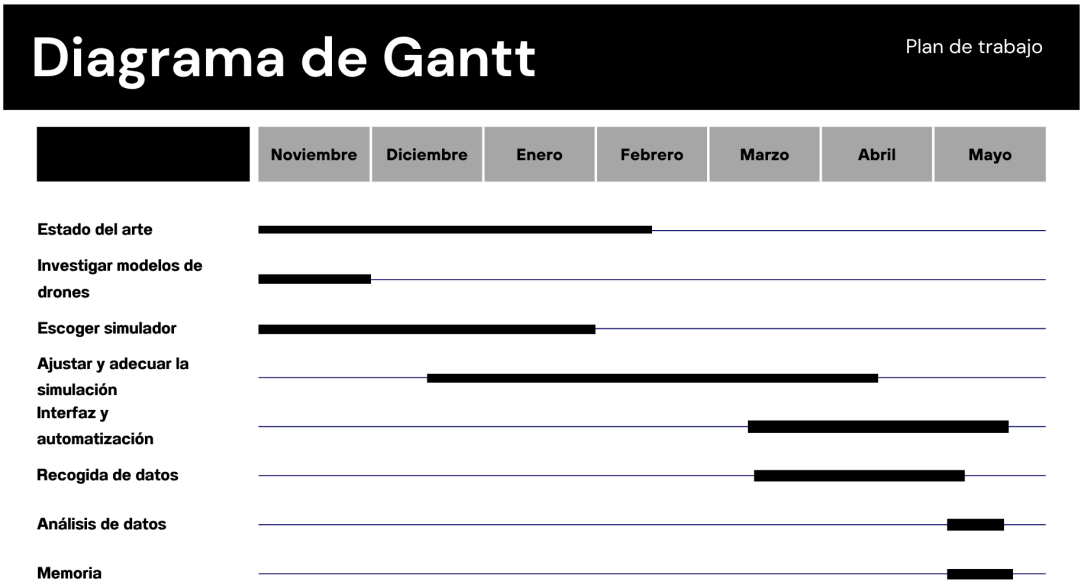

Figura 1.1: Diagrama de Gantt (Plan de trabajo)

1.4. Organización de la memoria

En esta memoria se aporta un breve resumen tanto en español como en inglés, seis capítulos (descritos a continuación), la bibliografía y las aportaciones personales de cada autor. Ahora se hará una breve introducción a lo que contendrá cada uno de los capítulos:

- Capítulo 1. Introducción: se expondrá una breve exposición de la motivación que respalda el proyecto. También se incluyen los objetivos iniciales que se plantearon. Además, cuenta con el plan de trabajo y un diagrama de Gantt que refleja cómo ha evolucionado el estado del TFG a lo largo del tiempo.
- Capítulo 2. Estado de la Cuestión: se hace una revisión de otros artículos e investigaciones que tratan temas afines y con impacto directo en nuestro campo de estudio. A su vez, se describen las distintas alternativas que se barajaron para llevar a cabo la simulación y la automatización. Indicando cuales fueron elegidas y porque. Habrá una breve descripción de modelos de drones competitivos ideales para los fines que se desean emplear.
- Capítulo 3. Diseño y configuración del entorno de simulación: se tratarán en detalle los pasos seguidos para llegar al estado final de la simulación conseguido. Se describirán los distintos pasos a realizar para llevar a cabo la simulación, indicando las decisiones concretas que tienen un impacto en la simulación y como repercuten en la misma. Describiendo un ejemplo de una para entenderlas en profundidad.
- Capítulo 4. Ejecución de pruebas y análisis de resultados: se describen las pruebas que se han hecho así como su fin para validar la simulación. Se extraen

datos de las distintas pruebas para poder arrojar luz sobre la validez del modelo y alcanzar conclusiones útiles mediante el análisis de las mismas.

- Capítulo 5. Interfaz y automatización: diseño de una interfaz para facilitar la tarea de realizar rutas a usuarios no expertos, permitiendo la modificación de algunos parámetros. Se explica su funcionamiento y utilidad. Además, se describe el proceso de automatismo para facilitar el lanzamiento de la simulación a usuarios sin conocimientos sobre el tema.
- Capítulo 6. Conclusiones y Trabajo Futuro: se realiza una retrospectiva del proyecto y de los resultados obtenidos. Se incluyen futuras líneas de investigación y su interés.

Capítulo 2

Estado de la Cuestión

En este capítulo discutiremos la situación del estado del arte, viendo las distintas aplicaciones del uso de drones no tripulados. Prestaremos especial atención a las propuestas para afrontar la cuestión de la entrega de última milla utilizando drones como elemento principal. También, veremos distintas opciones adecuadas para llevar a cabo la simulación de las misiones y modelos de drones adecuados para el fin que nos concierne en este estudio.

2.1. Drones, vehículos no tripulados

Los drones son un vehículo altamente flexible y práctico que ha ido ganando popularidad y mejorando su tecnología especialmente en la última década. Hoy en día los vemos a menudo usados con distintos fines. Quizá el más común y ligado a su origen sea el de vigilar áreas con facilidad gracias a las cámaras que en muchos casos llevan integradas, son ideales para estas tareas por su tamaño reducido y la autonomía que han logrado hoy en día.

Sin embargo, cada vez más se van encontrando otras aplicaciones y usos que difieren de su cometido original y que repercuten y benefician a la población de otras maneras. Como puede ser el caso de lo estudiado por (Wihbey, 2017), donde se citan variedad de usos entre los que destacan la recolección de datos, el mapeo de zonas, la vigilancia de áreas o la fotografía entre otros.

Los avances más importantes en la industria que han permitido que se popularicen hasta este punto, lleguen al público general y permeen en las casas de los entusiastas de la tecnología son la reducción de su precio y el aumento de la autonomía.

Esto ha sido posible gracias al aumento de su estudio y el abaratamiento de sus costes. También han contribuido las mejoras en las baterías, tanto en capacidad como en peso. El peso es el otro gran obstáculo que concierne a la autonomía de los drones, pues una reducción en el peso se traduce en un aumento de la autonomía de vuelo.

2.2. Aplicaciones

Hoy en día las aplicaciones y usos de los drones son cada vez más y más variados. Por ello, no podemos abordarlos todos. No obstante y debido a que muchos tienen amplia relación y aplicaciones directas si discutiremos la situación actual de algunos de los más relevantes y con mayor relación con nuestros objetivos.

2.2.1. Agricultura

En el sector de la agricultura los drones pueden presentar múltiples usos que reporten a las explotaciones múltiples ventajas y beneficios frente a la ausencia de los mismos. Existen diferentes propuestas referentes a distintas aplicaciones por ejemplo Bayas et al. (2021) describe una red inalámbrica entre drones que permita actuar con anticipación a los requerimientos que presenta el campo agrícola gracias a ser un sistema de tiempo real. Permitiendo entre otros activar el riego de zonas con un bajo nivel de humedad o fumigar zonas que presentan plagas. Otro caso de uso es el propuesto por González et al. (2016), similar al anterior pero donde destaca la prevención de situaciones climatológicas adversas como heladas o sequías u otros inconvenientes reduciendo la necesidad de personarse al poder valorar la situación sobrevolando la zona con drones.

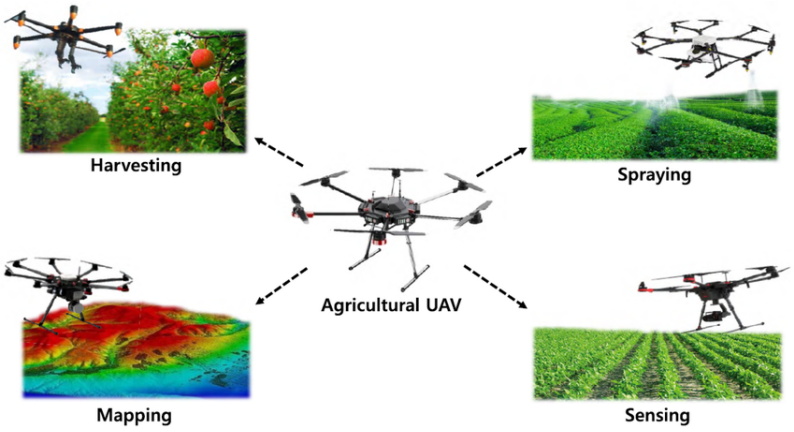

Figura 2.1: Aplicaciones de drones en agricultura. Fuente: Islam et al. (2021).

2.2.2. Zonas de desastre y rescates

Aunque aún no se utilicen drones con tanta asiduidad para el rescate de personas o para realizar trabajos en zonas de desastre, poco a poco se van incorporando a estas tareas. Además, este uso ha llamado la atención de los investigadores que cada vez invierten más recursos en estudiar aplicaciones para estas situaciones con expectativas muy positivas.

Por un lado, posibles aplicaciones de drones en zonas de desastre son las que se proponen por ejemplo en Daud et al. (2022). Donde se dividen en cuatro grandes grupos. Mapeo o gestión del desastre (la aplicación principal y que más aplicación real tiene en la actualidad), búsqueda y rescate, transporte y, por último, con fines

de entrenamiento.

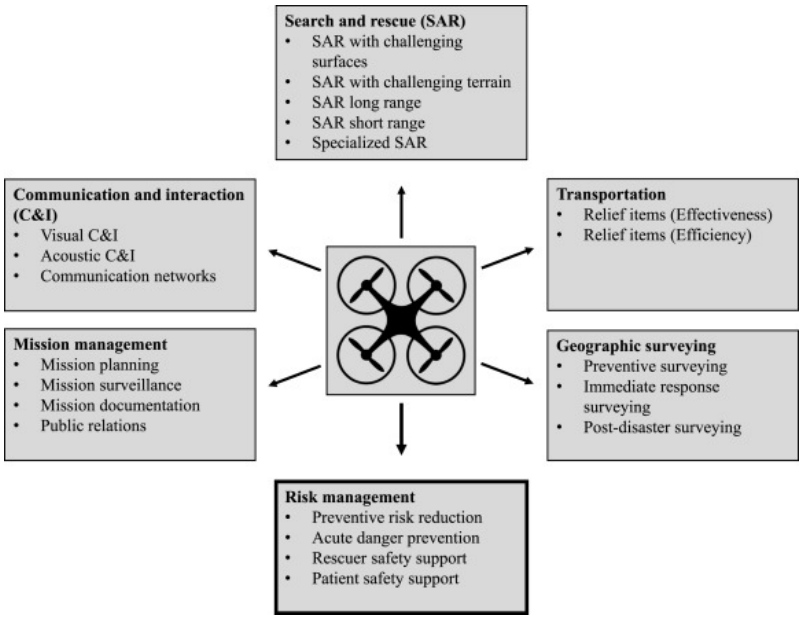

Figura 2.2: Aplicaciones de drones en zonas de desastre y rescates. Fuente: Wankmüller et al. (2021).

Por otro lado, también se plantean múltiples aplicaciones para el caso de rescates como caso particular del anterior. Puesto que muchas veces las situaciones de rescate también comprometen la vida de los rescatadores se pretende utilizar los drones para poder conocer mejor la situación y minimizar los riesgos. Ejemplo de esto es lo propuesto por Wankmüller et al. (2021) donde se discute el uso de drones en variedad de escenarios en situaciones críticas y con urgencias de tiempo. Siendo especialmente útiles en tareas de búsqueda y rescate, cumpliendo su cometido de reducir el riesgo de las operaciones.

2.2.3. Incendios y plagas

Otro caso con múltiples aplicaciones es el de prevención y lucha contra incendios y plagas. Con múltiples similitudes en aspectos como la vigilancia y la prevención. Pues permiten la monitorización de amplias zonas para vigilar y conocer la situación, permitiendo batidas óptimas que dan una idea rápida del estado y la situación del terreno, facilitando la rápida actuación y toma de decisiones. El caso de la lucha contra cada uno de los casos tiene más particularidades que hace que para cada uno se propongan distintas soluciones.

Para el caso de los incendios, es difícil actuar debido a lo errática que puede ser la expansión de las llamas y como las mismas pueden ir limitando la operación en las distintas bases de drones conforme aumenta el área afectada por el incendio. Una propuesta que busca solucionar este problema es la dada por (Akhloufi et al., 2021) donde se propone un sistema de UAS (por sus siglas en inglés Unmanned Aerial System). Proporcionando una alternativa económica para la prevención, detección y apoyo en tiempo real a la extinción de incendios.

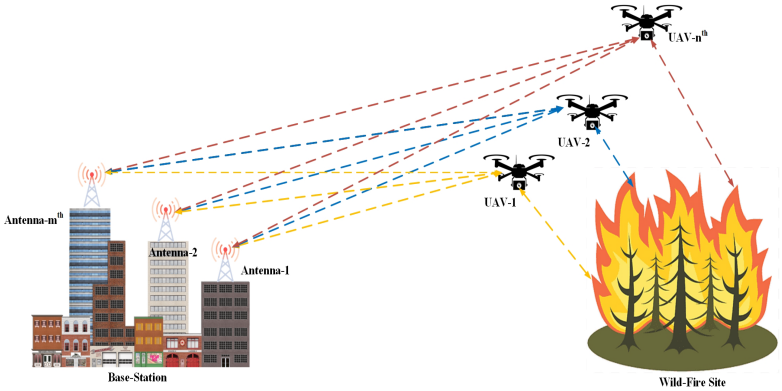

Figura 2.3: Uso de drones en incendios. Fuente: Ali et al. (2024).

En el caso de las plagas, la mayoría de tareas que se pueden realizar realmente están más relacionadas con la prevención. Aunque también se pueden tomar acciones valiéndose de drones para combatir la situación. Un ejemplo de estas medidas que se pueden tomar es el propuesto en Freitas et al. (2020), donde se propone el uso de drones para, aprovechando las propias regulaciones naturales que generan la cantidad de especiménes de ciertas especies frente a otras o barreras naturales como ríos, controlar ciertas poblaciones. Se propone un algoritmo para ubicar de manera eficiente estos enemigos naturales haciéndolo de la mejor manera posible, ya que en estudios anteriores se comprobaba que es vital el punto en el que se despliegan estos enemigos naturales para obtener los resultados deseados.

2.2.4. Logística de reparto con drones

Debido al auge de las entregas y el reparto a domicilio en los últimos años a causa principalmente del aumento de las compras online, se están estudiando nuevas alternativas destacando principalmente los drones. A continuación, se exponen distintas casuísiticas de uso, desafíos que se están afrontando y soluciones propuestas a los mismos.

2.2.4.1. Reparto de paquetes

El tema de la entrega de paquetes puede abordarse desde muchas perspectivas adoptando distintas soluciones y haciendo distintas asunciones. Una propuesta muy interesante es la dada por Brunner et al. (2019). El modelo que se plantea consiste en que el dron de entrega navega de forma autónoma a una ubicación sobre la casa del destinatario mediante navegación basada en GPS. Esto es especialmente factible mientras el dron se desplace sobrevolando los tejados.

Una vez sobre la posición destino, el dron realiza un cambio de modo a una navegación basada en visión (a través de sensores, es decir, una cámara) y comienza el descenso buscando el balcón objetivo marcado por un marcador visual (ver Figura 2.4). Al detectarlo, el dron se aproxima al balcón y deja el paquete. La idea es que para salir realice la ruta inversa (salga por donde entró).

Figura 2.4: Marcador visual en balcón. Fuente: Brunner et al. (2019).

Esto requiere coordinación por parte del cliente debido a que debe marcar en este caso el balcón (ubicación destino) con un marcador visual. También debe aportar las coordenadas GPS. No obstante, en el caso de una infraestructura grande si fuese necesario marcar la ubicación debería ser necesario poder distinguir entre marcas pues varios vecinos pueden haber solicitado paquetes, así mismo la ubicación GPS con herramientas adecuadas no debería ser difícil de adquirir.

La detección de marcadores visuales puede requerir de hardware específico en función de los algoritmos que se quieran usar para la navegación visual. Para el último paso de acceder al balcón del mismo modo que para descender de manera segura se debe contar con un algoritmo dedicado para la esquiva de obstáculos.

2.2.4.2. Reparto de comida a domicilio

Un caso particular de reparto mucho más exigente, es el de reparto de comida a domicilio, pues incluye un factor crítico, el del tiempo de entrega. Un ejemplo en el que se mezcla el reparto tradicional con el uso de drones y que aborda los intervalos de tiempo como parte de su solución es el propuesto por Lu et al. (2024). Partiendo como ejemplo de un caso particular de comida para llevar, podemos asumir que hay múltiples comercios en una ciudad y los diferentes grupos comerciantes cercanos pueden entenderse como centros de distribución debido a la proximidad entre sí. Sabiendo que la eficiencia de entrega de los repartidores se ve afectada por la distancia de entrega, la distancia euclídea se utiliza para agrupar los pedidos en los distintos centros de distribución (almacenes), reduciendo las distancias al asignar los destinos a los almacenes más próximos.

Para los pedidos en que la ubicación del cliente está fuera del rango de entrega del centro de distribución, en caso de un sistema híbrido (uso de drones y vehículos tradicionales como, por ejemplo, una furgoneta) los drones permiten alcanzar dicha área y satisfacer al cliente (ver Figura 2.5).

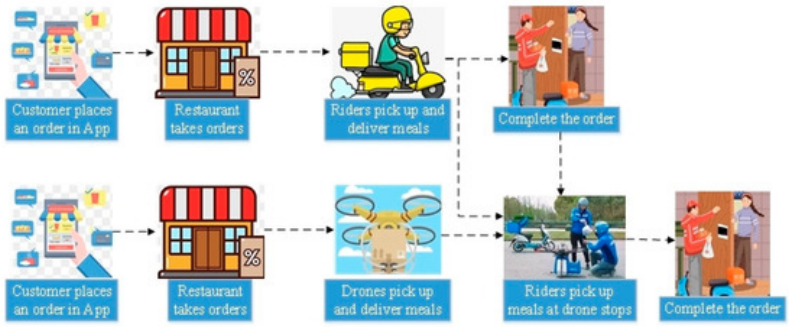

Figura 2.5: Modelo híbrido de reparto. Fuente: Lu et al. (2024).

En este modelo se hacen algunas suposiciones. La capacidad es la misma y no hay restricción en la distancia máxima, independientemente de la autonomía del vehículo. El tiempo de servicio es finito, es decir, se tiene en cuenta que el cliente puede disponer de un tiempo concreto para recoger la mercancía. Se asume una relación uno a uno entre los puntos de envío y los clientes. Varios clientes que realizan pedidos en la misma área se asignan al mismo punto de envío. Si un mismo cliente realiza pedidos a varios comerciantes, se establecerá como múltiples clientes con las mismas coordenadas. Los drones y otros vehículos vuelan a velocidad constante. En el proceso de entrega no se tienen en cuenta las condiciones meteorológicas, accidentes u otras circunstancias. Para el modelo se ha tenido en cuenta una curva de satisfacción del cliente y el consumo de energía del vehículo.

La aproximación a la solución del problema se plantea con un algoritmo heurístico dividido en dos etapas. En la primera etapa se tiene en cuenta la distancia entre el punto de envío y el de entrega, para esto se utiliza la distancia euclídea para asignar la región (punto de envío) al que pertenecen. En la segunda etapa, debido a que el algoritmo de búsqueda tabú tradicional es sensible al valor inicial, la capacidad para alcanzar una solución global es mala por lo que solo podemos buscar la solución óptima local. Las rutas iniciales de recogida y entrega se generan mediante el método de clasificación, y varios datos locales que están diseñados para optimizar las rutas de recogida y entrega incorporando la idea de cambiar el conjunto de estos datos locales en búsquedas variables y algoritmos para ampliar el rango de búsqueda y obtener la solución óptima.

En la primera etapa, se usa el algoritmo de agrupamiento AP (propagación por afinidad). La idea básica es ver los datos como nodos en una red y seguir modificando el número y la posición de los centros de agrupamiento pasando mensajes entre puntos de datos hasta que se maximiza la similitud de todo el conjunto de datos. Al mismo tiempo, se generan centros de agrupamiento altos y los puntos restantes se asignan a los grupos correspondientes. Dado que el algoritmo no necesita formular el número de grupos finales, múltiples ejecuciones del algoritmo dan exactamente los mismos resultados y no hay ningún requisito de simetría de los datos de la matriz de similitud inicial. Los resultados se caracterizan por un pequeño error cuadrático, robustez y precisión. Por lo tanto, el algoritmo de agrupación AP se utiliza para la asignación de pedidos.

En la segunda etapa, se usa el algoritmo de búsqueda tabú mejorado, este es un algoritmo metaheurístico basado en la búsqueda de vecindarios locales. Su idea básí-

ca es almacenar el proceso de búsqueda histórico reciente en la tabla tabú durante el proceso de búsqueda, para evitar que el algoritmo ingrese repetidamente las mismas regiones, a fin de prevenir el ciclo en el proceso de búsqueda y así obtener la solución óptima. El algoritmo de búsqueda tabú tradicional es sensible al valor inicial por lo que sólo puede buscar la solución óptima local. Por lo que en este artículo, en el proceso de entrega de comida para llevar, se mejora el algoritmo de búsqueda tabú tradicional en términos de generar la solución inicial, la estructura de vecindad y la longitud tabú.

2.2.4.3. Reparto teniendo en cuenta dinámicas variables

A la hora de proponer modelos, las distintas investigaciones se centran en algunos aspectos mientras ignoran otros para poder manejar la complejidad del campo de estudio. Uno de los puntos mayormente ignorados para el caso del reparto con drones suele ser el clima. Por eso es tán interesante la propuesta de Khanda et al. (2021).

Debido a las capacidades limitadas de la batería y que los drones pueden atender a un solo cliente a la vez, un sistema de entrega basado en drones (DBDS), tiene como objetivo minimizar el uso de energía de los drones para completar una ruta (ida y vuelta) para poder realizar nuevas entregas.

En general, la ruta de entrega más corta no tiene porque ser la elección óptima, ya que factores externos como el viento (que varía con el tiempo) puede afectar el consumo de energía. En este artículo se desarrolla un sistema centralizado para calcular rutas energéticamente eficientes y variables en el tiempo para drones en un sistema de entrega de múltiples depósitos y múltiples drones (ver Figura 2.6).

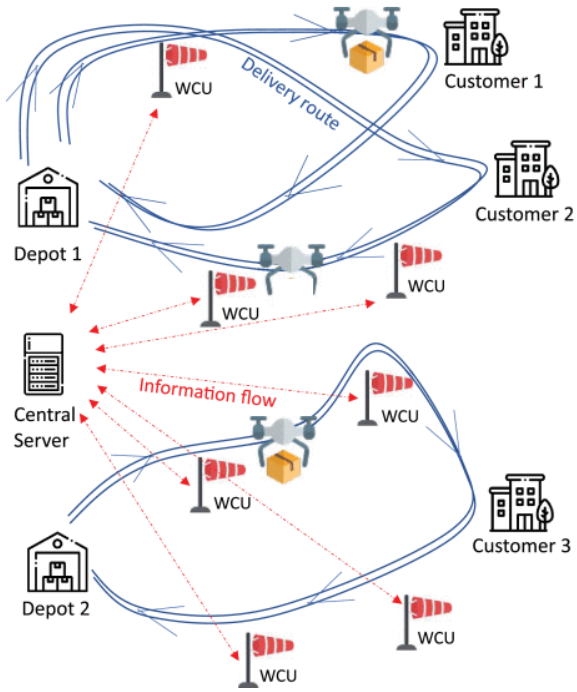

Figura 2.6: Reparto teniendo en cuenta variables dinámicas, en este caso el viento. Fuente: Khanda et al. (2021).

Específicamente, se propone un novedoso algoritmo llamado Parallel Shortest Route Update (PSRU) que, con el tiempo, actualiza la rutas de entrega de drones evitando muchos re-cálculos. El PSRU es 4,5 veces más rápido que los algoritmos de última generación.

Normalmente, para el cálculo de rutas se tienen en cuenta parámetros estáticos como la velocidad de vuelo, la carga máxima o la batería, que en esencia hacen referencia a lo mismo. Sin embargo, parámetros como la dirección o la velocidad del viento pueden influir notablemente en la velocidad reduciendo o aumentando también el coste del desplazamiento. Por ello se propone que el propio dron calcule mientras se desplaza la ruta teniendo en cuenta estas variables. Esto, por su parte, también supone un aumento en el gasto de batería. Lo cual se traduce en un descenso de la autonomía.

El problema puede representarse como un grafo en el que los vértices son los almacenes y los clientes; y las aristas los caminos con un coste asignado, para representar está variación del coste se propone utilizar grafos temporales. Por último, se propone manejar todas estas rutas desde un sistema centralizado que pueda aprovechar las rutas calculadas en snapshots anteriores.

Las principales aportaciones de este artículo son las siguientes: un DBDS centralizado que se encarga de calcular rutas de entrega eficientes para los drones cuando dinámicas externas afectan al consumo de energía del dron (estas rutas luego se transmiten a los drones), un novedoso algoritmo llamado PSRU que determina la ruta de entrega más corta con los gráficos temporales y la implementación de PSRU usando la arquitectura de NVIDIA GPU y demostrar su eficacia y eficiencia en comparación con técnicas de última generación.

Se asume que, todos los pedidos y las ubicaciones de los clientes se conocen de antemano, se conocen todos los caminos entre un almacén y un cliente, todos los drones son idénticos, se supone que un dron sirve a un solo cliente a la vez, ya que puede transportar un solo artículo, un dron regresa a su punto de partida después de realizada la entrega y todos los drones están completamente cargados cuando comienzan a realizar una entrega y solo pueden recargarse o cambiar la batería después de regresar al almacén.

2.3. Simuladores

Para poder llevar a cabo de la mejor manera y con las máximas garantías las simulaciones de acuerdo a los elementos que hemos considerado claves en nuestra investigación hemos valorado y estudiado distintos simuladores y tenido en cuenta sus pros y sus contras, hasta finalmente elegir la mejor opción de acuerdo a nuestras necesidades. A continuación, expondremos dichos simuladores y sus características más destacables.

2.3.1. ArduPilot

Ardupilot (ArduPilot, 2025) es un sistema software de código abierto que soporta gran variedad de vehículos distintos tales como drones, helicópteros, vehículos te-

rrestres, barcos, submarinos, etc. En sí mismo es un compendio de otros programas, es decir, un paquete. Todos los simuladores posteriores que veremos funcionan con la simulación del hardware y del dispositivo manejada con ArduPilot por debajo. Es decir prestarán otros servicios e interfaces pero la simulación correrá a cargo de ArduPilot.

ArduPilot ofrece un conjunto de instrucciones comunes para el manejo de todos los vehículos y también conjuntos de instrucciones específicas para vehículos concretos, en nuestro caso nos interesan las relacionadas con los drones.

En resumen, ArduPilot es un sistema de control de vehículos autónomos. Con software para poder simular una gran variedad de vehículos distintos. Es desarrollado por una gran comunidad y es usado tanto en proyectos de aficionados como con fines profesionales.

2.3.2. Mission Planner

Mission Planner (Mission Planner, 2025) es un software de código abierto, especialmente conocido y usado para el manejo de drones (especialmente vuelos controlados en ArduPilot). Permite tanto el desarrollo de misiones reales como la simulación de las mismas.

Como simulador cuenta con una interfaz que permite de manera cómoda e integrada en la simulación realizar variedad de acciones que simplifican la planificación de las misiones y el aumento en complejidad de las mismas, permitiendo planificar, monitorear y analizar las misiones mientras se realizan.

Utilizando Mission Planner para la simulación se pueden planificar misiones (crear rutas de vuelo mediante puntos de navegación e indicar acciones en dicho puntos), configurar el dron (ajustar parámetros del firmware de ArduPilot y calibrar sensores), monitorizar la misión en tiempo real (pudiendo conocer la información del dron, es decir, su telemetría y pudiendo conocer el estado del sistema) y descargar y analizar datos del vuelo (accediendo a los archivos de log de los vuelos).

2.3.3. QGroundControl

Finalmente, se ha utilizado QGroundControl (QGroundControl, 2025). Este actúa como una estación de control terrestre y permite la planificación, supervisión y gestión de misiones autónomas con drones compatibles con los protocolos MAVLink, como PX4 o ArduPilot.

Este programa ofrece una interfaz gráfica intuitiva, herramientas de configuración detalladas y una integración sencilla con simuladores.

El motivo de la elección de este programa en lugar del Mission Planner ha sido principalmente por la facilidad de modificar parámetros. Con este programa, usando una integración de Linux, conseguimos modificar los archivos que contienen las configuraciones de los drones, y luego podemos conectarlos al QGroundControl para ver cómo se comportan los vuelos en nuestras rutas diseñadas.

2.4. Modelos de drones

Con la finalidad de elegir un dron adecuado y competente para las tareas que se esperan desempeñar se ha realizado un estudio de los drones del mercado. Prestando especial interés en los más competentes y sus características.

Las características más importantes extraídas son la batería y la capacidad de carga (peso capaz de manejar). Siendo vital la autonomía de los drones para delimitar el área de reparto entorno a la base de operaciones y la carga máxima que puedan soportar para poder garantizar que los paquetes lleguen a su destino en óptimas condiciones.

De acuerdo a estos parámetros existen dos tipos de drones. Drones de carga ligera y drones de carga pesada. Por un lado, los drones de carga ligera pueden cargar objetos de entre 1 Kg y 5 Kg. En general tienen una autonomía de 30 minutos aunque algunos modelos son capaces de superar la hora de vuelo. Por otro lado, los drones de carga pesada pueden llegar a cargar objetos de hasta 10 Kg, modelos específicos pensados para actuar en incendios o contra plagas pueden llegar a cargar objetos de más de 15 Kg y esto con una autonomía de 80 minutos.

Teniendo esto en cuenta los modelos que más se adecuan en nuestro caso son los drones de carga ligera. Siendo algunos modelos que ejemplifican lo dicho y que en la actualidad se utilizan para el proposito descrito los siguientes:

Modelo	**Carga útil máx.**	**Duración de vuelo**	**Distancia** (sin carga)	**Alcance** (con carga máx.)
DJI FlyCart 30	30 kg	18 min	28 km	16 km
Wingcopter 178	6 kg	85 min	100 km	100 km
Wingcopter 178 HL	6 kg	120 min	100 km	100 km
Matternet M2	2 kg	30 min	20 km	20 km
DJI Matrice 600 PRO	6 kg	38 min	5 km	5 km

Tabla 2.1: Comparativa de capacidad de carga, autonomía y rango.

Modelo	**Velocidad máx.**	**Altura máx.**	**Precio**
DJI FlyCart 30	72 km/h	6000 m	19 599 €
Wingcopter 178	150 km/h	3000 m	–
Wingcopter 178 HL	150 km/h	5000 m	–
Matternet M2	58 km/h	121 m	7 500 €
DJI Matrice 600 PRO	65 km/h	4500 m	5 990 €

Tabla 2.2: Velocidad, techo de servicio y precio de los drones analizados.

Nota: la información sobre los modelos en mucho casos puede ser escasa. Se aporta la más relevante obtenida de modelos competitivos.

De acuerdo a los datos mostrados podemos deducir que quizá el modelo que por especificaciones y precio más se aproxima a lo que estamos buscando sea el

modelo **DJI Matrice 600 PRO** (ver Figura 2.7). Debido a que encaja con la autonomía y los pesos esperados y se encuentra en un rango de precios inferior al de sus competidores. No obstante, otra característica que podría tenerse en cuenta a la cual no le hemos dado tanta importancia podría ser la estabilidad del dron.

Figura 2.7: Dron DJI Matrice 600 PRO. Fuente: DJI (2025).

2.5. Herramientas de desarrollo de la interfaz

De cara a diseñar y desarrollar la interfaz se valoraron distintas alternativas. Se valoraron de acuerdo al resultado final esperado, las facilidades que aportarían de cara al desarrollo y su adecuación a las exigencias de la aplicación. Seguidamente, detallaremos dichas opciones y el porqué de la decisión final.

2.5.1. Android Studio

Android Studio (Android Studio, 2025) es el IDE oficial para desarrollar apps para Android, cuenta con el respaldo de Google. Android Studio brinda un amplio conjunto de herramientas que permiten diseñar, programar, probar y depurar aplicaciones móviles. Además, incluye emuladores de dispositivos, ideales para probar distintos modelos y configuraciones con facilidad.

Una de las ideas que se contempló, fue desarrollar una aplicación nativa para Android desde la que poder configurar cómodamente las rutas. Tenía sentido porque Android Studio permite integración con modelos como los DJI entre otros que ofrecen SDKs para Android. Además, permitiría su uso desde móviles aportando portabilidad y facilidad en su uso gracias a la pantalla táctil.

2.5.2. Aplicación web

Realizar una aplicación web conllevaba bastantes ventajas, pues permitiría el acceso de distintos usuarios desde variedad de dispositivos. Permitiendo realizar todas las acciones deseadas y haciéndolo muy portable y flexible gracias a la accesibilidad multiplataforma. Otra ventaja era la compatibilidad con plataformas de control remoto de drones, pero al no utilizarlas no era especialmente interesante. También, aportaba mucha facilidad para integrar e interactuar con mapas en tiempo real. Uti-

lizando alguna de las múltiples bibliotecas y APIs disponibles como Google Maps o Mapbox.

2.5.3. Unity

Unity (Unity, 2025) es un motor 3D y 2D multiplataforma muy utilizado para crear videojuegos, simuladores y visualizaciones interactivas entre otros. Su potencia, flexibilidad, y set de herramientas lo hacen ser una muy buena opción para diseñar interfaces complejas. Como puede ser el caso que nos concierne, una aplicación para la planificación de rutas de vuelo de drones.

Finalmente, esta ha sido la opción elegida. Los motivos principales han sido su facilidad para exportar la aplicación a diversidad de plataformas, permitiendo todo lo que las opciones anteriores en ese sentido. Permite gran interacción con el usuario pues cuenta con un desarrollado kit de herramientas de UI (User Interface) y también cuenta con un gran ecosistema de herramientas y una comunidad grande. Esto último es especialmente importante ya que facilita el desarrollo que haya una gran cantidad de ejemplos y de usuarios que respaldan la plataforma.

2.6. Herramientas usadas por la interfaz

Para facilitar la tarea de desarrollar la interfaz, una vez decidido el entorno de desarrollo (Unity). Se contemplaron distintas herramientas que permitiesen facilitar la tarea del desarrollo de la aplicación en sí misma. El mayor desafío a afrontar en este aspecto es la representación del mapa y su interacción con el mismo. Por ello se valoraron distintas opciones recogidas a continuación.

2.6.1. API Google Maps

Para poder mostrar un mapa y configurar las misiones adecuadamente se valoró el uso de la API de Google Maps (Google Maps Platform, 2025). Una API muy completa y robusta que permitía realizar todas las acciones deseadas. La API de Google Maps facilita un conjunto de herramientas y servicios proporcionados por Google que permiten integrar las distintas funcionalidades de sus mapas con distintos tipos de aplicaciones. De modo, que a través de la API los desarrolladores pueden mostrar sus mapas configurables y obtener información específica de los mismos.

No obstante, no presentaba especialmente buena compatibilidad con Unity y conseguir una key actualmente no es tan sencillo ni cómodo como en la alternativa Mapbox descrita a continuación.

2.6.2. Mapbox

Mapbox (Mapbox, 2025) es un proveedor de mapas en línea y una herramienta de mapeo altamente personalizable. Permite integrar mapas, datos geoespaciales y visualizaciones 3D en aplicaciones. Utiliza OpenStreetMap (OSM), un proyecto colaborativo de cartografía donde voluntarios crean y mantienen mapas libres y

editables de todo el mundo. Permite fácilmente tener distintas keys con distintos permisos para distintos propósitos, muy útil para tener control sobre el acceso y tener keys concretas para usos concretos. Además, Mapbox destaca en su flexibilidad para la personalización visual, ofreciendo variedad de estilos de visualización.

Finalmente, se escogió está opción. Por un lado, permite una fácil integración gracias a Mapbox Unity SDK, una SDK que se instala como un paquete que aporta una estructura sólida con ejemplos variados para desarrollar directamente desde Unity. Por otro lado, cuenta con una alta precisión geoespacial, interactividad en tiempo real y una buena escalabilidad y rendimiento. También, es una gran opción por sus excelentes mapas interactivos y responsivos, ideales para tareas como la que nos ocupa. Cuenta con una excelente documentación y una comunidad muy activa.

Capítulo 3

Diseño y configuración del entorno de simulación

Este capítulo se centra en la construcción práctica del entorno que permitirá poner a prueba las ideas expuestas. En primer lugar se justifica la elección de QGroundControl como plataforma de referencia; a continuación se describe, cómo se conecta y configura el dron simulado para reproducir las especificaciones de un modelo en concreto y finalmente se detalla el proceso de planificación de misiones y rutas, parte fundamental para posteriormente ser analizadas y estudiar la viablidad de las entregas con drones en los escenarios definidos.

3.1. Herramienta de simulación: QGroundControl

Para llevar a cabo la simulación de las misiones de entrega, se ha seleccionado QGroundControl como herramienta principal de planificación y control de vuelo. Este programa de código abierto, se ha convertido en una de las más utilizadas en el ámbito de los vehículos aéreos no tripulados (UAVs), tanto en entornos reales como simulados. QGroundControl ofrece compatibilidad total con los principales firmwares de drones como PX4 y ArduPilot, y permite tanto el control manual como la planificación de misiones autónomas mediante una interfaz gráfica intuitiva.

La elección de este programa se ha basado en diversos factores:

- Facilidad de uso: su interfaz es clara y fácil a la hora de crear rutas hasta para un usuario no experto.
- Compatibilidad: permite operar con drones simulados sin la necesidad de tener uno físico.
- Configurabilidad: ofrece ajustar de manera sencilla los parametros tanto del dron como de las propias rutas (alturas, pesos, batería, etc.)
- Monitorización en tiempo real: el programa permite observar en todo momento el estado del dron y registrar sus logs para posteriormente ser analizados.

Dado que el desarrollo del trabajo se ha realizado en un entorno Windows, fue necesario configurar un subsistema Linux para poder ejecutar correctamente el simulador y facilitar la comunicación con QGroundControl. Para ello, se utilizó WSL (Windows Subsystem for Linux), una herramienta que permite ejecutar una distribución de Linux directamente en Windows sin necesidad de máquinas virtuales.

La conexión entre el simulador y QGroundControl se estableció mediante el protocolo UDP, lo cual es posible gracias a que QGroundControl está preparado para detectar automáticamente drones simulados que se comuniquen por el puerto adecuado (generalmente el 14550). De esta forma, al ejecutar el simulador desde WSL, QGroundControl en Windows detecta la presencia del dron y establece la conexión sin necesidad de configuración adicional.

Aunque fue un paso clave para el proyecto, la configuración inicial presentó algunas dificultades técnicas relacionadas con la integración de WSL en Windows, en particular al establecer la comunicación de red entre ambos entornos.

3.2. Configuración y conexión con el dron simulado

Una vez establecida la conexión entre el dron simulado y el software QGroundControl, se procedió a la personalización de los parámetros físicos y energéticos del modelo virtual, con el objetivo de reflejar de forma realista el comportamiento de un dron comercial en operaciones de reparto de última milla.

3.2.1. Personalización del dron simulado

Para adaptar el comportamiento del dron virtual a las características de un modelo real utilizado en reparto de última milla, fue necesario modificar directamente los parámetros físicos y eléctricos del simulador.

Esta personalización se realizó sobre el archivo fuente SIM_Multicopter.cpp del entorno SITL (Software In The Loop) de ArduPilot, ubicado dentro del código fuente del simulador.

Tras cada modificación realizada en el archivo SIM_Multicopter.cpp, como la que se puede ver en la Figura 3.1 (donde se modifica mass), fue necesario recompilar el entorno de simulación para que los nuevos parámetros físicos y energéticos se aplicaran correctamente en el modelo del dron. Para ello, se utilizó la herramienta de construcción waf, que es el sistema de compilación utilizado por ArduPilot.

```
MultiCopter::MultiCopter(const char *frame_str) :
    Aircraft(frame_str)
{
    frame = Frame::find_frame(frame_str);
    if (frame == nullptr) {
        printf("Frame '%s' not found", frame_str);
        exit(1);
    }

    frame->init(frame_str, &battery);

    mass = frame->get_mass();
    mass = 6.0f;
    frame_height = 0.1;
    ground_behavior = GROUND_BEHAVIOR_NO_MOVEMENT;
    lock_step_scheduled = true;
}
```

Figura 3.1: Fragmento de código de SIM_Multicopter.cpp donde se modifica la masa.

El comando `./waf copter` recompila únicamente el binario correspondiente al modelo ArduCopter, asegurando así que los cambios introducidos en el código fuente se reflejen en la próxima simulación. Una vez completada la compilación, se podía lanzar el simulador utilizando el *script* simvehicle.py, como se describirá próximamente.

El archivo SIM_Multicopter.cpp define los valores por defecto que el dron simulado utiliza durante la ejecución, como:

- model.mass: Masa total del dron, incluyendo la carga útil.
- model.refVoltage y model.maxVoltage: Voltaje nominal y máximo del sistema de baterías.
- model.battCapacityAh: Capacidad total de la batería en amperios-hora.
- model.refCurrent: Corriente de referencia para vuelo estacionario (hover).
- powerfactor: Coeficiente que relaciona empuje con potencia consumida.
- model.spinmax y model.pwmMax: Configuración de motores y señal máxima de control (PWM).

En nuestro caso el modelo elegido fue el DJI Matrice 600 PRO, por lo que ajustamos los parámetros más significativos en relación a éste. A continuación se muestran la autonomía aproximada del dron con diferentes pesos y las velocidad a las que se ha movido en las simulaciones (ver Tabla 3.1 y 3.2).

Carga	**6 kg**	**4 kg**	**2 kg**
Distancia (km)	11	14	18
Duración (min)	16	21	27

Tabla 3.1: Relación entre la carga transportada y el rendimiento del dron en términos de distancia y duración del vuelo.

Parámetro	**Valor**
Velocidad por defecto	10 m/s
Velocidad máxima	14 m/s
Velocidad de ascenso	2 m/s
Velocidad de descenso	-2.5 m/s

Tabla 3.2: Velocidades en distintas fases de vuelo.

3.2.2. Ejecución de la simulación

Una vez personalizados los parámetros del dron en el entorno SITL, se procede a ejecutar la simulación mediante el siguiente comando:

```
(venv-ardupilot) jrubiozoz@DESKTOP-3U73KUR:~$ \
  ../Tools/autotest/sim_vehicle.py
  -v ArduCopter -I0 --sysid 1 \
  --map --console --out=udp:127.0.0.1:14550 \
  --custom-location=40.415363,-3.707398,650,30
```

Este comando lanza una instancia del simulador ArduCopter e inicia automáticamente la transmisión de datos de vuelo a través de UDP en el puerto 14550, lo que permite que QGroundControl se conecte al dron simulado de forma automática si se encuentra en ejecución.

A continuación, una breve explicación de los argumentos principales utilizados:

- `-v ArduCopter`: selecciona el tipo de dron a simular.
- `-I0`: identifica la instancia del dron (útil si se simulan varios).
- `-sysid 1`: establece el identificador del sistema MAVLink.
- `-map` y `-console`: habilitan la visualización del mapa y la consola.
- `-out=udp:127.0.0.1:14550`: define el canal de salida hacia QGroundControl.
- `-custom-location=40.415363,-3.707398,650,30`: fija la posición GPS inicial del dron en la Plaza Mayor de Madrid, con altitud 650 m y rumbo 30°.

Una vez ejecutado el comando, el dron simulado aparece automáticamente en el mapa de QGroundControl. Desde ahí, es posible monitorizar en tiempo real parámetros como altitud, velocidad, nivel de batería o modo de vuelo. Además, se

puede iniciar una misión de prueba, simular un despegue automático o cargar rutas previamente definidas mediante *waypoints*.

Opcionalmente, si se desea introducir condiciones meteorológicas en la simulación, como viento constante o ráfagas, es posible utilizar la consola MAVProxy conectado al simulador y aplicar comandos como `wind 5 90`, lo cual permite observar el comportamiento del dron ante, en este caso, un viento de 5 m/s que sopla desde el este hacia el oeste.

3.3. Planificación de misiones y rutas

Una vez conectado el dron simulado y ajustados sus parámetros físicos, el siguiente paso era en planificar las rutas de vuelo que permitirían evaluar su comportamiento en distintos escenarios de entrega.

El objetivo principal de esta etapa fue definir trayectorias que simularan casos reales de reparto en entornos urbanos, variando la longitud de las misiones y la cantidad de carga transportada. Para ello, se diseñaron rutas clasificadas como cortas, medias y largas, con el fin de analizar cómo afectan la distancia y la duración del vuelo al rendimiento del dron.

3.3.1. Creación de rutas

Para la planificación de misiones, se optó por utilizar el sistema de generación de rutas basado en *waypoints* que proporciona QGroundControl, aunque no de forma manual. En lugar de crear cada misión directamente desde la interfaz del programa, se desarrolló un *script* en Python encargado de generar archivos de misión con extensión .plan, formato estándar utilizado por QGroundControl para importar rutas.

Estos archivos .plan contienen información estructurada en formato JSON sobre los distintos puntos de la misión, como la latitud, longitud, altitud, tipo de acción en cada punto (navegación, espera, aterrizaje, etc.) y parámetros adicionales como la velocidad o el número de repeticiones de un tramo.

```
{
    "AMSLAltAboveTerrain": null,
    "Altitude": 0,
    "AltitudeMode": 3,
    "autoContinue": true,
    "command": 22,
    "doJumpId": 1,
    "frame": 10,
    "params": [
        0,
        0,
        0,
        0,
        40.415363,
        -3.707398,
        30
    ],
    "type": "SimpleItem"
},
```


Figura 3.2: Ejemplo de *waypoint* en formato JSON.

El uso del *script* permitió automatizar la creación de rutas personalizadas y repetibles para cada uno de los escenarios definidos: trayectos cortos, medios y largos.

Una vez generado el archivo .plan, el procedimiento consistía en abrir QGroundControl, seleccionar la opción "Cargar misión desde archivo", y elegir el fichero correspondiente. Al hacerlo, la ruta aparecía directamente sobre el mapa, con todos los puntos de control correctamente definidos y listos para su ejecución en el entorno simulado.

3.3.2. Tipología de misiones

Para evaluar el comportamiento del dron simulado en distintos contextos, se diseñaron varias misiones diferenciadas por su distancia y duración estimada, con el objetivo de simular escenarios reales de reparto en entornos urbanos. Estas misiones fueron clasificadas en tres categorías principales: corta, media y larga distancia, cada una de ellas representando un tipo de trayecto con características distintas.

La elección de esta clasificación responde a la necesidad de estudiar cómo afecta la duración del vuelo y la cantidad de carga transportada al rendimiento del dron, así como a la planificación en términos de autonomía, tiempo de entrega y eficiencia.

A continuación se describen las tres tipologías:

- Misión corta: trayectos de aproximadamente 1–2 km, representando entregas de barrio o dentro de una misma zona urbana. Este tipo de misión simula un servicio rápido, con bajo consumo energético y poca exigencia de planificación.
- Misión media: distancias entre 4 y 6 km, correspondientes a desplazamientos

entre barrios adyacentes. Aquí se busca observar un equilibrio entre tiempo de vuelo, consumo y efectividad de la ruta.

- Misión larga: vuelos de más de 8–10 km, simulando entregas entre distintas zonas de la ciudad o hacia periferias. Estas misiones ponen a prueba la autonomía del dron y su capacidad para mantener estabilidad durante trayectos prolongados.

Cada tipo de misión fue generada mediante el *script* en Python mencionado en el apartado anterior, permitiendo controlar la longitud de la ruta y el número de puntos de paso. Además, estas rutas fueron adaptadas al modelo de dron simulado (DJI Matrice 600 PRO), ajustando las altitudes de vuelo y velocidades previstas según sus especificaciones técnicas.

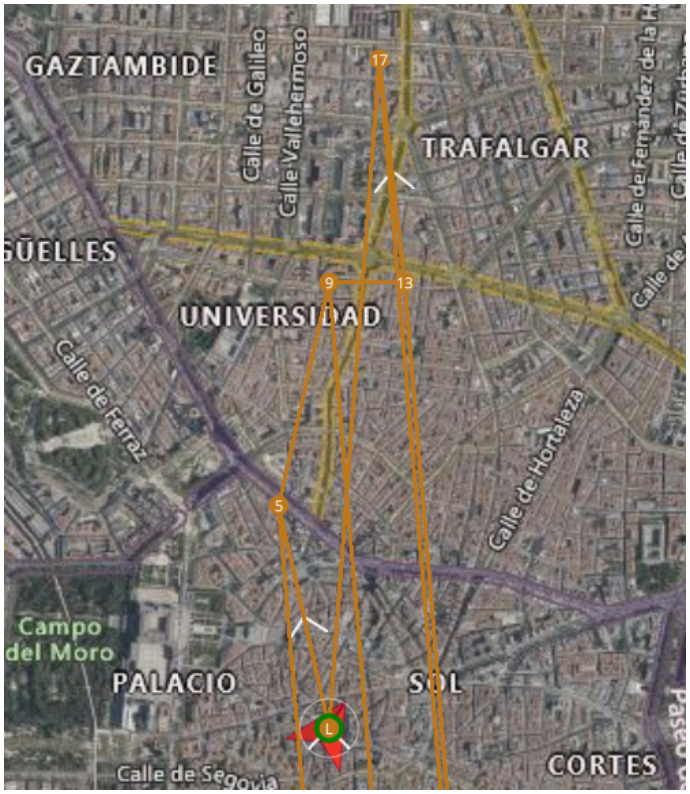

Figura 3.3: Ejemplo de ruta de entrega de distancia media.

3.3.3. Consideraciones operativas y restricciones

Además de definir la longitud y estructura general de las rutas, se incorporaron una serie de consideraciones operativas diseñadas para simular con mayor realismo un servicio de reparto de última milla con drones. Estas decisiones afectaron directamente a cómo se comporta el dron en cada misión y fueron clave para obtener resultados representativos del contexto urbano.

Una de las decisiones principales fue la inclusión de múltiples *waypoints* en las rutas, representando distintos puntos de entrega de paquetes a lo largo de un mismo trayecto. Las rutas generadas con el *script* en Python variaban en complejidad,

desde trayectos lineales con pocos puntos hasta recorridos con múltiples paradas, simulando misiones de tipo multientrega.

En cada uno de estos *waypoints*, el dron estaba programado para realizar la siguiente secuencia:

- Descender a una altitud de 0.5 metros, simulando la aproximación al punto de entrega.
- Permanecer en esa posición durante 30 segundos, simulando el tiempo necesario para dejar o recoger el paquete.
- Ascender nuevamente a la altitud establecida de la misión (normalmente entre 30 y 50 metros) para dirigirse al siguiente destino.

Esta lógica de comportamiento se define dentro del propio archivo .plan, configurando para cada *waypoint* los parámetros de altitud y una acción de espera (Loiter) con la duración correspondiente.

Otro aspecto considerado fue la gestión del retorno al punto de origen al finalizar la misión. En algunas rutas, el dron regresaba automáticamente al lugar de despegue tras completar todas las entregas, mientras que en otras misiones se definía un último *waypoint* como punto final alternativo, simulando escenarios de distribución por zonas donde el dron no vuelve inmediatamente a su base. Esta variación permitió comparar el impacto en el consumo energético y el tiempo total de operación.

Por último, se prestó especial atención a que las rutas generadas fueran coherentes desde el punto de vista operativo, evitando diseños poco realistas como trayectorias abruptas o acumulación excesiva de *waypoints* en zonas cercanas. Con ello, se procuró que las misiones reflejaran de manera realista el comportamiento de un dron operando en un entorno urbano.

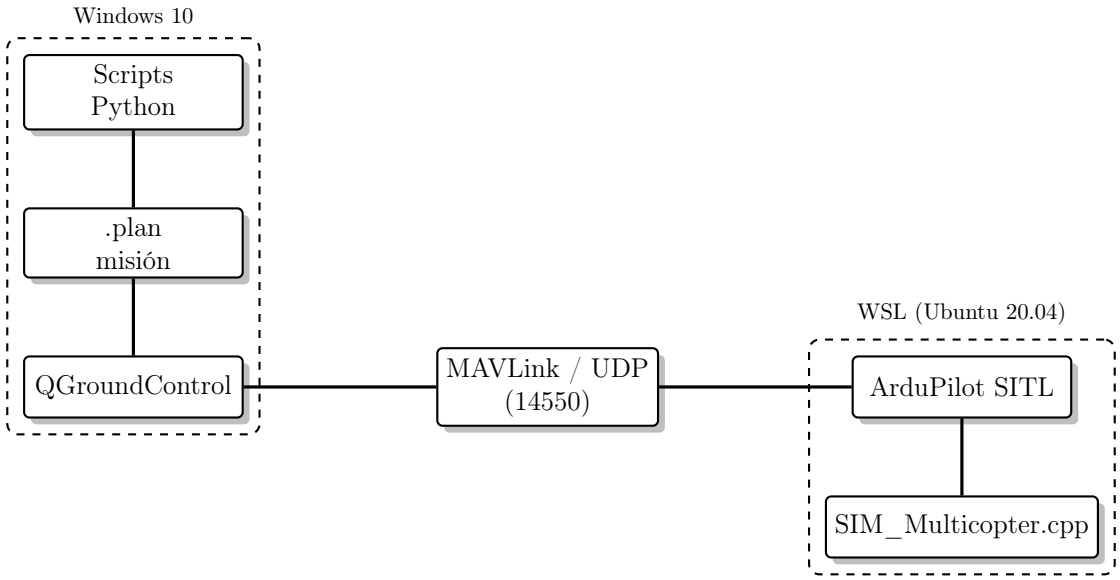

Figura 3.4: Flujo mixto de la arquitectura de simulación y control de drones.

3.4. Proceso operativo de la entrega de última milla con drones

Para resolver el reto de la entrega de última milla, nuestro sistema combina drones, micro-hubs urbanos y un planificador inteligente que optimiza cada paso del reparto. La idea es sencilla: agrupar los pedidos en nodos cercanos al cliente, calcular rutas que minimicen distancia y consumo de batería y, por último, dejar cada paquete con precisión en su destino. Con ello reducimos tiempos de entrega, evitamos tráfico terrestre y recortamos la huella de carbono, todo mientras mantenemos los costes bajo control. Para ello, el sistema seguirá los siguientes pasos:

3.4.1. Consolidación de pedidos y planificación de misión

Una vez cierra la tanda de pedidos, la plataforma arranca un proceso en cuatro pasos para estructurar las misiones:

- Agrupar los pedidos en el punto de salida adecuado: los paquetes se distribuyen entre los micro-hubs o furgonetas lanzadera situados en la periferia según tres criterios muy sencillos: que el peso y el tamaño no superen lo que puede llevar un dron y que la hora de reparto encaje con la ruta prevista.
- Calcular la mejor ruta para cada dron: para cada micro-hub, el planificador busca la combinación de paradas que recorra la menor distancia posible sin agotar la batería ni exceder la carga útil. En esencia, resuelve un problema de rutas con dos límites: peso y autonomía. El resultado es la lista ordenada de direcciones que debe seguir cada dron.
- Crear automáticamente el "plan de vuelo": esa lista de paradas se convierte en un fichero que el piloto automático entiende. Incluye el punto de despegue, la altura de crucero, cada destino y el regreso al punto de carga. Así, con solo cargar el archivo, el dron sabe exactamente adónde ir y en qué orden.
- Vincular cada paquete con su dron: finalmente, el sistema guarda qué paquete viaja en qué dron. De este modo, durante la operación se puede seguir cada envío en tiempo real y comprobar cuándo se deja en su destino.

Con esta secuencia sencilla queda preparada toda la operativa para que los drones salgan a reparto sin intervención manual adicional.

3.4.2. Preparación y carga en el micro-hub

Cada pedido se introduce en un contenedor de liberación rápida que queda bloqueado bajo el fuselaje. Para hacer esto tenemos dos opciones:

- Carga robotizada: un robot posiciona el contenedor y verifica, mediante célula de carga, que el peso coincide.
- Carga manual asistida: el operario engancha la caja, escanea un QR y cierra el pestillo de seguridad.

3.4.3. Pre-vuelo y ejecución de la misión

A través de QGroundControl se ejecuta la lista de comprobaciones, arma los motores y lanza el dron con la misión ya cargada. La telemetría se transmite vía MAVLink y se registra para auditoría. Cualquier incidencia durante el *arm* cancela la salida automáticamente.

El dron mantiene una velocidad de crucero de 10 m/s y una altura suficiente para cumplir la normativa local. Al alcanzar cada *waypoint* de entrega, inicia la maniobra correspondiente: descenso, estabilización y ejecución del mecanismo de descarga seleccionado.

3.4.4. Entrega en destino

Se contemplan tres modos, elegidos dinámicamente según la topología del destino:

Modo	**Descripción**	**Uso recomendado**
Aterrizaje + liberación *servo*	El dron posa sus patas, desbloquea el pestillo eléctrico y despega vacío.	Plazas de parking, azoteas amplias.
Winch / malacate	Desde 20–30 m, desciende el paquete con cable Kevlar y sensor de tensión, liberándolo al tocar la superficie.	Jardines, patios interiores, ausencia de plataforma de aterrizaje.
Descenso guiado a balcón	El dron permanece en estacionario mientras baja el paquete los últimos 2 m guiado por un marcador visual.	Balcones, ventanas con baliza instalada.

Tabla 3.3: Modos de entrega de paquetes y escenarios de uso recomendados.

Tras la liberación, el software capta una fotografía de prueba de entrega y envía la confirmación al cliente.

3.4.5. Post-entrega y retorno

Al salir del punto de entrega, el piloto automático actualiza la masa útil (masa total – masa del paquete), lo que permite recalcular la autonomía restante y decidir:

- Continuar hacia el siguiente destino si la energía es suficiente.
- Return to Launch (RTL) al micro-hub más cercano si se alcanza el umbral de batería mínima.

Los registros de vuelo y sus marcas temporales se guardan en la base de datos, de modo que cada misión pueda rastrearse y analizarse posteriormente.

Capítulo 4

Ejecución de pruebas y análisis de resultados

Con el entorno de simulación plenamente configurado y las misiones definidas, este capítulo aborda la viabilidad del sistema. En primer lugar se describen los procedimientos de ejecución de las pruebas y la estrategia de recogida de datos, prestando especial atención a las variables que permiten medir el rendimiento energético y operativo del dron. A continuación, se presentan y discuten los resultados obtenidos, comparando cómo afectan la distancia, la carga y los perfiles de ruta al tiempo de vuelo y al consumo de batería. Las conclusiones extraídas sirven de base tanto para evaluar la viabilidad de las entregas con drones en entornos urbanos como para plantear mejoras y líneas de trabajo futuro.

4.1. Ejecución de pruebas y recogida de datos

Una vez configurado el entorno de simulación, definido el modelo de dron y planificadas las misiones, se procedió a la fase de ejecución de pruebas. El objetivo de esta etapa fue obtener datos representativos del comportamiento del dron simulado bajo diferentes condiciones, con el fin de analizarlos posteriormente y extraer conclusiones sobre su eficiencia y viabilidad en tareas de reparto urbano.

4.1.1. Proceso de simulación y ejecución de cada misión

La ejecución de cada misión siguió un proceso que comenzaba con la configuración del entorno de simulación y finalizaba con la recogida de resultados tras completar el recorrido planificado. Cada prueba estaba diseñada para simular un escenario realista de reparto con drones, ajustando los parámetros del entorno y del modelo virtual para mantener la coherencia entre misiones.

En primer lugar, se lanzaba el entorno de simulación mediante el comando sim_vehicle.py, iniciando el modelo de ArduCopter con los parámetros físicos previamente definidos en el archivo SIM_Multicopter.cpp. Este comando abría la consola de simulación y establecía la comunicación vía UDP con QGroundControl.

Una vez abierta la interfaz de QGroundControl y confirmada la conexión automática con el dron simulado, se cargaba la misión correspondiente mediante la opción "Cargar misión desde archivo". Los archivos .plan utilizados habían sido generados previamente a través de un *script* en Python, y contenían los *waypoints*, altitudes, velocidades y acciones específicas definidas para cada tipo de prueba (misión corta, media o larga, con una o varias entregas).

Antes de iniciar el vuelo, se realizaba una comprobación visual de la ruta en el mapa para validar su trazado, distribución de puntos de entrega y posibles errores de configuración. A continuación, se armaban los motores desde QGroundControl y se daba comienzo a la misión automática.

Durante el vuelo, el dron recorría los distintos *waypoints* siguiendo la secuencia establecida. En cada punto de entrega simulado, el comportamiento programado consistía en lo ya mencionado:

- Bajar a 0.5 metros.
- Permanecer 30 segundos.
- Retomar la altitud establecida de la misión.

Una vez alcanzado el último *waypoint*, el comportamiento del dron dependía de la configuración de la misión: en algunos casos, retornaba automáticamente al punto de origen, y en otros aterrizaba en una localización final distinta, simulando escenarios de finalización por zona de cobertura.

4.1.2. Variables monitorizadas

Durante la ejecución de cada misión, se realizó un seguimiento detallado de diversas variables clave que permiten evaluar el rendimiento del dron simulado en distintos escenarios. Estas variables fueron seleccionadas en función de su relevancia y de su disponibilidad a través de la interfaz de QGroundControl y los registros generados por el simulador.

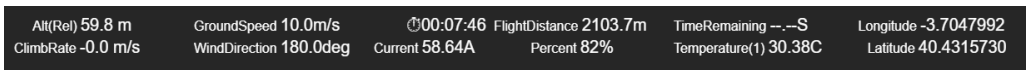

Figura 4.1: Parámetros seleccionados en la interfaz de QGroundControl.

A continuación, se describen las principales variables monitorizadas:

- Tiempo total de vuelo: medida desde el despegue hasta el aterrizaje final o retorno al origen.
- Distancia recorrida: incluye los desplazamientos horizontales entre *waypoints*.
- Altitud en cada fase del vuelo: usada para verificar que los descensos a 0.5 metros se ejecutaban correctamente en los puntos de entrega, y que el dron mantenía una altitud segura en los tramos de desplazamiento.

- Porcentaje de batería restante: supervisado durante todo el vuelo para detectar tendencias de descarga y estimar la autonomía restante tras completar una misión.

- Velocidad de avance: velocidad a la que el dron se desplaza horizontalmente entre *waypoints*.

- Velocidad de ascenso: determina cómo de rápido sube o baja el dron en el eje vertical cuando cambia de altitud, ya sea al despegar, al acercarse a un punto de entrega o al aterrizar.

- Corriente eléctrica: cantidad de carga eléctrica que fluye por segundo a través del sistema. Representa el esfuerzo eléctrico que requiere el dron para mantenerse en vuelo.

Estas variables fueron observadas de forma visual en tiempo real mediante los paneles de QGroundControl durante la ejecución de cada misión. Posteriormente, los valores relevantes fueron anotados manualmente en una hoja de cálculo para su posterior análisis.

4.1.3. Recogida y organización de los datos

La recogida de datos tras la ejecución de cada misión se realizó de forma manual, organizando la información en hojas de cálculo estructuradas. Para cada simulación, se creó una fila donde se registraron las variables más relevantes, así como observaciones específicas sobre el comportamiento del dron durante la misión.

Cada entrada incluye una breve descripción de la ruta y un campo de observaciones técnicas, donde se anotaron aspectos como variaciones de altitud inesperadas, comportamiento del dron frente a obstáculos virtuales, etc.

% BATERIA	DURACION	ALTURA	PUNTOS DE RUTA	DISTANCIA
100%	0:16	30m	0 (ascenso)	0m
97%	1:38	AAT	1 (desplazamiento)	766,5m
97%	1:59	12,2m	1 (descenso)	766,5m
96%	2:31	12,2m	1 (espera)	766,5m
96%	2:45	38,1m	1 (ascenso)	766,5m
94%	4:05	AAT	2 (desplazamiento)	1534,9m
93%	4:26	0,7m	2 (descenso)	1534,9m
93%	4:56	0,7m	2 (espera)	1534,9m
93%	5:11	29,5m	2 (ascenso)	1534,9m
89%	7:44	AAT	3 (desplazamiento)	3035,2m
89%	8:05	19,4m	3 (descenso)	3035,2m
89%	8:35	19,4m	3 (espera)	3035,2m
88%	8:51	48,6m	3 (ascenso)	3035,2m

Figura 4.2: Ejemplo de recogida de datos.

DESCRIPCION	OBSERVACIONES
Ruta probada en el centro de Madrid. Ruta con 3 paradas y regresos a la base. Se ha realizado la prueba por el centro de Madrid para comprabar que puede sortear edificios y variar su altura respecto al terreno	Durante el vuelo no mantiene de manera del todo consistente las alturas. Pues se presupone que la altura de vuelo sobre el terreno es de 30m. No obstante si que varía la altura en función del terreno para mantenerse sobre este pero no los 30m indicados como uno esperaría. Esto se nota sobre todo en los descensos que desciende 30m respecto a la altura absoluta que maneja el dron pero no parece comportarse respecto al terreno en algunos casos alcanzando alturas negativas lo cual es extraño porque no se choca como cabría esperar.

Figura 4.3: Ejemplo de descripción de una misión.

Además, se añadió una columna de observaciones generales que resume el comportamiento global de las misiones, los posibles problemas detectados y recomendaciones para futuras mejoras o pruebas adicionales.

Esta hoja de cálculo permitió centralizar todos los datos obtenidos en las simulaciones de forma clara y ordenada, facilitando el análisis posterior.

4.2. Análisis de resultados

Una vez ejecutadas todas las misiones y recopilados los datos correspondientes, se procedió al análisis de los resultados con el objetivo de evaluar el rendimiento del dron simulado en diferentes escenarios. Este análisis permite comprobar la validez del modelo configurado, identificar patrones de comportamiento y detectar posibles mejoras en la planificación y ejecución de las misiones.

El enfoque se centró en comparar las variables recogidas entre rutas de distinta distancia y carga, observando cómo estas afectan a parámetros clave como la duración del vuelo y el consumo de batería.

En los siguientes apartados se presentan los resultados más representativos obtenidos a partir de las simulaciones.

4.2.1. Análisis del consumo energético

Uno de los objetivos principales de este trabajo fue evaluar el comportamiento energético del dron simulado bajo diferentes condiciones operativas. Para ello, se analizaron las misiones ejecutadas teniendo en cuenta variables como el porcentaje de batería restante al finalizar cada trayecto, la corriente eléctrica consumida, la duración del vuelo y las velocidades asociadas a cada fase del desplazamiento.

A continuación, se presentan los datos de tres misiones representativas: una corta, una media y una larga, que permiten comparar el impacto de la distancia y la operación sobre el consumo energético.

Antes de entrar en los números, conviene detallar la topología de cada recorrido:

la ruta corta enlaza cinco puntos de entrega consecutivos y regresa al origen; la ruta media realiza tres entregas antes de volver; y la ruta larga cubre un único punto de entrega seguido del trayecto de retorno.

Parámetro	**Ruta corta**	**Ruta media**	**Ruta larga**
Distancia total (m)	998	4803	8817
Duración	7 min 53 s	14 min 30 s	18 min 11 s
Batería restante	82 %	65 %	56 %
Corriente eléctrica (A)	54–55	57–60	58–60
Velocidad de avance (m/s)	10	10	10
Velocidad de ascenso (m/s)	2.5	2.5	2.5
Velocidad de descenso (m/s)	–1.5 / –0.5 (último)	–1.5	–0.7

Tabla 4.1: Comparativa de parámetros operativos y energéticos en las tres rutas simuladas.

Los resultados muestran una relación directa y coherente entre la complejidad de la misión y el consumo energético. A medida que aumentan la distancia y el tiempo de vuelo, se observa una disminución progresiva del porcentaje de batería restante. La ruta corta finaliza con un 82 % de batería, mientras que la ruta media baja hasta el 65 %, y la ruta larga, la más exigente en distancia, termina con un 56 %.

También se aprecia un ligero aumento en la corriente eléctrica en las misiones más exigentes, lo que refuerza la idea de que el esfuerzo requerido por el dron para mantener el vuelo se incrementa con la distancia y la duración.

Las velocidades de avance se mantuvieron constantes en las tres misiones (10 m/s), lo que permite aislar el impacto de la distancia y el número de maniobras verticales en el rendimiento energético. Las velocidades de ascenso fueron idénticas en todas las pruebas (2.5 m/s), mientras que las de descenso variaron ligeramente, siendo más lentas en los tramos finales o en misiones de mayor carga.

En conjunto, los resultados obtenidos reflejan un comportamiento coherente del modelo simulado y proporcionan una base para evaluar la autonomía y el rendimiento del dron en distintos escenarios operativos.

4.2.2. Consumo energético en función de la distancia

Con el objetivo de analizar de forma precisa el comportamiento energético del dron simulado en función de la distancia recorrida, se diseñó una misión específica basada en un trayecto rectilíneo sin paradas ni maniobras adicionales. En esta prueba se mantuvo una carga constante de 6 kg y se evitó cualquier tipo de descenso o acción intermedia, solo la primera subida para poder empezar la misión, con el fin de aislar el consumo derivado exclusivamente del desplazamiento horizontal.

Durante la ejecución de esta misión, el dron mantuvo una altitud aproximada de 30 metros, realizando pequeñas correcciones de altura para evitar obstáculos en el entorno . Durante la misión, se registró de forma periódica el porcentaje de batería restante, el tiempo transcurrido y la velocidad tras cada kilómetro recorrido. Esto permitió obtener una curva de consumo energético progresivo, útil para estimar

la autonomía máxima del dron en condiciones ideales, así como para evaluar su eficiencia por kilómetro.

A continuación, se resumen los datos más relevantes obtenidos durante la misión:

- **1000 m**: 93 % batería – 3:00 min – velocidad media 6.7 m/s
- **2000 m**: 86 % batería – 2:45 min – 6.0 m/s
- **3000 m**: 79 % batería – 2:45 min – 6.0 m/s
- **4000 m**: 72 % batería – 2:53 min – 5.8 m/s
- **5000 m**: 64 % batería – 2:53 min – 5.8 m/s
- **6000 m**: 56 % batería – 3:25 min – 4.5 m/s
- **7000 m**: 39 % batería – 6:20 min – 4.0 m/s (con tramos irregulares)
- **8000 m**: 30 % batería – 3:10 min – 5.2 m/s
- **8932 m**: 21 % batería – 3:07 min – 5.2 m/s

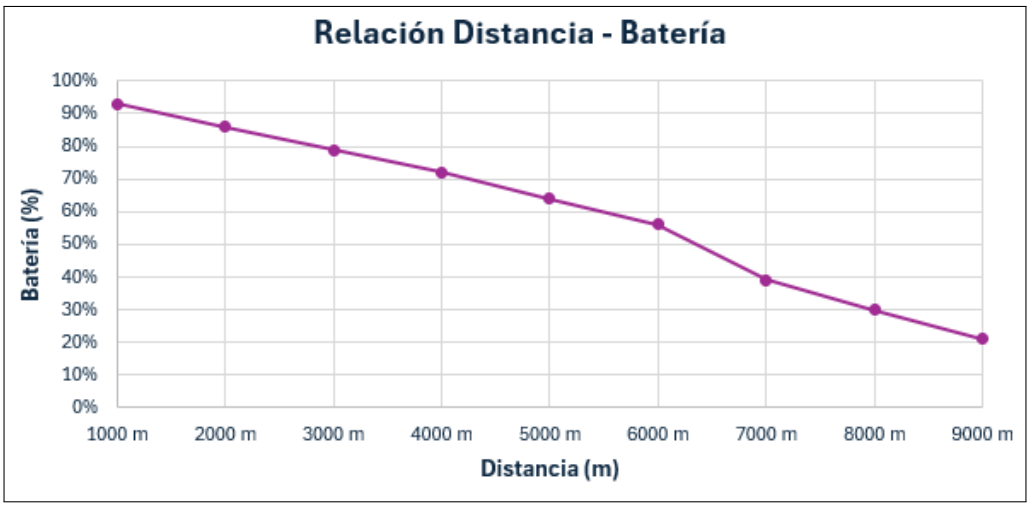

Figura 4.4: Relación entre la distancia recorrida y la batería consumida.

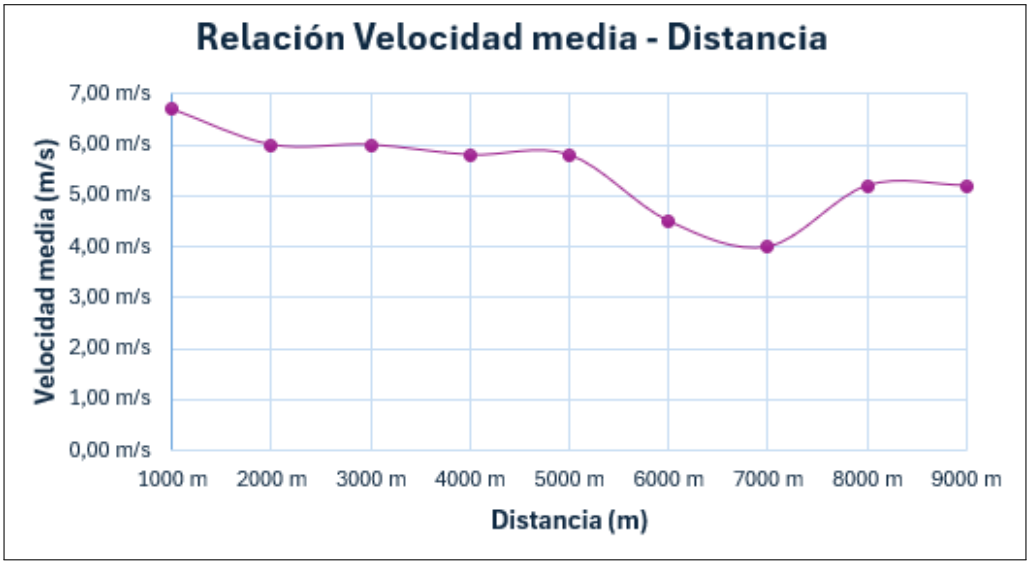

Figura 4.5: Relación entre la distancia recorrida y la velocidad media.

Las gráficas generadas a partir de los datos recopilados refuerzan las conclusiones extraídas del análisis numérico. En la Figura 4.4, se observa una relación prácticamente lineal decreciente entre la distancia recorrida y el porcentaje de batería restante, lo que sugiere un consumo energético progresivo y constante durante la mayor parte del trayecto. Esta tendencia refuerza la idea de que, en condiciones de vuelo estables, el dron mantiene una eficiencia energética predecible, útil para planificaciones logísticas.

Por otro lado, la Figura 4.5 muestra la evolución de la velocidad media en función de la distancia recorrida. Hasta el kilómetro 5000, el dron mantiene una velocidad bastante estable, con pequeñas oscilaciones. Sin embargo, a partir del sexto kilómetro se aprecia una disminución significativa de la velocidad, alcanzando su punto más bajo en torno a los 7000 metros. Esta caída coincide con un incremento en la demanda de corriente como también hemos detectado durante la simulación. Esto sugiere que, con niveles bajos de batería, el rendimiento del dron comienza a degradarse visiblemente. Posteriormente, el sistema recupera parte de la velocidad, aunque sin volver a los niveles iniciales.

Ambas representaciones visuales permiten consolidar las conclusiones del análisis: el modelo simulado presenta un comportamiento energético predecible hasta cierto punto de descarga, tras el cual se evidencia una pérdida de eficiencia que afecta directamente a la velocidad y estabilidad del vuelo.

4.2.3. Influencia de la carga transportada sobre el rendimiento

Con el objetivo de analizar el impacto de la carga útil sobre el comportamiento del dron simulado, se diseñó una serie de misiones idénticas en cuanto a trayecto, estructura y distancia (2 km en línea recta), variando únicamente el peso total del dron entre 6 kg, 4 kg y 2 kg. De este modo, fue posible aislar el efecto del peso sobre parámetros clave como el consumo energético, la velocidad de avance y la duración

del vuelo.

Durante cada misión se registraron los tiempos parciales por kilómetro, el porcentaje de batería restante, la corriente eléctrica consumida y la velocidad media estimada. Los resultados se recogen en la siguiente tabla:

Carga (kg)	**Distancia (m)**	**Tiempo**	**Batería restante(%)**	**Corriente (A)**
6	1000	2:57	93	50
	2000	2:38	87	
4	1000	2:54	95	33
	2000	2:32	91	
2	1000	3:14	97	18
	2000	2:45	95	

Tabla 4.2: Comparativa de consumo y rendimiento según la carga transportada

A partir de los resultados, se confirma que una mayor carga implica un consumo energético significativamente superior. La misión con 6 kg de peso finalizó con un 87 % de batería restante, mientras que la misma ruta con 2 kg de carga finalizó con un 95 %. La diferencia se acentúa al observar la corriente consumida: 50 A con 6 kg frente a 18 A con 2 kg, lo que refleja un esfuerzo mucho mayor por parte del sistema para mantener el vuelo con más peso.

Por otro lado, la velocidad media fue ligeramente mayor con las cargas más altas, lo cual puede explicarse por la mayor inercia del sistema y la calibración de los parámetros físicos del modelo. Aun así, las diferencias en el tiempo por kilómetro fueron pequeñas, y todas las misiones se completaron en tiempos similares. Esto indica que el sistema mantiene un buen control de vuelo independientemente del peso, aunque el consumo de energía aumenta cuanto mayor es la carga.

Esta prueba deja claro cómo influye el peso transportado en el rendimiento del dron, y demuestra lo importante que es planificar bien la carga para aprovechar al máximo la autonomía.

4.2.4. Observaciones generales y problemas

Durante las simulaciones se detectaron ciertos comportamientos inesperados, así como limitaciones propias del entorno, que conviene destacar. Estos aspectos, aunque no impidieron completar los experimentos, sí condicionan la interpretación de los resultados y marcan el alcance real del modelo simulado.

4.2.4.1. Comportamiento con batería baja

En vuelos prolongados, especialmente en la misión de 9 km con 6 kg de carga, se observó una disminución progresiva de la velocidad media a medida que el nivel de batería descendía por debajo del 40 %. En ciertos tramos, el dron llegó a detenerse brevemente antes de continuar el trayecto, aunque a menor velocidad.

Además, en otras pruebas el dron se detuvo completamente antes de alcanzar el destino, a pesar de que aún quedaba un porcentaje de batería disponible (20 %-30 %).

No se identificó ningún fallo como tal, lo que sugiere que esta parada podría deberse a una medida de protección del sistema frente a descargas completas de la batería, simulando un comportamiento realista en el que el dron prioriza la conservación del sistema energético para evitar daños graves.

Aunque el entorno de simulación no proporciona información explícita sobre estos mecanismos de protección, estos comportamientos destacan la importancia de mantener una batería que no llegue a descargarse demasiado, ya sea porque baja sus prestaciones o porque se detiene por completo incapacitando la misión en la que se encuentre.

4.2.4.2. Finalización incompleta de la misión

En una ocasión, el dron no alcanzó exactamente el punto final previsto en el plan de vuelo, deteniéndose aproximadamente 70 metros antes. Aunque el desvío fue pequeño, este comportamiento podría tener consecuencias relevantes en contextos reales donde la precisión es clave. Esta desviación probablemente se debió a la combinación de batería baja y posibles limitaciones internas del modelo de navegación.

Además, se detectaron errores relacionados con la gestión de la altitud en los puntos de entrega. En rutas donde el dron debía descender hasta 0.5 metros sobre el suelo, el comportamiento no fue siempre el esperado. En ciertos casos, el dron ascendía previamente a alturas superiores (por ejemplo, hasta 70 metros) para evitar obstáculos, y posteriormente descendía exactamente esos mismos metros sin tener en cuenta la altura absoluta del terreno.

Es decir, si la altitud objetivo durante la ruta era 30 metros y el dron subía a 70 metros, al ejecutar el descenso a "0.5 m", lo interpretaba como "70 m - 29.5 m = 0.5 m de descenso", alcanzando realmente los 40.5 m de altitud, en lugar de acercarse al suelo. Esta incoherencia entre la altitud relativa y la real provocaba que en algunos puntos el dron creyera haber descendido completamente, sin haberlo hecho en términos operativos.

Este problema refleja una limitación del sistema para gestionar correctamente los descensos cuando hay variaciones de altitud no previstas en la planificación.

4.2.4.3. Ajuste de parámetros y coherencia del modelo

Durante la realización de las pruebas fue necesario realizar diversos ajustes en los parámetros físicos y energéticos del dron simulado con el fin de representar correctamente distintas configuraciones de carga (2 kg, 4 kg y 6 kg). Aunque a priori podría parecer suficiente con modificar únicamente el valor de la masa (`model.mass`), en la práctica esto generaba comportamientos incoherentes.

Por ejemplo, al reducir el peso del dron sin modificar otros valores como la corriente de referencia (`refCurrent`) o el coeficiente de potencia (`power_factor`), el sistema mantenía un consumo y una respuesta propios de una configuración más pesada, lo que se traducía en velocidades más bajas o maniobras inestables, especialmente en vuelos con 2 kg de carga.

Para corregir estas inconsistencias, fue necesario recalibrar manualmente varios

parámetros de los archivos SIM_Multicopter.cpp y SIM_Frame.cpp para cada nivel de carga, ajustando la corriente, el empuje y el factor de eficiencia energética en proporción al nuevo peso. Estos cambios requerían recompilar el entorno de simulación (`./waf copter`) tras cada modificación y luego comprobar sobre el programa si el cambio se reproducía en la simulación.

A pesar de ello, una vez corregidos, los modelos se comportaron de forma estable y permitieron realizar un análisis comparativo realista entre diferentes configuraciones de carga.

4.2.4.4. Limitaciones del entorno de simulación

Aunque QGroundControl permite una simulación bastante detallada del comportamiento de un dron en vuelo, existen ciertas limitaciones que deben tenerse en cuenta al interpretar los resultados.

Por un lado, el sistema no ofrece un control sobre variables ambientales como el viento, la temperatura o las condiciones climáticas adversas desde la propia interfaz de QGroundControl. Cualquier intento de introducir condiciones meteorológicas realistas requiere intervención externa a través de consola o *scripts*, lo que limita al usuario a la hora de realizar simulaciones más complejas con distintos escenarios operativos.

Por otro lado, la interfaz de planificación de misiones ofrece varias funcionalidades, pero la edición manual de rutas o ajustes finos como el tiempo de espera en los puntos de entrega o los perfiles de velocidad por tramo deben configurarse individualmente para cada *waypoint*. Esto ralentiza el proceso de diseño y puede derivar en errores (sobre todo en los tiempos de espera y altura de cada *waypoint*).

Además, la exportación de datos para análisis posteriores (batería, tiempo, distancia, velocidad, etc.) no se realiza de forma automatizada. La recogida de resultados se hizo de forma manual, anotando los valores observados y escribiéndolos en hojas de cálculo para su posterior análisis. Esto limita la posibilidad de generar grandes volúmenes de datos de forma rápida y sistemática.

En conjunto, el análisis realizado ha permitido evaluar de forma detallada el comportamiento del dron simulado en diferentes escenarios operativos. Se han comparado trayectos de distinta longitud, cargas y patrones de vuelo tanto continuos como con múltiples entregas, analizando su impacto sobre el consumo energético, la duración del vuelo y la eficiencia general del sistema.

A pesar de las limitaciones propias del entorno de simulación, los resultados obtenidos han sido consistentes y útiles para entender cómo distintos factores afectan al rendimiento global de una misión de reparto con drones. Las pruebas realizadas y los ajustes hechos al modelo muestran el enfoque utilizado y confirman lo importante que es planificar bien las misiones para sacar el máximo rendimiento y autonomía del dron y evitar problemas en las rutas.

Este análisis constituye la base sobre la que se apoyan las conclusiones generales y los posibles trabajos futuros, los cuales pueden coger este proyecto como guía para desarrollar sus propias necesidades.

Capítulo 5

Interfaz y automatización

Con la intención de facilitar la tarea de generar las rutas (misiones) y hacer más amigable y accesible el trabajo a usuarios no expertos, hemos desarrollado una interfaz. Las características principales que pretendemos ofrecer con ella es la capacidad de buscar ubicaciones para facilitar la tarea de crear puntos de ruta, crear y destruir los mismos configurando sus parámetros principales y que al cerrar el programa, se genere un archivo con extensión .plan con la misión lista para introducir en el simulador.

También hemos creado un pequeño .bat (*script* bat) que, con una cierta configuración del entorno, permite lanzar la simulación del dron automáticamente, por vía de otro *script* .ps1 (*script* PowerShell). Facilitando así la simulación al permitir omitir el contacto de usuarios inexpertos con la consola, pues tendrían que realizar una serie de pasos utilizando la línea de comandos.

Combinando ambos elementos (la interfaz y los *scripts*) se genera un entorno muy amigable para todo tipo de usuarios. Solo hace falta ejecutar la interfaz y, al cerrarla, automáticamente se crearán todos los archivos necesarios para la simulación, se configurarán elementos de la misma y se lanzará.

5.1. Interfaz

Para el desarrollo de la interfaz hemos utilizado Unity, un motor 3D. Hemos tomado esta decisión por la facilidad que da para exportar el proyecto a otras plataformas, su amplia comunidad de usuarios y nuestra familiaridad con el mismo. Finalmente, nos hemos decidido por desarrollar un programa pensado para ejecutarse exclusivamente en PC, aunque podría exportarse a otras plataformas sin demasiadas dificultades.

Para todo lo relacionado con el mapa, hemos utilizado Mapbox. Funciona utilizando OpenStreetMap (OSM), un proyecto colaborativo que permite crear mapas libres. Además, es simple de usar y fácilmente integrable en Unity por medio de un paquete.

La interfaz resultante permite conocer la latitud y longitud de un determinado punto (es vital porque al final los puntos de ruta lo que utilizan son dichas coorde-

nadas), tiene otros dos campos para modificar la altura y el tiempo de espera que realizará el dron en las paradas, otro campo para indicar el peso inicial, una barra de búsqueda para localizar fácilmente localizaciones y otros dos botones que permiten añadir y eliminar puntos de ruta con facilidad. Todo esto se puede apreciar con facilidad en la Figura 5.1.

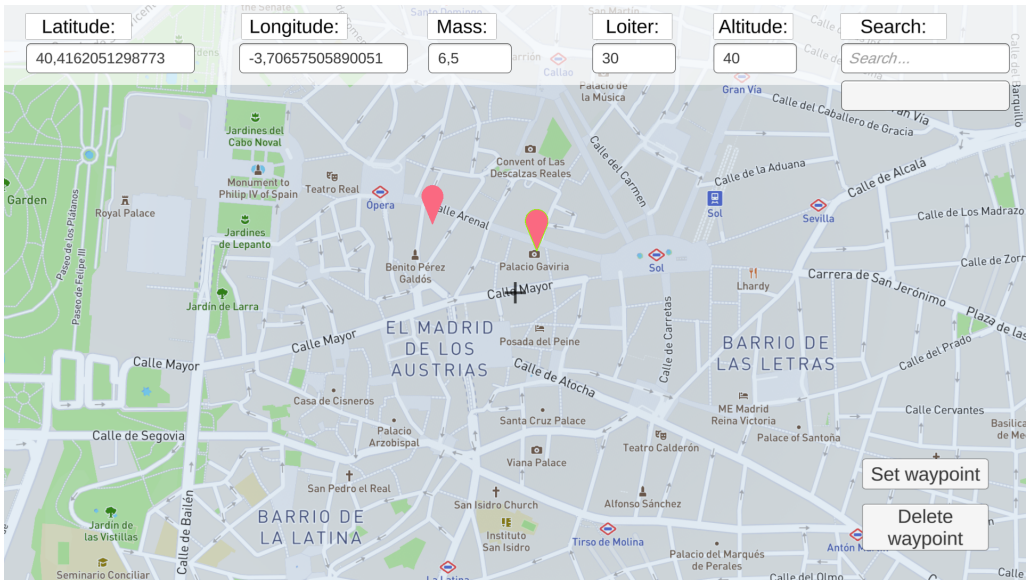

Figura 5.1: Interfaz desarrollada para generar misiones.

Para poder desplazarse por el mapa basta con arrastrar el ratón por la pantalla con el clic izquierdo pulsado, se puede realizar zoom in y zoom out con la rueda del ratón. Para colocar un waypoint en una posición concreta basta con realizar clic en dicho punto. Para poder fijar el punto de ruta se debe presionar el botón "Set waypoint", del mismo modo para eliminar un punto de ruta se debe hacer clic sobre él para seleccionarlo, se indicará que está seleccionado con un ligero reborde verde (ver Figura 5.1) y una vez esté seleccionado pulsando el botón "Delete waypoint"se eliminará.

Cuando se modifican los parámetros de la parte superior (tiempo de espera y altura) estos se aplican al próximo punto de ruta que se vaya a introducir. Si quisiésemos conocer dichos valores de algún punto de ruta ya definido, bastaría con seleccionarlo y ambos campos tomarían los valores concretos de dicho punto de ruta. Mientras el punto de ruta esté seleccionado, si se modifican dichos valores estos se modifican. De esta manera, se pueden editar los puntos ya añadidos.

Otro parámetro modificable es "Mass", que permite modificar el peso inicial del dron. Lo cual es un detalle importante porque permite medir la condición límite de dicha simulación. Al permitir comprobar la completitud de la misión en el caso peor, que es mantener el peso máximo durante la totalidad de la misma.

Además, hemos asociado un *script* de python que se ejecuta al cerrar el programa. Este utiliza el archivo waypoints.txt que se genera a partir de los puntos de ruta introducidos desde la interfaz con la información correspondiente (altitud, longitud, tiempo de espera y altura), un ejemplo del formato adoptado por el mismo se puede

observar en la Figura 5.2. No es un formato con una gran escalabilidad, pero al estar realizando constantes pruebas lo hemos elegido por su simplicidad y conveniencia.

El *script* no se limita a generar una misión colocando los puntos por el orden en el que se introducen en la interfaz. Para ordenar los puntos de modo que se minimice la distancia recorrida por el dron y, en consecuencia, su consumo de batería, utilizamos la fórmula de Haversine. Es una fórmula que se utiliza precisamente para calcular rutas de vuelo o distancias en mapas por ejemplo. Permitiendo conocer la distancia entre dos puntos a partir de su latitud y su longitud. De este modo podemos conocer las distancias entre los puntos y ubicarlos minimizando la distancia total.

```
40,4495019156744
-3,730095869873
30
30
40,4492739658509
-3,72679330562402
50
50
40,4498439748434
-3,72492107790985
70
70
```

Figura 5.2: Ejemplo de formato de waypoints.txt.

Una vez seguido este proceso, un usuario sin conocimientos especializados en el tema puede generar un archivo listo para usarse y llevar a cabo una misión o, en este caso, para simular una ruta y comprobar la viabilidad de la misma. El punto fuerte de nuestra interfaz es que no hay una gran variedad de opciones a configurar ni elementos que puedan abrumar a un usuario con pocas nociones, permitiendo solo configurar los elementos principales de nuestro modelo. De este modo, sin tener conocimientos respecto al formato del archivo o las distintas fases del vuelo se pueden crear fácilmente misiones. Siendo estas completamente funcionales y, además, incluyendo todas las fases del vuelo.

5.2. Automatización

Al realizar todos los pasos previos, solo quedaría lanzar la simulación, de modo que, tras facilitar el .plan (misión) antes generado se lleve a cabo la misma. Sin embargo, este es otro paso para nada accesible para usuarios que no tienen por qué conocer cómo se lleva a cabo la simulación.

Sin el trabajo de automatización realizado, para lanzar la simulación se debería acceder al entorno de WSL. Luego, por línea de comandos configurar el dron e iniciar la simulación. Una vez realizado esto, ejecutar QGroundControl y configurar la conexión de ambos para que se conecten y, finalmente, poder realizar la simulación con todas las garantías.

Para ello hemos generado un pequeño .bat (*script* bat) que realiza principalmente dos acciones. Por un lado, desde el WSL que se espera que tenga instalado y configurado en una ruta en concreto todo el código fuente de ArduPilot, lanza la simulación y, posteriormente, ejecuta el programa QGroundControl, este *script* se lanza automáticamente al cerrar la interfaz con la tecla .Escz con ello ya se tendría la simulación lista para solo tener que aportar el .plan y así poder visualizar fácilmente la simulación.

Como se puede apreciar en la Figura 5.3 se espera que QGroundControl esté instalado en la ruta por defecto que ofrece para su instalación. Luego se comprueba el usuario local del equipo y la ruta del entorno virtual esperado con el código fuente de ArduPilot y todas las dependencias instaladas. Posteriormente, se realizan pasos intermedios para configurar la masa del dron en el modelo. Si todo esto se cumple, se lanzará con normalidad la simulación y QGroundControl para gestionar la misma. Permitiendo, al finalizar la ejecución del *script* tener el entorno completamente listo para funcionar.

```
# Ejecutar QGroundControl
Start-Process -FilePath "C:\Program Files\QGroundControl\QGroundControl.exe"

# Obtener el nombre de usuario de WSL
$user = wsl whoami

# Ruta al entorno virtual en WSL
$venvPath = "/home/$user/venv-ardupilot/bin/activate"

# Archivo generado con la masa
$actDir = Split-Path -Parent $MyInvocation.MyCommand.Path
$massF = Join-Path $actDir "mass.txt"
$massV = Get-Content $massF

# Crear comando para asignar masa inicial
$MCcpp = "/home/$user/ardupilot/libraries/SITL/SIM_Multicopter.cpp"

# Reemplaza el valor de masa
$massCommand = "sed -i 's/mass = [0-9.]*f;/mass = ${massV}f;/g' $MCcpp"

wsl bash -c "$massCommand"

# Ruta proyecto ArduPilot
$ardupilot = "/home/$user/ardupilot"

# Comando para compilar con waf
$wafCommand = "source $venvPath && cd $ardupilot && ./waf copter"

wsl bash -c "$wafCommand"

# Crear el comando para ejecutar el simulador
$command = "source $venvPath && cd /home/$user/ardupilot/Tools/autotest && \
./sim_vehicle.py -v ArduCopter \
--custom-location=40.415363,-3.707398,650,30 --map --console"

# Ejecutar el comando dentro de WSL
wsl bash -c "$command"
```

Figura 5.3: *Script* auto.ps1 ejecutado por auto.bat.

Capítulo 6

Conclusiones y Trabajo Futuro

6.1. Conclusiones

Este proyecto ha confirmado, mediante simulación SITL con ArduPilot y QGroundControl, la viabilidad de emplear un dron de la clase DJI Matrice 600 PRO para misiones de reparto de última milla en entornos urbanos, siempre que las distancias y las cargas transportadas se mantengan dentro de márgenes específicos descritos en esta memoria. Para alcanzar esta conclusión se recrearon diferentes escenarios que combinan:

- Tres perfiles de misión: corta, media y larga; de 1 km, 5 km y 9km, respectivamente.
- Tres niveles de carga útil (2 kg, 4 kg y 6 kg).
- Rutas multiparada con despegue, parada en cada punto de entrega y aterrizaje.

6.1.1. Principales hallazgos técnicos

- Comportamiento energético: en los tres perfiles de distancia la batería retuvo 80 %, 65 % y 55 % de capacidad restante, respectivamente. La descarga se mantiene casi lineal hasta el 40 % y a partir de ahí se observa una caída nítida del rendimiento. Con menos del 30 % el firmware activa rutinas de autoprotección que reducen la velocidad o pausan el avance.
- Impacto de la carga: en igualdad de rutas, pasar de 2 kg a 6 kg multiplicó la corriente media de unos 18 A a unos 50 A. Además, se recortó la reserva de batería en 8 puntos, confirmando que el peso transportado es una de las principales restricciones.
- Impacto de la distancia. En un vuelo rectilíneo de 9 km la batería descendió al 21 % y la velocidad empezó a caer pasado el quinto kilómetro; aun así, el dron conservó margen para un retorno seguro al punto de origen.

6.1.2. Contribuciones prácticas

- Generador de misiones (.plan): *script* Python que define automáticamente longitudes, waypoints y perfiles verticales para experimentos.
- Lanzadores auto.bat / auto.ps1: desde WSL, compilan sim_vehicle.py y abren QGroundControl con un clic, agilizando la configuración inicial para usuarios sin experiencia.
- Guía de ajuste de parámetros físicos: procedimiento para sincronizar masa, batería y potencias en SIM_Multicopter.cpp, aplicable a cualquier dron hexacóptero.

Los resultados evidencian que es factible operar misiones de entrega en un radio de hasta 5 km con cargas en torno a los 4 kg, manteniendo márgenes de seguridad tanto energéticos como de rendimiento. Más allá de esos valores, la autonomía disminuye rápidamente y conviene recurrir a estrategias de relevo o recarga intermedia durante las entregas. El trabajo proporciona así una base para planificar rutas, dimensionar infraestructuras de recarga y, en última instancia, tomar decisiones fundamentadas sobre el despliegue de drones en la ciudad.

6.2. Trabajo futuro

Las simulaciones han demostrado la viabilidad técnica del sistema en un escenario controlado, el siguiente paso es acercarlo a la operación real. A continuación se describen, los frentes de investigación y desarrollo que permitirían evolucionar el prototipo hacia una solución plenamente comercial.

6.2.1. Planificación colaborativa de flotas

Hasta ahora la optimización se ha centrado en un único dron, pero las operaciones reales implicarán decenas de UAV que comparten tanto el espacio aéreo como los puntos de recarga. Será necesario desarrollar algoritmos de asignación de pedidos capaces de adjudicar cada paquete al dron más adecuado en función de la franja horaria de entrega, la carga disponible, el nivel de batería y la posible congestión aérea.

6.2.2. Planificación dinámica y factores meteorológicos

Debido a las condiciones del simulador, el peso no se puede modificar dinámicamente y siempre hemos trabajado con el mismo peso, 6 kg, ya que, si la misión es satisfactoria con el mayor peso soportado, con menos peso también tendrá éxito. En la práctica, el dron no mantiene una carga constante a lo largo de la ruta: cada entrega disminuye progresivamente la masa útil y, con ella, la demanda energética. Para reflejar este comportamiento realista, el planificador debe poder ajustar dinámicamente el peso del vehículo y recalcular en tiempo real el consumo previsto, los márgenes de batería y la velocidad óptima de crucero. Esa actualización continua del

modelo de masa permitirá aprovechar mejor la autonomía disponible y, si es preciso, decidir si se requiere una parada intermedia de recarga o un cambio de trayectoria.

A esto, también se suma la influencia del clima urbano, donde rachas de viento y turbulencias entre edificios alteran el consumo y la estabilidad del vuelo. Integrar datos meteorológicos hará posible generar rutas alternativas cuando las condiciones pongan en riesgo la seguridad de la misión. De igual forma, la lógica del sistema debe combinar la información de peso actualizado, meteorología y restricciones del espacio aéreo para seleccionar siempre la trayectoria más eficiente y segura, garantizando que el dron complete su misión dentro de los límites energéticos y regulatorios establecidos.

6.2.3. Entrega de precisión

Para que la logística de última milla resulte verdaderamente cómoda para el cliente y eficiente para el operador, el dron debe ser capaz de depositar la carga con una precisión de unos pocos centímetros, ya sea en un balcón de edificio alto o en un casillero inteligente situado a ras de calle. Lograrlo exige dotar a la aeronave de un sistema de percepción que consiga dejar el paquete en el sitio indicado, esto requeriría de algoritmos de localización basados en vista como pueden ser OF (optical flow) o SVO (stereo visual odometry).

Primero, se debería conseguir la detección robusta del punto de entrega. Se propone colocar balizas visuales o infrarrojas en la plataforma de aterrizaje. A bordo, un lector debe ser capaz de validar la baliza y así depositar el paquete.

Segundo, la estimación de la altura durante los últimos metros debe ser muy precisa. Se deberían conseguir medidas que permitiran suavizar el perfil de descenso, prevenir impactos y aterrizar con la delicadeza necesaria para no dañar la carga ni la superficie de entrega.

De forma complementaria, la percepción ambiental puede enriquecerse con una pequeña red neuronal entrenada para reconocer obstáculos inesperados (cables, ramas, aves) y activar maniobras evasivas.

Por último, el sistema debe ofrecer retroalimentación al usuario: una notificación o mensaje en la aplicación móvil confirmará la entrega mediante una imagen capturada en el momento de dejar el paquete. De cara al operador, los logs de cámara y telemetría servirán para evaluar la precisión real lograda y mejorar los modelos de control en futuras misiones.

6.2.4. Modelado avanzado de batería y logística de recarga

Las simulaciones han empleado un modelo de descarga lineal, suficiente para un primer análisis, pero lejos de la realidad electroquímica de las baterías. Incorporar un modelo de batería más preciso que sea capaz de tener en cuenta otros factores (temperatura, ciclos de degradación, etc.) permitiría predecir la tensión y la capacidad residual de la batería.

Sobre esa base podrían evaluarse dos estrategias operativas: estaciones de recarga rápida (15-20 min) instaladas en cubiertas estratégicas y sistemas de battery-swap que sustituyan la batería del dron en poco tiempo. La simulación de ambas alterna-

tivas permitirá medir con detalle el tiempo efectivo de servicio, el coste energético por kilovatio-hora y la degradación de las celdas, indicando con precisión en qué momento resulta más rentable aumentar la flota o expandir la infraestructura de recarga.

6.2.5. Gestión de fallos y ciberseguridad

La aceptación social y la certificación de un sistema de reparto con drones dependen, en buena medida, de su capacidad para sobrevivir a situaciones anómalas sin provocar daños a personas o bienes. Para alcanzar eso es preciso avanzar en dos ejes complementarios: la tolerancia a fallos y la defensa cibernética.

En el plano operativo se propone una batería de ensayos de inyección de fallos que cubra los escenarios más críticos: pérdida de un rotor, degradación severa del enlace de control, apagón total de sensores inerciales o sobrecalentamiento en la batería. Cada ensayo debe verificar que el dron activa la respuesta adecuada (auto-hover, Return-to-Home, etc.) con rutas alternativas o descensos controlados a la zona segura más próxima y que esa maniobra se ejecuta dentro de márgenes de energía y tiempo aceptables.

También, el sistema necesita un endurecimiento de ciberseguridad acorde con las guías y las exigencias consensuadas por empresas y gobiernos, para ello sería bueno esta serie de acciones que deberían tener los drones utilizados:

- Arranque seguro y firma de firmware: el dron debe verificar criptográficamente la imagen de vuelo antes de ejecutarla, evitando cargas maliciosas.
- Cifrado de telemetría y control: todo tráfico MAVLink/UDP se encapsulará en enlaces TLS o Datagram TLS, con gestión de claves y rotación periódica.
- Reducción de la superficie de ataque: deshabilitar puertos MAVLink no utilizados, limitar comandos, etc.

Introduction

Postal and courier services were traditionally used to take care of the delivery of letters and other documents, allowing people to be connected to people who live far away. With the arrival of the email in the 1970s, thanks to Ray Tomlinson, and its popularization in subsequent decades, the workload of these companies decrease in this fields. Due to the immediacy and convenience of the new alternative, it went from being barely present in the academic world to becoming a standard in our society. Furthermore, recently, there are attempts to reduce paper communications with an environmental motivation, which accentuates this trend.

However, in the last decade, courier and postal services have experienced a significant resurgence. This is primarily due to the success and normalization of online shopping and other online services available to the general population. Home deliveries and pickup points have become more popular, raising new questions and issues related to how to deliver these packages and how to address all the resulting logistics challenges.

In this global context, the possibility of alternatives to traditional delivery methods takes on particular interest. Drones stand out, due to their versatility and great potential, as one of the options that could revolutionize courier services as we know them.

The drone industry is still starting, at a very early stage in terms of legislation and research in many aspects. However, thanks to pilot projects and research into applications for different uses in different types of contexts (package deliveries, food delivery, exploration and surveillance of areas, recreational shows, etc.), it is gradually becoming the best option in many cases. Proof of this is how its presence and use are gradually becoming normalized in many aspects of our lives.

Motivation

The motivation behind this work is based on several reasons. On the one hand, package deliveries are a daily occurrence, with approximate delivery times, availability requirements for package pickup, and the deliveries themselves are subject to traffic conditions and other inconveniences that limit the system's efficiency. On the other hand, drones, due to their size, power, and flexibility in design and use, present a highly interesting avenue for research to address and solve all these challenges in

an elegant and efficient manner.

Although we are still some way off from drone package delivery becoming a global and everyday reality, solid models are already beginning to emerge that individually address and propose different solutions to these problems. Pilot projects are also underway to test some of these models, bringing us closer to a future where all this work culminates in a reality that puts them into practice.

For all the reasons mentioned above, we consider it interesting to study the issue and how to approach it, trying to propose a reasonable and interesting solution that addresses its main problems. Furthermore, simulation is a very interesting vehicle for addressing all these issues in a simple, practical way that allows us to abstract the main problem and thus solve a complex problem in a straightforward manner.

Objectives

The main objective of the project is to develop simulations of package delivery using drones that will allow for analyzing the feasibility of using this technology in urban environments.

Specifically, the objectives are:

- Design a simulation model that realistically represents an urban environment, obstacles, etc.
- Implement optimization algorithms that minimize total time, energy consumption, or distance traveled.
- Analyze the impact of different variables such as the number of drones, order weights, etc.
- Evaluate the technical and logistical feasibility of delivering packages with drones in cities

This work seeks to provide a basis for decision-making for future applications based on drone-based package delivery.

Workplan

The development of the work has been carried out through different phases:

- Researching and reading of scientific literature: with the intention of understand the state of the art and set reasonable and interesting objectives. Learn about other applications by reading articles and other sources to extract interesting information that could be applied to our models.
- Search for suitable drone models: In the initial phase, a review of the drone models currently available on the market will be carried out to use as examples for our simulations.

- Selection of the simulation tool: Once the models have been defined, different simulation platforms will be evaluated, and the one that best suits the project requirements will be chosen; in our case, the QGroundControl program.

- Connection to the drone and parameter adjustment: The virtual drone will be configured in the selected software, and its parameters will be adjusted to faithfully reproduce the technical specifications of the real drones analyzed.

- Route design: Once the simulated model has been calibrated, various delivery routes will be designed to serve as case studies for efficiency and feasibility testing.

- Getting and interpreting results: Finally, the data generated by the simulations will be collected, and an analysis will be performed to evaluate the efficiency of the deliveries and the feasibility of the proposed routes.

- Interface development: we will develop an interface to facilitate route creation for non-specialized users. This will simplify the task of generating routes, allowing to configure the waypoints and most important parameters for the simulation to be defined naturally on a map.

All the code, *scripts*, and documentation needed to reproduce the simulations are available on GitHub at `https://github.com/jrubioczo/UltimaMillaTFG.git`

Below, as can be seen in Figure 6.1, we have created a Gantt chart that reflects the progress of the TFG over time and how the different objectives have been achieved.

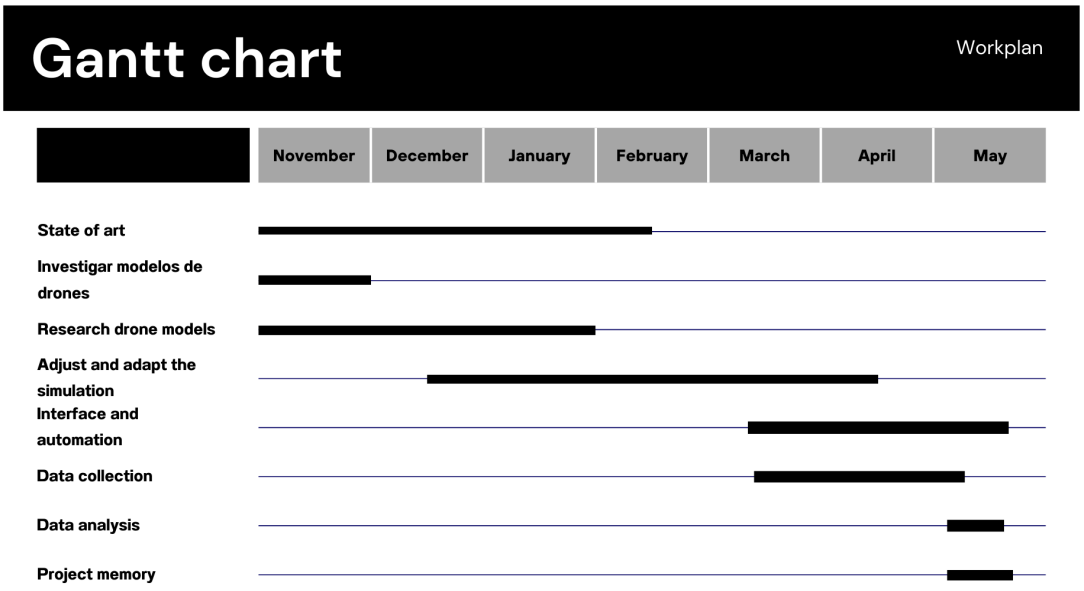

Figure 6.1: Gantt Chart (Workplan).

Memory organization

This report provides a brief summary, six chapters (described below), a bibliography, and the personal contributions of each author. A brief introduction to each chapter will now follow:

- Chapter 1. Introduction: A brief description of the motivation behind the project will be provided. The initial objectives set will also be included. It also includes a work plan and a Gantt chart that reflects how the status of the final project has evolved over time.
- Chapter 2. State of Art: A review of other articles and research that address related topics and have a direct impact on our field of study is provided. The various alternatives considered for simulation and automation are also described, indicating which ones were chosen and why. A brief description of competitive drone models ideal for the intended purposes is provided.
- Chapter 3. Design and configuration of the simulation environment: The steps followed to achieve the final simulation state will be discussed in detail. The different steps involved in carrying out the simulation will be described, indicating the specific decisions that impact the simulation and how they affect it. An example of one will be described to provide a deeper understanding of the simulation.
- Chapter 4. Test execution and results analysis: The tests performed are described, along with their purpose, to validate the simulation. Data is extracted from the various tests to shed light on the model's validity and draw useful conclusions through analysis.
- Chapter 5. Interface and automation: Design of an interface to facilitate the task of creating routes for non-expert users, allowing for the modification of some parameters. Its operation and usefulness are explained. In addition, the automation process is described to facilitate launching the simulation for users without prior knowledge.
- Chapter 6. Conclusions and Future Work: A retrospective review of the project and the results obtained is provided. Future lines of research and their potential are included.

Conclusions and Future Work

This project has confirmed, through SITL simulation using ArduPilot and QGroundControl, the feasibility of using a DJI Matrice 600 PRO drone for last mile delivery missions in urban environments. Always that the distances and loads transported remain within the specific limits described in this report. To reach this conclusion, we have recreated different scenarios combining:

- Three types of missions: short, medium, and long; 1 km, 5 km, and 9 km, respectively.
- Three payload levels (2 kg, 4 kg and 6 kg).
- Routes with multiple stops with takeoff, stops in each delivery point and landing.

Main technical findings

- Energy performance: in the three kinds of distances, the battery end with 80%, 65%, and 55% of remaining capacity, respectively. Discharge remains almost linear up to 40%, after this we will observe a sharp drop in performance. At less than 30%, the firmware activates self-protection routines that reduce speed or pause progress.
- Load impact: in equality of routes, going from 2 kg to 6 kg multiplied the average current from about 18 A to about 50 A. Furthermore, the battery reserve was reduced by 8 points, confirming that the weight carried is one of the main constraints.
- Impact of distance. during a 9 km straight-line flight, the battery dropped to 21% and the speed began to drop past the 5 km. However, the drone still had enough time and battery to return safely to its home point.

Practical contributions

- Mission generator (.plan): python script that automatically defines longitudes, waypoints, and vertical profiles for experiments.

- Auto.bat / auto.ps1 launchers: from WSL, compile sim_vehicle.py and open QGroundControl with one click, speeding up initial setup for inexperienced users.
- Physical parameters configuration guide: guide of steps for synchronizing ground, battery, and power in SIM_Multicopter.cpp, applicable to any hexacopter drone.

The results show that it is feasible to operate delivery missions within a radius of 5 km with payloads of around 4 kg, while maintaining safety margins in terms of energy and performance. Beyond these values, range decreases rapidly, and it is advisable to resort to relieve or intermediate charging strategies during deliveries. The work provides a basis for planning routes, sizing charging infrastructure, and, ultimately, making informed decisions about the deployment of drones in the city.

Future work

Simulations have demonstrated the technical feasibility of the system in a controlled setting; the next step is to bring it closer to real-world operation. The research and development lines that would allow the prototype to evolve into a fully commercial solution are described below.

Collaborative fleet planning

So far, optimization has focused on a single drone, but real-world operations will involve dozens of UAVs sharing both airspace and charging stations. Order allocation algorithms will need to be developed that can allocate each package to the most suitable drone based on the delivery time slot, available cargo, battery level, and potential air congestion.

Dynamic planning and weather factors

Due to the simulator's conditions, the weight cannot be dynamically modified, and we have always worked with the same weight, 6 kg, since if the mission is successful with the highest supported weight, it will also be successful with a lower weight. In practice, the drone does not maintain a constant payload throughout the route: each delivery progressively decreases the payload and, with it, the energy demand. To reflect this realistic behavior, the planner must be able to dynamically adjust the vehicle's weight and recalculate in real time the expected consumption, battery margins, and optimal cruising speed. This continuous update of the mass model will allow for better utilization of the available range and, if necessary, decide whether an intermediate recharging stop or a change of trajectory is required.

Added to this is the influence of the urban climate, where gusts of wind and turbulence between buildings alter fuel consumption and flight stability. Integrating meteorological data will make it possible to generate alternative routes when conditions jeopardize mission safety. Similarly, the system logic must combine updated

weight information, meteorological conditions, and airspace restrictions to always select the most efficient and safe trajectory, ensuring that the drone completes its mission within established energy and regulatory limits.

Precision delivery

For last-mile logistics to be truly convenient for the customer and efficient for the operator, the drone must be able to deposit cargo with an accuracy of a few centimeters, whether on the balcony of a tall building or in a smart locker located at street level. Achieving this requires equipping the aircraft with a perception system that can deliver the package to the correct location. This would require vision-based localization algorithms such as OF (optical flow) or SVO (stereo visual odometry).

First, robust detection of the delivery point must be achieved. It is proposed to place visual or infrared beacons on the landing platform. On board, a reader must be able to validate the beacon and then deposit the package.

Second, the height estimate during the final meters must be very precise. Measures should be taken to smooth the descent profile, prevent impacts, and land with the necessary delicacy to avoid damaging the cargo or the delivery surface.

In addition, environmental perception can be enhanced with a small neural network trained to recognize unexpected obstacles (wires, branches, birds) and activate evasive maneuvers.

Finally, the system must provide feedback to the user: a notification or message on the mobile app will confirm delivery via an image captured at the time of delivery. For the operator, camera and telemetry logs will be used to evaluate the actual accuracy achieved and improve control models for future missions.

Advanced battery modeling and charging logistics

The simulations used a linear discharge model, sufficient for an initial analysis, but far from the electrochemical reality of batteries. Incorporating a more precise battery model that can take into account other factors (temperature, degradation cycles, etc.) would allow for the prediction of battery voltage and residual capacity.

Based on this, two operational strategies could be evaluated: rapid charging stations (15-20 min) installed on strategic roofs and battery-swap systems that quickly replace the drone's battery. Simulating both alternatives will allow for detailed measurements of effective service time, energy cost per kilowatt-hour, and cell degradation, accurately indicating when it is most cost-effective to increase the fleet or expand the charging infrastructure.

Fault management and cybersecurity

The social acceptance and certification of a drone delivery system depend, to a large extent, on its ability to survive anomalous situations without causing harm to people or property. To achieve this, progress must be made on two complementary axes: fault tolerance and cyber defense.

At the operational level, a battery of fault injection tests is proposed to cover the most critical scenarios: loss of a rotor, severe degradation of the control link, total blackout of inertial sensors, or battery overheating. Each test must verify that the drone activates the appropriate response (auto-hover, Return-to-Home, etc.) with alternative routes or controlled descents to the nearest safe zone, and that this maneuver is executed within acceptable energy and time limits.

The system also requires cybersecurity hardening in accordance with the guidelines and requirements agreed upon by companies and governments. To achieve this, the following series of actions that the drones used should have would be beneficial:

- Secure boot and firmware signing: the drone must cryptographically verify the flight image before executing it, preventing malicious payloads.
- Telemetry and control encryption: all MAVLink/UDP traffic will be encapsulated in TLS or Datagram TLS links, with key management and periodic rotation.
- Reducing the attack surface: disabling unused MAVLink ports, limiting commands, etc.

Contribuciones Personales

Francisco Mollá Astrar

Por medio del desarrollo de este Trabajo de Fin de Grado he aprendido y adquirido competencias sobre un tema ajeno a mí con carácter previo a la realización del mismo. Permitiéndome, tomar un primer acercamiento en profundidad a la literatura científica y enfrentar problemas que no en todos los casos han tenido soluciones triviales. Seguidamente, expondré cuáles han sido mis aportaciones personales a cada una de las partes del trabajo, junto con los elementos que considero más relevantes a detallar de las mismas.

Para comenzar a tomar decisiones y conocer la situación actual de la tecnología y los desafíos que afronta, realizamos conjuntamente una importante etapa inicial de lectura de artículos científicos, blogs, noticias y diversidad de otras fuentes para definir con mayor claridad los objetivos del trabajo. Esto me permitió entender adecuadamente la complejidad de la cuestión y los múltiples desafíos que afronta, más allá de los tecnológicos, entre los que cabe destacar el legal, pues actualmente hay muchos impedimentos legales al vuelo de drones. Lo más importante a extraer fue que aunque ya existen proyectos piloto que empiezan a evaluar la viabilidad en la práctica de estos modelos. Un simulador como el nuestro, que de manera controlada permite realizar multitud de pruebas, es una herramienta útil que permite realizar pruebas de concepto con facilidad.

Otro punto importante en esta línea, fue la investigación de distintos drones que realizamos de manera conjunta tanto mi compañero como yo, permitiéndonos conocer el mercado actual de drones y las mejores opciones para finalmente elegir el modelo DJI Matrice 600 PRO. Para llegar a esta conclusión comparamos variedad de modelos teniendo en cuenta sobre todo la adecuación de sus parámetros a la tarea que nos ocupaba, centrándonos en su batería y capacidad de carga. Toda esta información en muchos casos se encontraba muy dispersa y no había regularidad en los datos presentados entre diferentes modelos.

Para continuar, mi compañero y yo dirigimos nuestra atención a conseguir realizar la simulación. Es decir, realizar la simulación y ser capaces de conectar el dron simulado con la herramienta utilizada para la simulación, lo cual no fue tarea fácil. Inicialmente, íbamos a utilizar el simulador Mission Planner, nos costó tiempo poder conectar ambos (ArduPilot y Mission Planner), a pesar de que son proyectos con un

fuerte vínculo. Finalmente, conseguimos conectarlos por medio de la configuración de un puerto UDP y realizar las primeras pruebas.

Después, nos vimos obligados a utilizar otro simulador por la escasa configuración que ofrecía Mission Planner de parámetros vitales para nosotros como la batería. Comenzamos a realizar pruebas con QGroundControl la que sería nuestra opción final. Aunque, de nuevo, tuvimos una etapa inicial de complicaciones para realizar la conexión de la simulación (ejecución de sim_vehicle.py que se lanza por medio de MAVLink). Finalmente, de nuevo, al configurar adecuadamente los puertos de envío y escucha la conexión se estableció y pudimos comenzar a trabajar de manera estable.

Una vez conseguimos poner a funcionar el simulador, comenzamos a realizar pruebas de distintas rutas con distinta longitud. En esta etapa, el mayor problema fue conocer el modelo de alturas del simulador para finalmente configurar las altitudes de manera que sean relativas a la altura del terreno y no absolutas. Esto nos permitió finalmente realizar rutas pudiendo garantizar su finalización. Debido a que si en la simulación la altura del dron es inferior a la del terreno, este se detenía y debíamos volver a lanzar la simulación. Para ello, desde los scripts que fuimos realizando tuvimos que cambiar la forma de creación de los puntos de ruta para que se ajustasen a la altura sobre el terreno utilizando como referencia AMSL (above mean sea level).

Después, partiendo de un primer script hecho por mi compañero para la realización de rutas con mayor facilidad, creamos varias rutas y comenzamos a realizar misiones para llevar a cabo un primer acercamiento a la extracción de datos. Para ello, en esta etapa temprana grabamos la realización de las misiones. Con carácter posterior, anotamos los datos más relevantes que se concretaron en un análisis más exhaustivo posterior.

Para simplificar la tarea de la simulación y reducir los pasos cree un pequeño script .bat. Con su ejecución se llama a otro archivo .ps1 que realizaba tanto la acción de ejecutar el simulador QGroundControl, como lanzar la simulación y conectar ambos automáticamente. Por medio de estos archivos reduje la cantidad de pasos que debían darse para ejecutar las simulaciones, reduciendo a un simple doble clic el arrancar la simulación.

Finalmente, desarrollé una pequeña interfaz capaz de realizar las acciones principales que abarcan configurar y crear una misión. La interfaz básicamente presenta un mapa sobre el que se puede navegar y en el cual se pueden añadir puntos de ruta a golpe de clic. Se puede configurar tanto la altura como la espera que ha de realizar el dron en cada punto de ruta. Además, como un añadido a dicha interfaz se creó un script en python que ordena los puntos de ruta para minimizar la distancia total recorrida haciendo uso de la fórmula Haversine. Al terminar su ejecución, crea un archivo .plan listo para utilizarse desde el simulador y cargar directamente la misión creada.

Los desafíos técnicos que supuso esta parte fueron, conectar adecuadamente la SDK de Mapbox con Unity, debido a las versiones elegidas la compatibilidad no fue directa. Aunque, tras las configuraciones iniciales todo funcionó con gran fluidez. Otro desafío, fue poder ejecutar cómodamente scripts python desde Unity, lo que no fue trivial. Consiguiendo ejecutarlo inicialmente desde el editor (entorno de desarrollo de Unity) y, finalmente por otros medios desde la *build* (ejecutable final).

Jorge Rubio Carrizo

El desarrollo de este Trabajo de Fin de Grado me ha permitido adquirir competencias técnicas avanzadas y, al mismo tiempo, poner a prueba mi capacidad de planificación y resolución de problemas. A continuación, detallo las aportaciones personales más relevantes, estructuradas en las distintas fases del proyecto, junto con las dificultades superadas y los aprendizajes derivados de cada una de ellas.

Para obtener resultados significativos era imprescindible escoger un modelo de dron cuyo comportamiento fuese representativo con la realidad. Mi compañero y yo nos encargamos de realizar una búsqueda por el mercado y finalmente, tras un análisis comparativo de especificaciones técnicas, nos decantamos por el DJI Matrice 600 PRO. Esta elección requirió una búsqueda exhaustiva de:

La fase de recogida de información me permitió familiarizarme con fuentes de datos tales como manuales de fabricante, bases de datos de proyectos open-source y artículos académicos para asi poder evaluar de manera crítica la fiabilidad técnica del modelo del dron.

Después, tanto mi compañero como yo, nos centramos en la conexión del entorno con el dron. Posiblemente, la tarea que más problemas dio en el desarrollo del proyecto. Al principio, sim_vehicle.py y QGroundControl ni siquiera llegaban a establecer enlace: no se recibía ningún flujo MAVLink y la misión quedaba bloqueada en el arranque. Tras varias pruebas descubrí que el problema no era pérdida de paquetes, sino un desajuste en la configuración de puertos. Al forzar ambos programas a comunicarse por UDP en el puerto 14550, que es el canal predeterminado de ArduPilot, la conexión se estableció de forma estable y los waypoints comenzaron a ejecutarse sin interrupciones.

Superado este obstáculo, abordé el cambio de parámetros en SIM_Multicopter.cpp (masa total, capacidad de la batería, limitadores de corriente, etc.) con el fin de replicar fielmente la ficha técnica del M600 PRO. Cada modificación se validó mediante pruebas básicas para asegurar que el empuje máximo y la tasa de subida coincidieran con los datos de fábrica.

Tras todo esto, el siguiente paso que realicé fue la creación de los archivos .plan que QGroundControl emplea para cargar misiones. Partiendo de la sintaxis JSON propia de este formato diseñé un script de Python que, a partir de un conjunto mínimo de parámetros (longitud del trayecto, número de entregas, altitud de crucero y velocidades verticales), construye automáticamente la secuencia completa de waypoints, tomas de tierra intermedias y retorno al punto de origen.

La automatización ha supuesto un ahorro de tiempo grande frente al método manual de creación de rutas en QGroundControl, ya que había que gestionar manualmente dónde poner un punto de ruta y añadir altura, tiempo de espera, etc. De esta manera, ya se generaba el archivo .plan con los puntos requeridos, etc. Además, la estructura del código facilita modificar variables como patrones de espera o cambios de altitud sin necesidad de reescribir la lógica.

Las misiones las diseñé con una serie de características específicas para tener datos concretos de cómo se comportaba el dron en diferentes ámbitos. Lo mejor era crear tres tipos diferentes: corta, media y larga. La corta tendría 5 puntos de

entrega pero no recorrería más de 1 km. La ruta media tendría 3 puntos y unos 5 km. Por último, la ruta larga sólo tenía un punto de entrega pero la distancia recorrida llegaba hasta los 9 km. En cada una consideré que lo mejor era mantener una altura similar en las tres misiones, así como el tiempo de espera en cada punto, en este caso 30 m de altura y 30 segundos de espera.

Una vez preparadas las misiones, operé cada una de ellas y fui anotando todos los datos de telemetría (tensión, corriente, velocidad, posición y tiempo) para disponer de un histórico completo. El objetivo era contar con la información necesaria para, más adelante, estudiar el consumo y el comportamiento del dron para su análisis posterior.

Tras las primeras misiones de tipo entrega, diseñé dos experimentos muy concretos. El primero consistió en una ruta continua pensada exclusivamente para comprobar la duración real de la batería; el vuelo finalizaba cuando el nivel de carga caía por debajo del umbral de seguridad. El segundo ensayo fue una misma misión repetida con 2 kg, 4 kg y 6 kg de carga útil para observar cómo variaba el consumo en función del peso. Todos los registros generados se volcaron en hojas de Excel, donde elaboré tablas y gráficos sencillos que permitieron visualizar rápidamente la diferencia entre situaciones y extraer las conclusiones incluidas en el apartado de resultados.

Más allá de la parte técnica, este proyecto me ha servido para soltarme con herramientas y tareas que antes solo conocía por encima. He aprendido a buscar información de manera global, no quedándome con lo primero que veía, he aprendido a hacer informes en LaTeX, a recopilar resultados de forma clara y ordenada para que cualquiera los entienda y sobre todo he aprendido a buscar soluciones si algo falla y a investigar a fondo errores y problemas que surgían a menudo durante la realización del proyecto. Pasar por todo el ciclo, me ha dado mucha más seguridad para enfrentarme a problemas abiertos y me ha demostrado lo útil que es automatizar procesos y dejar todo bien documentado para que otro usuario pueda repetirlo de una manera asequible.

Bibliografía

Akhloufi, M. A., Couturier, A. y Castro, N. Unmanned aerial systems for wildland and forest fires. 2021. Disponible en https://www.researchgate.net/publication/378259224_Unmanned_Aerial_Systems_for_Wildland_and_Forest_Fires (último acceso, Mayo, 2025).

Ali, M. F., Jayakody, D. N. K. y Muthuchidambaranathan, P. Revolutionizing firefighting: Uav-based optical communication systems for wildfires. 2024. Disponible en https://www.mdpi.com/2304-6732/11/7/656 (último acceso, Mayo, 2025).

ArduPilot. Ardupilot. 2025. https://ardupilot.org/ (último acceso: Mayo, 2025).

Bayas, B. O., Castro, K. F. S. y Mera, E. Z. Red de drones autónomos utilizando una arquitectura de red para uso alternativo de levantamiento de información agrícola a pequeña escala. 2021. Disponible en https://conrado.ucf.edu.cu/index.php/conrado/article/view/1698 (último acceso, Mayo, 2025).

Brunner, G., Szebedy, B., Tanner, S. y Wattenhofer, R. The urban last mile problem: Autonomous drone delivery to your balcony. 2019. Disponible en https://ieeexplore.ieee.org/abstract/document/8798337 (último acceso, Mayo, 2025).

Daud, S. M. S. M., Yusof, M. Y. P. M., Heo, C. C., Khoo, L. S., Singh, M. K. C., Mahmood, M. S. y Nawawi, H. Applications of drone in disaster management: A scoping review. 2022. Disponible en https://www.sciencedirect.com/science/article/pii/S1355030621001477 (último acceso, Mayo, 2025).

DJI. Dji matrice 600 pro. 2025. https://www.dji.com/es/downloads/products/matrice600-pro#app (último acceso: Mayo, 2025).

Freitas, H., Faiçal, B. S., e Silva, A. V. C. y Ueyama, J. Use of uavs for an efficient capsule distribution and smart path planning for biological pest control. 2020. Disponible en https://www.sciencedirect.com/science/article/pii/S0168169919323488 (último acceso, Mayo, 2025).

González, A., Amarillo, G., Amarillo, M. y Sarmiento, F. Drones aplicados a la agricultura de precisión. 2016. Disponible en https://www.researchgate.net/publication/315988656_Drones_Aplicados_a_la_Agricultura_de_Precision (último acceso, Mayo, 2025).

Islam, N., Rashid, M. M., Pasandideh, F. y Ray, B. A review of applications and communication technologies for internet of things (iot) and unmanned aerial vehicle (uav) based sustainable smart farming. 2021. Disponible en https://www.researchgate.net/publication/349158479_A_Review_of_Applications_and_Communication_Technologies_for_Internet_of_Things_IoT_and_Unmanned_Aerial_Vehicle_UAV_Based_Sustainable_Smart_Farming (último acceso, Mayo, 2025).

Khanda, A., Corò, F., Sorbelli, F. B., Pinotti, C. M. y Das, S. K. Efficient route selection for drone-based delivery under time-varying dynamics. 2021. Disponible en https://ieeexplore.ieee.org/document/9637806 (último acceso, Mayo, 2025).

Lu, F., Jiang, R., Bi, H. y Gao, Z. Order distribution and routing optimization for takeout delivery under drone–rider joint delivery mode. 2024. Disponible en https://www.mdpi.com/0718-1876/19/2/41 (último acceso, Mayo, 2025).

Mapbox. Mapbox. 2025. https://www.mapbox.com/ (último acceso: Mayo, 2025).

Maps, G. Google maps platform. 2025. https://developers.google.com/maps?hl=es-419 (último acceso: Mayo, 2025).

Planner, M. Mission planner. 2025. https://ardupilot.org/planner/ (último acceso: Mayo, 2025).

QGroundControl. Qgroundcontrol. 2025. https://qgroundcontrol.com/ (último acceso: Mayo, 2025).

Studio, A. Android studio. 2025. https://developer.android.com/studio?hl=es-419 (último acceso: Mayo, 2025).

Unity. Unity. 2025. https://unity.com/es (último acceso: Mayo, 2025).

Wankmüller, C., Kunovjanek, M. y Mayrgündter, S. Drones in emergency response – evidence from cross-border, multi-disciplinary usability tests. 2021. Disponible en https://www.sciencedirect.com/science/article/pii/S2212420921005288 (último acceso, Mayo, 2025).

Wihbey, J. La revolución de los drones. 2017. Disponible en https://www.lincolninst.edu/es/publications/articles/la-revolucion-los-drones/ (último acceso, Mayo, 2025).

Zieher, S., Olcay, E., Kefferpütz, K., Salamat, B., Olzem, S., Elsbacher, G. y Meess, H. Drones for automated parcel delivery: Use case identification and derivation of technical requirements. 2024. Disponible en https://www.sciencedirect.com/science/article/pii/S2590198224002392 (último acceso, Mayo, 2025).