Trabajo Fin de Grado
Grado en Ingeniería Electrónica, Robótica y Mecatrónica

Planificación de caminos para sistemas multi-UAV en aplicaciones de inspección y mantenimiento

Autor: Francisco Javier Román Escorza

Tutores: Álvaro Caballero Gómez
Aníbal Ollero Baturone

Dpto. Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2023

Trabajo Fin de Grado

Grado en Ingeniería Electrónica, Robótica y Mecatrónica

Planificación de caminos para sistemas multi-UAV en aplicaciones de inspección y mantenimiento

Autor:

Francisco Javier Román Escorza

Tutores:

Álvaro Caballero Gómez

Aníbal Ollero Baturone

Dpto. Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2023

Trabajo Fin de Grado: Planificación de caminos para sistemas multi-UAV en aplicaciones de inspección y mantenimiento

Autor: Francisco Javier Román Escorza
Tutores: Álvaro Caballero Gómez
Aníbal Ollero Baturone

El tribunal nombrado para juzgar el trabajo arriba indicado, compuesto por los siguientes profesores:

Presidente:

Vocal/es:

Secretario:

acuerdan otorgarle la calificación de:

El Secretario del Tribunal

Fecha:

Agradecimientos

Con este trabajo cierro mi etapa como estudiante de grado. Durante estos años me he formado como ingeniero, pero también han moldeado la persona que soy ahora. Por eso quiero agradecer a todas las personas que han hecho esto posible y han estado conmigo este tiempo. Empezando por mis padres, que siempre han creído en mí y me han dado lo necesario para llegar hasta aquí. A mis amigos que han estado conmigo estos años, en quienes me he apoyado en los momentos duros. A mi hermana, por recordarme lo que hay después de la Universidad.

También quiero agradecer a Álvaro Caballero Gómez, por haberme guiado durante todo el proyecto. Y a Ánibal Ollero Baturone, sin el cual, este proyecto no formaría parte de otros más grandes.

Este trabajo ha sido financiado por el proyecto nacional RESISTO (2021/C005/00144188) de los fondos FEDER (Fondo Europeo de Desarrollo Regional) del Ministerio de Asuntos Económicos y Transformación Digital y por el proyecto europeo AERIAL-CORE (H2020-2019-871479).

Francisco Javier Román Escorza
Sevilla, 2023

Resumen

Dentro de la robótica actual, muchas líneas de investigación tienden al trabajo coordinado de sistemas autónomos multi-UAV (del inglés, Unmanned Aerial Vehicle) que sustituyan a operarios humanos en tareas peligrosas. Bajo este paradigma nacen proyectos como AERIAL-CORE o RESISTO, que pretenden utilizar las ventajas de los sistemas multi-UAV para la inspección de largo alcance de líneas eléctricas, mitigando riesgos para los operarios, reduciendo los costes de mantenimiento y evitando fallos en los sistemas eléctricos de distribución. Sin embargo, para que esto sea posible, se requiere disponer de métodos de planificación que permitan coordinar al sistema multi-UAV de forma segura y eficiente. En ese contexto, este trabajo se centra en el diseño, desarrollo y validación de un planificador de rutas, realizando un análisis de sus distintas funcionalidades, las cuales vienen asociadas a los requisitos impuestos por las misiones planteadas en ambos proyectos anteriormente citados.

Abstract

Within current robotics, many lines of research aim towards the coordinated work of autonomous multi-UAV (Unmanned Aerial Vehicle) systems that replace human operators in dangerous tasks. Under this paradigm, projects like AERIAL-CORE or RESISTO are born, which aim to leverage the advantages of multi-UAV systems for long-range inspection of power lines, mitigating risks for operators, reducing maintenance costs, and avoiding failures in electrical distribution systems. However, to make it possible, it is necessary to have planning methods that allow the safe and efficient coordination of the multi-UAV system. In this context, this work focuses on the design, development, and validation of a route planner, conducting an analysis of its different functionalities, which are associated with the requirements imposed by the missions proposed in both aforementioned projects.

Índice Abreviado

Índice

1 Introducción

Uno de los campos de la robótica donde más se está investigando en la actualidad es la planificación de rutas ya que es un problema intrínseco en cualquier proyecto que requiera la utilización de robótica móvil, sobre todo en misiones en la que se requiera que un sistema presente cierta autonomía. Lo cual, hace de este un campo de desarrollo imprescindible para la robótica actualmente. Sin embargo, en el presente no existe ningún planificador que pueda cubrir todo el registro posible de misiones que pueden ser encomendadas a los robots móviles. Esto se debe a que los requisitos de las rutas varían de una aplicación a otra puesto que los criterios cambian desde la optimización de la batería hasta la minimización del tiempo empleado o centrarse en dar más importancia a recorrer ciertas zonas frente a otras. Estas preferencias son algo muy específico de cada misión. Por tanto, si alguno de los escenarios presenta requisitos muy concretos, se suele optar por realizar un planificador específico para esa misión.

Debido a la importancia que tiene y a la variedad de escenarios que se pueden encontrar, existen diversas formas de plantear el planificador. Por lo que el enfoque que se le debe dar siempre tiene que estar centrado en el escenario en el que se va a trabajar y en los criterios que se quieran primar. De esta manera, a pesar de existir planificadores con un enfoque más genérico y que sean aptos para un gran abanico de aplicaciones, cuando se quiera realizar aplicaciones con especificaciones más concretas es importante preparar un planificador que se ajuste a las necesidades específicas del proyecto.

1.1 Motivación y objetivos

Ante la situación descrita previamente, existen proyectos como AERIAL-CORE y RESISTO que pretenden aprovechar las ventajas que ofrece la robótica aérea móvil para la inspección y mantenimiento de tendidos eléctricos.

AERIAL-CORE [10] es un proyecto financiado por la Comisión Europea y liderado por el GRVC Robotics Lab [5] de la Universidad de Sevilla que persigue el desarrollo de un entorno tecnológico capaz de trabajar de manera segura y conjunta con humanos, apto para realizar operaciones a grandes distancias de manera precisa. Entre estas operaciones se encuentran tareas como el trabajo en conjunto con humanos que estén realizando operaciones en torres eléctricas, instalación en el tendido eléctrico de salvapájaros o la inspección de grandes áreas de tendido eléctricos. Estas son operaciones que hasta el momento las realizaban únicamente operarios humanos, suponiendo un riesgo para los mismos (al tener que manipular de forma directa líneas de alta tensión y trabajando frecuentemente a grandes alturas) y un gran coste asociado a las medidas de prevención de riesgos necesarias para garantizar la seguridad.

Por su parte, RESISTO [9] es un proyecto nacional financiado por FEDER (Fondo Europeo de Desarrollo Regional) del Ministerio de Asuntos Económicos y Transformación Digital y liderado por e-distribución, cuyo objetivo es fortalecer la resiliencia de redes de distribución críticas ante diferentes tipos de eventos, ejemplificándolo en la mejora de las instalaciones presentes en el entorno protegido del Parque Nacional de Doñana. Para ello, pretende integrar tecnologías punteras en los sistemas de detección y alerta temprana como el uso de inteligencias artificiales, UAVs (Unmanned Aerial Vehicle), cámaras termográficas, IoT (Internet Of Things), sensores inteligentes o realidad aumentada. Con esto se quiere agilizar la detección de posibles riesgos y una mejora de la actuación, a la par que cuidar el entorno natural en el que se pretende desarrollar.

Tanto AERIAL-CORE como RESISTO poseen un objetivo común: la utilización de UAVs para inspeccionar líneas eléctricas. De este objetivo compartido nace la necesidad de disponer de un planificador de rutas

concreto que pueda calcular los recorridos necesarios para la inspección de líneas eléctricas. Además, estas misiones de inspección y mantenimiento requieren de un planificador capaz de integrar múltiples UAVs con capacidades heterogéneas en términos de velocidades de inspección y autonomía, que asegure la inspección de manera eficiente de cada uno de los segmentos de la línea eléctrica, teniendo en cuenta el consumo de batería y buscando la rapidez a la hora de ejecutar la misión en su conjunto.

Con el fin de conseguir tal planificador, se ha optado por un modelo de desarrollo modular, en el cual se tendrán en cuentas los requisitos fundamentales que debería poseer el planificador. Este ha sido diseñado por etapas que se han unificado paulatinamente en el modelo final:

1. Planificador con un único UAV.
2. Planificador con múltiples UAVs heterogéneos.
3. Implementación de restricciones por limitación de batería.
4. Implementación del efecto del viento.

1.2 Contenido

Este trabajo está divido en varios capítulos:

- **Capítulo 2: Estado del arte,** donde se analiza el estado actual y futuro de los UAVs, además de explicar el problema del TSP (Travelling Salesman Problem) y de algunos de sus métodos de resolución exactos y heurísticos.
- **Capítulo 3: Herramientas de desarrollo,** donde se muestran las herramientas software que se han utilizado para la realización de este trabajo.
- **Capítulo 4: Planificador de Rutas,** muestra las diferentes etapas del planificador y cómo plasmar las restricciones necesarias para un caso general, con un número n de torres y m de UAVs, además de ejemplificarlo con un caso sencillo conformado por 3 torres y 3 UAVs.
- **Capítulo 5: Resultados,** enseña los resultados ante distintos escenarios que pretenden mostrar el funcionamiento y la importancia de cada una de las características del planificador y una validación en entorno real con datos geográficos del centro de vuelo ATLAS.
- **Capítulo 6: Conclusiones y trabajos futuros,** donde se analizan los resultados del apartado anterior y se comentan posibles mejoras sobre las conclusiones obtenidas.

2 Estado del arte

Durante este capítulo se pretende realizar un análisis de la situación en la que se encuentran en el momento actual los UAVs y hacía donde tienden las líneas de desarrollo de estos, además de introducir uno de los problemas de planificación fundamentales en robótica, el TSP, y algunos de los métodos de resolución del mismo.

2.1 Vehículos Aéreos No Tripulados

Los UAVs son un tipo de vehículos autónomos que han tenido un gran auge en los últimos años [11]. En gran parte, este auge se debe a su gran versatilidad de desplazamiento, ya que no son tan dependientes del terreno como otros vehículos no tripulados, permitiéndoles una mayor velocidad. A pesar de ello, sí se pueden ver muy afectados por la climatología.

Por otro lado, existen distintos tipos de UAVs, como los que se muestran en la figura 2.1, lo que permite una mayor adaptación a escenarios muy diversos:

- **Ala fija.** Son aquellos UAVs que poseen una estructura típica de los aviones con alas fijas. Estos tienen un coste energético muy bajo en comparación con otros, lo que les permite recorrer distancias más largas. Sin embargo, son los que menos versatilidad poseen ya que deben estar avanzando continuamente para mantenerse en el aire y requieren una pista para despegar y aterrizar, además de poder realizar menos tipos de movimientos.
- **Multi-rotores.** Cuya principal ventaja es la versatilidad de movimiento respecto a los primeros. Por un lado, estos UAVs no requieren una pista para comenzar o finalizar el vuelo, ya que pueden realizar el despegue y aterrizaje de manera vertical, además de poder permanecer estáticos en el aire (hovering), lo que los hace ideales para realizar manipulaciones e interactuar físicamente en operaciones con otras estructuras. Sin embargo, su mayor desventaja es que tiene un consumo energético muy elevado. Lo que los limita mucho respecto a los UAVs de ala fija para recorridos de gran distancia.
- **VTOL (Vertical Take-Off and Landing).** Son UAVs que intentan unir las ventajas de los *alas fijas* y los *multi-rotores*. Para ello, poseen una estructura con alas fijas, pero a la vez, se le añaden una estructura similar a la que se pueden encontrar en los *multi-rotores*. Con esto se pretende conseguir un gran rango de vuelo, típico de los *ala fija*, pero posibilitando el despegue y el aterrizaje de manera vertical. De esta manera, se consigue eliminar la necesidad de los *ala fija* de una pista de despegue, aunque no poseen la versatilidad de movimiento que caracteriza a los *multi-rotores* y no alcanzan la eficiencia de los *ala fija*.
- **Ala batiente.** Son un tipo de *UAVs* bioinspirados que pretenden imitar el vuelo de las aves, utilizando una estructura de alas que se doblan con el fin de imitar el aleteo de las aves. Actualmente existen varias líneas de investigación abiertas que siguen esta tendencias y que esperan poder mejorar la eficiencia de los *ala fija*.

Los UAVs nacieron y llevan siendo utilizados muchos años en el ámbito militar debido a que permite realizar gran cantidad de misiones sin poner en riesgo al piloto, quien lo puede teleoperar desde alguna base segura. Sin embargo, en los últimos años se ha estandarizado su uso en el ámbito civil, siendo común su utilización por parte de la policía en algunos casos donde antes se requeriría el uso de helicópteros (con el

(a) MQ-9 Reaper, UAV de ala fija del ejercito de EEUU [15].

(b) DJI Mavic 3 Pro, UAV multi-rotor [14].

(c) Mugin-5 Pro, VTOL UAV [25].

(d) Prototipo de UAV de ala batiente [18].

Figura 2.1 Ejemplo de diferentes tipos de UAVs.

coste que conlleva) para misiones de vigilancia. En la misma línea, los UAVs (en concreto los multi-rotores) han ido sustituyendo a los helicópteros en distintos ámbitos como pueden ser rodajes aéreos de películas [19] y grabación de eventos con gran aglomeración de personas (como conciertos o fiestas) [16]. Su bajo coste, la sencillez de control y el fácil acceso a ellos, ha hecho de los UAVs (en específico de los multi-rotores) una herramienta tanto de trabajo como de ocio que ha llegado a todo tipo de campos.

En el contexto actual, los UAVs pretenden seguir siendo elemento de revolución en muchos ámbitos. Uno de los principales son los servicios en entornos urbanos, enfocando su uso en el reparto de paquetería [23] o de entrega de herramientas a operarios que trabajan a gran altura [24]. Incluso existen proyectos de taxis autónomos aéreos [4], basados en UAVs capaces de portar a varias personas entre distintos puntos de una ciudad. Todas estas posibles aplicaciones implican que se debe avanzar mucho en sistemas de seguridad y robustez de la autonomía de los UAVs, además de facilitar la interacción entre los UAVs y los humanos.

En paralelo en el ámbito civil, la Unión Europea financia varias líneas de investigación y desarrollo dentro del programa Horizon 2020 centradas en la aplicación de los UAVs para inspección y mantenimiento (I&M) de estructuras industriales mediante la utilización de robots manipuladores aéreos. HYFLIERS [6] se centra en el diseño y aplicación de robots aéreo/terrestres para la inspección y mantenimiento en refinerías de petróleo. AEROARMS [7] busca probar la viabilidad del uso en los primeros manipuladores aéreos para inspecciones no destructivas, como se puede ver en la figura 2.2. Por su parte, PILOTING [1] es un proyecto que no pretende utilizar manipuladores aéreos, pero sí persigue el empleo de UAVs para la inspección y mantenimiento de infraestructuras industriales, realizando pruebas en tres estructuras de gran valor logístico: refinerías, puentes y túneles.

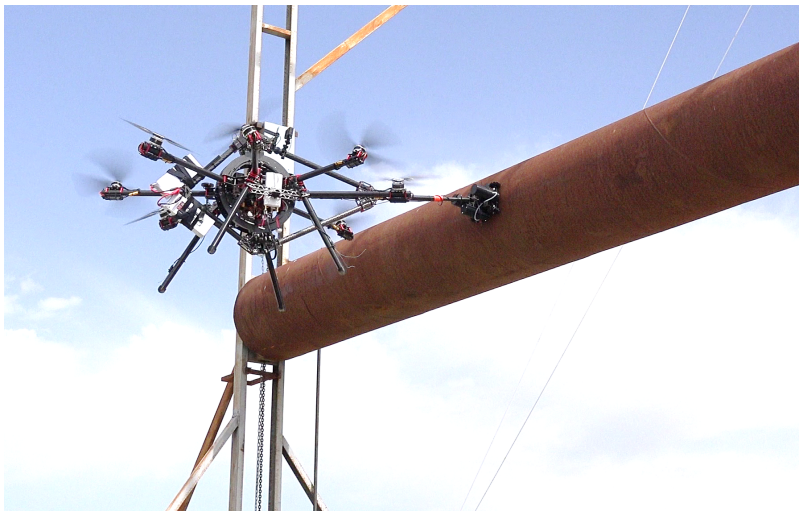


Figura 2.2 Prueba de robot manipulador aéreo ante una inspección no destructiva.

A día de hoy los UAVs están integrados en muchos sectores de la sociedad y en los próximos años se van a asentar en muchos otros ámbitos, intentando facilitar la vida y evitar riesgos y costes innecesarios.

2.2 Travelling Salesman Problem

Ante el escenario que se nos plantea en la actualidad, muchas de las aplicaciones en las que se pretende utilizar UAVs necesitan que sean autónomos para maximizar la productividad. Uno de los principales requisitos para ello es establecer la ruta que deban seguir y que puedan calcular nuevas rutas ante elementos imprevistos que afectasen a la original. Esto no es un problema trivial, si no que es un campo de estudio ampliamente explorado desde varios puntos. Una de las formas de enfrentarse a este problema más comunes es partiendo del TSP e ir particularizando las restricciones para el escenario concreto de la misión.

El TSP [12] es un problema de optimización clásico que plantea el caso de un vendedor ambulante que necesita visitar distintas ciudades cumpliendo una serie de requisitos:

- Comenzar y terminar el recorrido en la ciudad en la que se encuentra inicialmente.
- Recorrer la menor distancia posible.
- Visitar cada una de las ciudades una única vez.

Observando únicamente el planteamiento original, puede parecer un problema muy concreto. Sin embargo, abstrayendo el problema y entendiéndolo como un grafo donde las ciudades son los nodos, los caminos entre estos las aristas y la distancia del camino el coste asociado a las aristas, este problema (y las diferentes formas de resolverlo) se puede aplicar en ámbitos muy diversos y con gran importancia en la sociedad actual.

Si se toman los nodos como puntos de soldadura y quien debe visitarlos es un robot industrial con una herramienta de soldadura como terminador, las soluciones del problema son aplicables a una gran cantidad de industrias como la automovilística. Otro ejemplo del uso del problema del TSP puede ser cómo conectar todos los puntos de comunicación de una ciudad, o zonas incluso más grandes, con el fin de usar la mínima cantidad de cable posible.

Uno de los planteamientos más conocidos de los que derivan del TSP, es el VRP [20] (Vehicle Routing Problem). En este problema tenemos varios vehículos de reparto y diferentes puntos de entrega y se busca las diferentes rutas que deberán tomar cada uno de los vehículos disponibles para realizar el reparto de la manera más rápido posible, cumpliendo con la capacidad máxima de cada vehículo.

Resulta evidente que la habilidad de poder resolver el TSP posee un gran valor para muchas disciplinas de la ingeniería y la industria. No obstante, existe un gran problema ya que la resolución del TSP no es algo trivial. Está demostrado que el TSP es un problema tipo NP-hard(Nondeterministic Polynomial-time Hard), lo que dificulta obtener el resultado óptimo en gran cantidad de ocasiones, puesto que el tiempo necesario para resolver estos tipos de problemas puede crecer de manera desproporcionada al añadir pocos elementos o incluso al variar ligeramente los parámetros iniciales. En el caso del VRP, al ser un caso generalizado del TSP

que incluye un número m de vehículos y una capacidad máxima Q de cada uno, se añaden más restricciones al problema NP-hard, provocando un aumento del coste computacional que es difícil de gestionar.

Es clara la importancia tanto de obtener la solución a estos problemas como de conseguirlo en un tiempo razonable para cada una de las posibles aplicaciones. Por ello se ha intentado desarrollar distintos métodos desde una gran variedad de enfoques, todos centrados en minimizar el tiempo de computo necesario para obtener la solución.

Existen dos grandes enfoques para resolver este problema. Por un lado se encuentran los algoritmos exactos, que buscan encontrar siempre la solución óptima. En oposición a estos, están los métodos heurísticos que priman minimizar el tiempo de computo frente a la solución óptima, asegurando una solución cercana a la ideal, pero no pueden asegurar ofrecer la óptima.

2.2.1 Algoritmos exactos

Estos algoritmos persiguen encontrar la solución óptima al problema TSP, a la vez que suelen conllevar mucho tiempo de cómputo. Algunos de los métodos exactos son:

- **Fuerza bruta.** Es decir, calcular todas las permutaciones posibles. Esto asegura la solución exacta, pero el tiempo de computo es de orden O(n!). No es viable utilizarlo en casi ningún caso y se han buscado métodos exactos que mejoran el tiempo de cómputo ya que a partir de 20 ciudades, se necesitan aproximadamente 2.4×10^{18} ciclos de cómputo.
- **Basados en *Branch-and-bound*.** Empiezan desde el punto de partida y van buscando ciudad a ciudad, cuál es la ruta más corta. Es viable hasta 50 ciudades.
- **Basados en programación dinámica**. Uno de los primeros fue el **método de Held-Karp** que consigue mejorar el cómputo a $O(n^2 2^n)$, lo que supone 419430400 ciclos para 20 ciudades. En esta línea, el método propuesto por *Ambanis et al.* consigue reducir el orden a $O(1.728^n)$, 56348 ciclos para las mismas ciudades.

2.2.2 Métodos heurísticos

Debido a que no hay ningún método exacto que asegure encontrar la solución óptima de manera rápida, en paralelo a los algoritmos exactos se han desarrollado métodos heurísticos. Estos pretenden seguir unas ciertas reglas y patrones que no aseguran la solución exacta aunque sí aseguran encontrar una solución cercana a ella, así como tiempos de cómputo mucho menores a los de algoritmos exactos. En esto reside la fortaleza de estos métodos. Dan soluciones muy cercanas a la exacta en un cantidad de tiempo que se ajustan muchísimo mejor a los tiempos requeridos en aplicaciones reales.

Algunos de estos métodos son los Algoritmos Genéticos (GA), los métodos heurísticos constructivos (como los vecinos cercanos) o algoritmos de optimización de colonia de hormiga.

2.2.3 Formulación

Muchos de los métodos previamente propuestos para la resolución del TSP utilizan la programación lineal de enteros (ILP) para obtener una formalización del problema del que poder partir. Existen distintas formalizaciones del problema del TSP, siendo una de las más conocidas la formulación de **Dantzig-Fulkerson-Johnson (DFJ)**. Esta formulación enumera las ciudades desde 1 hasta n, mientras que utiliza la variable c_{ij} para marcar el coste que tiene el camino desde la ciudad i a la j y le asocia una variable binaria que marca si se realiza o no, x_{ij}. La formalización del TSP con la formulación DFJ quedaría como se muestra en las ecuaciones 2.1-2.5:

$$\min_{x_{i,j}} \sum_{\{i,j\}\in\mathcal{N}, i\neq j} c_{i,j}x_{i,j} \tag{2.1}$$

$$s.t. \sum_{j=1,\ldots,n} x_{ij} = 1, \qquad i = 1,\ldots,n; \tag{2.2}$$

$$\sum_{i=1,\ldots,n} x_{ij} = 1, \qquad j = 1,\ldots,n; \tag{2.3}$$

$$\sum_{\{i,j\}\subset S} x_{ij} \leq |S| - 1, \qquad S \subset \mathcal{V}, 2 \leq |S| \leq n-2 \tag{2.4}$$

$$x_{ij} \in \{0,1\}, \qquad i,j = 1,\ldots,n, i \neq j \tag{2.5}$$

donde la función objetivo 2.1 busca minimizar la longitud del camino, las restricciones 2.2 y 2.3 restringen que solo se llegue y se salga una única vez de cada ciudad, mientras que la restricción 2.4 evita que se creen subrutas como las que se ven en la figura 2.3, asegurando una única ruta que no deje desconectados varios grupos de ciudades. Típicamente esta restricción se consigue iterando, obteniendo una primera solución y comprobando si existe alguna subruta. Si existe, se indica que esa solución no es válida y se vuelve a ejecutar el algoritmo.

(a) Solución con una única ruta.

(b) Solución con 2 subrutas.

Figura 2.3 Ruta vs subruta.

3 Herramientas de Desarrollo

En esta parte del trabajo se van a nombrar y explicar las distintas herramientas que se emplean en el desarrollo del planificador, defendiendo el lenguaje en el que se ha desarrollado, mencionando algunas APIs (Application Programming Interface) integradas en el mismo que le permiten obtener más información y nombrando los modelos que se asumen y utilizan para poder obtener una estimación del consumo de los UAVs y poder calcular distancias sobre la superficie de la Tierra.

3.1 Matlab y optimizador *intlinprog*

Para la realización del planificador se ha optado por el lenguaje de programación MATLAB (en su versión R2023a) y el propio entorno de trabajo que proporciona. Esta decisión se fundamenta principalmente en la facilidad y velocidad que proporciona MATLAB para la depuración de algoritmos al ser un lenguaje interpretado, además de poseer herramientas de optimización potentes como *intlinprog*. Esta función está pensada para la resolución de problemas MILP (Mixed-Integer Linear Programming) y requiere una formulación muy específica, enfocada en encontrar el mínimo de la función objetivo $f^T x$, como la que se ve en las ecuaciones 3.1-3.5.

$$min_x(f^T x) \tag{3.1}$$
$$s.t.:$$
$$x(intcon) \in \mathbb{Z} \tag{3.2}$$
$$Ax \leq b \tag{3.3}$$
$$A_{eq}x = b_{eq} \tag{3.4}$$
$$l_b \leq x \leq u_b \tag{3.5}$$
$$\tag{3.6}$$

Al estar pensado para resolver problemas tipo MILP, se puede asemejar de manera sencilla a la formulación DFJ (que es ILP) del TSP (en el siguiente capítulo 4 veremos el planteamiento de cada uno de los requisitos específicos del planificador). La principal diferencia entre ambas formulaciones se encuentra en que la formulación ILP únicamente trabaja con números enteros. Por tanto, si queremos utilizar *intlinprog* para resolver este problema debemos asegurarnos de que la variable x sea un valor entero, concretamente un valor binario utilizando estas especificaciones:

- El parámetro *intcon* de la restricción 3.2 guarda qué elementos del vector x deben ser números enteros.
- Limitar x entre el 0 y el 1 mediante la restricción 3.5.

Al utilizar estas dos especificaciones de manera conjunta, se fuerza a que los valores guardados en x solo puedan ser 0 ó 1. Por otro lado, se necesitaría enumerar todos los caminos entre los nodos, obteniendo 2 caminos entre los nodos i y j: el camino de i a j y el de j a i. Este número asocia el elemento k-ésimo del vector x al camino con el número k. Para el caso del mapa de la figura 3.1, la lista de los posibles caminos quedaría como se muestra en la tabla 3.1.

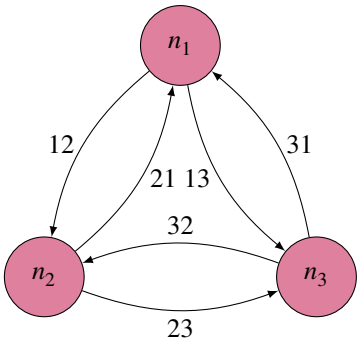


Figura 3.1 Mapa para ejemplo del TSP.

Tabla 3.1 Numeración de caminos posibles.

camino	identificador
1-2	1
1-3	2
2-1	3
2-3	4
3-1	5
3-2	6

Tabla 3.2 Matriz A_{eq} para salidas de los nodos.

Caminos	1-2	1-3	2-1	2-3	3-1	3-2
nodo 1	1	1	0	0	0	0
nodo 2	0	0	1	1	0	0
nodo 3	0	0	0	0	1	1

Si a la vez guardamos en cada componente de f la distancia del camino respectivo, la ecuación $f^T x$ dará la distancia total recorrida. Como f no puede variar, el algoritmo solo puede modificar los valores de x para poder minimizar la función 3.1, cambiando los caminos que recorrerá.

Por otro lado, utilizaremos las ecuaciones 3.3 y 3.4 para formalizar el resto de restricciones necesarias. Por ejemplo, si quisiéramos aplicar la restricción de que cada nodo solo sea visitado una única vez en el mapa 3.1, tendríamos que crear las matrices para la ecuación 3.4 para que en cada nodo haya un camino que entre y otro que salga.

En primer lugar vamos a ver cómo se forzaría a que todos los nodos tenga un camino que salga de ellos, utilizando una matriz A_{eq} de 3 filas y 6 columnas para que sean el mismo número de filas que de nodos y el mismo número de columnas que de caminos. De esta manera, A_{eq} tendría que quedar como en la tabla 3.2, mientras que b_{eq} tendría que ser un vector columna de 3 componentes todas a 1, para que solo pudiera haber un camino que sale de cada nodo y cumplir la restricción de igualdad 3.4.

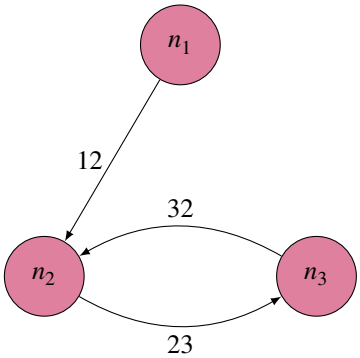


Figura 3.2 Solución incompleta al TSP.

En segundo lugar, para terminar de completar las condiciones necesarias para pasar una única vez por cada nodo, faltaría añadir la condición que obligue a que a cada nodo llegue un único camino de entrada. Si no añadiésemos esa condición, podrían salir soluciones válidas con respecto a las restricciones establecidas, como la de la figura 3.2, en la que se puede ver que el nodo 3 estaría siendo visitado dos veces, además de no poder iniciar y terminar la ruta en el mismo nodo. Por tanto, para que se tenga en cuenta esta restricción habría que aumentar las matrices A_{eq} y b_{eq}, añadiéndole una fila por cada restricción nueva a cada nodo, quedando la matriz final A_{eq} como se muestra en la tabla 3.3.

Tabla 3.3 Matriz A_{eq} para una única visita a los nodos.

Caminos	1-2	1-3	2-1	2-3	3-1	3-2
nodo 1 salida	1	1	0	0	0	0
nodo 2 salida	0	0	1	1	0	0
nodo 3 salida	0	0	0	0	1	1
nodo 1 entrada	0	0	1	0	1	0
nodo 2 entrada	1	0	0	0	0	1
nodo 3 entrada	0	1	0	1	0	0

3.2 Modelos de consumo de batería

Uno de los aspectos más restrictivos dentro de la robótica móvil es el consumo energético, ya que por la naturaleza de este campo es imprescindible el uso de baterías para la alimentación de los sistemas. Esto supone un gran reto a afrontar por diversos motivos. En primer lugar, las baterías tiene un peso considerable para los estándares en robótica aérea, lo que implica que no se pueden poner baterías de gran peso en los UAV porque el vehículo tendría que destinar gran parte de la energía extra que pueden aportar baterías de mayor tamaño a mover el peso extra de la propia batería. En segundo lugar, al no poder utilizar baterías de gran tamaño la energía que se le puede aportar al UAV es limitada. Lo que implica que el rango de acción va a estar estrictamente marcado por esto, siendo un aspecto vital a la hora de planificar rutas el tener en cuenta el consumo energético de los UAVs.

Justamente por esto, una de las limitaciones más restrictivas que debe cumplir el planificador es que ninguno de los UAVs pueda consumir más energía de la que la batería pueda ofrecer. Para poder cumplir este requisito es imprescindible saber la energía que consume cada uno de los posibles caminos. Con el fin de estudiar el gasto energético de los caminos se considera que todos los movimientos que realicen los UAVs van a ser suma de los diferentes movimientos puros (ascender, descender, avanzar y vuelo a punto fijo) y se utilizarán cuatro modelos (uno por cada movimiento puro) que nos ofrecerán la potencia eléctrica que requiera el UAV para cada movimiento. Cada uno de estos modelos relaciona los parámetros propios de cada nave como la masa, el número de rotores y otros parámetros aerodinámicos con la potencia necesaria que requeriría la nave para alcanzar al velocidad propuesta.

Una vez obtenida la potencia eléctrica que se va a utilizar, se obtiene el consumo de la batería modelando esta como una fuente ideal de voltaje constante y aplicando la fórmula de la potencia eléctrica ($P_e = V_e I_e$), estimando así la intensidad que debe ceder la batería para cada desplazamiento. Si además se considera la velocidad a la que se deber realizar cada desplazamiento (el cuál es un parámetro a determinar para el vuelo) y la distancia a desplazarse, sacamos el tiempo de ejecución, con lo que se puede saber la capacidad de batería (mAh) que se consumirá en cada movimiento.

3.3 Obtención del viento

Típicamente, a los planificadores de vuelo se les da como parámetros las velocidades a las que se desea el desplazamiento con respecto al suelo. Sin embargo, los modelos de consumo mencionados en la sección anterior estiman la potencia en función de la velocidad respecto al viento, ya que esta velocidad es la que genera la fuerza de sustentación que permite a los UAV mantenerse y desplazarse en el aire. Si no existe viento, la velocidad respecto al suelo (v_s) es la misma que la velocidad respecto al aire (v_a). Si por el contrario, la velocidad del viento no es nula, sería necesario calcular el modulo de la velocidad resultante mediante la

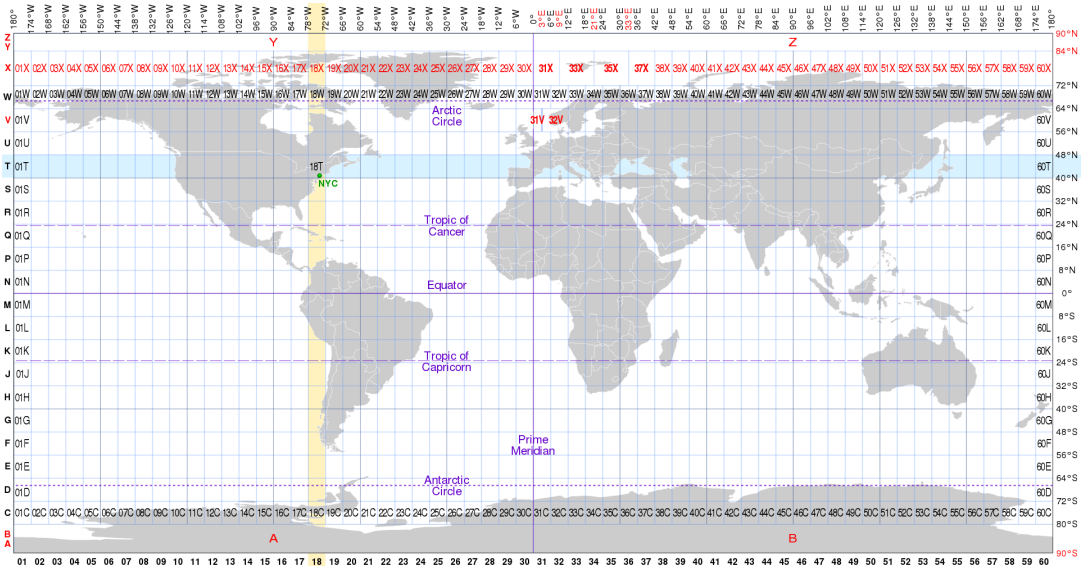


Figura 3.3 Mapa UTM.

ecuación 3.7, donde α es el ángulo entre ambas velocidades.

$$v = \sqrt{v_s^2 + v_a^2 - 2v_s v_a cos(\alpha)} \tag{3.7}$$

Para poder calcular esto, se necesita el módulo y la dirección de la velocidad del viento en donde se quiera realizar la planificación. Estos datos se obtienen mediante la utilización de la API (Application Programming Interface) pública que facilita Open Weather Map [22] al proporcionar las coordenadas de la misión.

3.4 Actualización de elevación del terreno

A la hora de preparar misiones en la realidad, es común el uso de *Google Earth Pro* para extraer las coordenadas de la base de datos de Google de las líneas eléctricas que se debe inspeccionar. Para ello se utiliza un formato de texto específico del programa (*.kml), que puede ser utilizado fácilmente por otros programas o leerse al ser un fichero de texto. Sin embargo, esta herramienta no permite exportar la elevación del terreno, facilitando únicamente la latitud y la longitud. La ausencia de la elevación supone una gran perdida de información que puede originar errores durante la misión como el cálculo erróneo de la distancia entre torres o que los UAVs no alteren su altura, pudiendo provocar choques contra el terreno o que las cámaras del UAV no pueda obtener las imágenes deseadas. Para evitar esos problemas se utiliza la API pública que proporciona Open Topo Data [8], la cual ofrece la elevación de cualquier punto de Europa al darle las coordenadas. Para esto, se basa en la base de datos de la Unión Europea *EU-DEM* [3]. Esta base se datos posee una resolución de 25 m^2 y una precisión de $\pm 7m$.

3.5 Sistema de coordenadas universal transversal de Mercator

Existen multitud de sistemas que permiten dividir y representar la superficie de la Tierra en distintas partes con la finalidad de identificar y transmitir ubicaciones importantes, siendo el más extendido el sistema de coordenadas geográficas que divide la tierra según la latitud y la longitud. Sin embargo, este sistema presenta ciertas desventajas a la hora de calcular distancias, propias de intentar representar una superficie esférica en un plano bidimensional, como una mayor distorsión en los lugares más próximo a los polos. Para intentar solucionar esto se creó el sistema UTM (Universal Transverse Mercator) que divide la superficie terrestre en áreas rectangulares casi uniformes como se muestra en la figura 3.3, permitiendo trabajar con distancias euclídeas de manera sencilla. Cada una de las áreas en las que divide este sistema la superficie de la Tierra proporciona un sistema de referencia cartesiano propio, disminuyendo así los errores por distorsión previamente mencionados.

4 Planificador de rutas

Una vez explicadas las herramientas que van a ser utilizadas, en este capítulo se va a mostrar cómo se aplican estas al planificador. Este va a consistir en una variante del TSP donde los nodos serán las torres del tendido eléctrico, los segmentos a inspeccionar se tomarán como caminos que se tienen que visitar y el coste de cada camino no será la distancia entre nodos, si no el tiempo que se necesita para recorrerlo.

Recapitulando los requisitos que debe de tener el planificador, tanto los genéricos como los específicos para la inspección y mantenimiento de líneas eléctricas, el planificador debe asegurar:

- Rutas para sistemas multi-UAVs heterogéneos.
- Cada UAV empiece y termine su ruta en su propia base.
- Cada vez que un UAV entre en un nodo, debe salir de él.
- Cada UAV debe ser capaz de realizar su ruta sin consumir toda su batería.
- Minimizar el tiempo total de la misión.
- Cada UAV debe realizar su ruta lo más rápido posible.
- Considerar el efecto del viento para realizar la planificación.

Estos requisitos deben ser formalizados mediante una formulación MILP para poder aplicarlos al optimizador *intlinprog* de MATLAB. Para ello, primero se debe definir una serie de elementos:

- Número de UAVs: m
- Número de torres y base: n
- Base de los UAVs: nodo 0
- Conjunto de UAVs: $\mathscr{D} = \{x | x \in \mathbb{Z}, 1 \leq x \leq m\}$
- Conjunto de torres: $\mathscr{T} = \{x | x \in \mathbb{Z}, 1 \leq x \leq n-1\}$
- Conjunto de nodos (torres y base): $\mathscr{N} = 0 \cup \mathscr{T} = \{x | x \in \mathbb{Z}, 0 \leq x \leq n-1\}$
- Conjunto de líneas a inspeccionar: $\mathscr{G} = \{(i,j) | i,j \in \mathscr{N}\}$

A partir de estos elementos, podemos formular el problema de optimización con las ecuaciones 4.1-4.11. Estas son las que se van a utilizar a lo largo de las distintas secciones de este capítulo para explicar cómo construir las matrices A_{eq}, b_{eq}, A y b (ver Sección 3.1) para un caso general. Además, se va a tomar un caso simple con 4 nodos y 2 UAVs, como el que se muestra en la figura 4.1, para ejemplificar las explicaciones del caso general y mostrar las matrices resultantes.

$$\min_{x_{i,j|d},\sigma_{rp}} \sum_{\{i,j\}\in\mathscr{N},i\neq j,d\in\mathscr{D}} c_{ij|d}x_{ij|d} + \sum_{\{r,p\}\in\mathscr{D},r\neq p} \sigma_{rp} \tag{4.1}$$

$s.t.$

$$\sum_{i\in\mathscr{T}} x_{i0|d} = 1 \qquad \forall d \in \mathscr{D} \tag{4.2}$$

$$\sum_{j\in\mathscr{T}} x_{0j|d} = 1 \qquad \forall d \in \mathscr{D} \tag{4.3}$$

$$\sum_{i\in\mathscr{T}} x_{ij|d} - x_{ji|d} = 0 \qquad \forall j \in \mathscr{T}, \forall d \in \mathscr{D} \tag{4.4}$$

$$\sum_{\{i,j\}\in\mathscr{N},i\neq j} E_{ij|d}x_{ij|d} \leq 1 \qquad \forall d \in \mathscr{D} \tag{4.5}$$

$$\sum_{d\in\mathscr{D}} x_{ij} + x_{ji} \geq 1 \qquad \forall \{i,j\} \in \mathscr{G} \tag{4.6}$$

$$\sum_{d\in\mathscr{D}} x_{ij|d} \leq 1 \qquad \forall \{i,j\} \in \mathscr{G} \tag{4.7}$$

$$\sum_{d\in\mathscr{D}} x_{ji|d} \leq 1 \qquad \forall \{i,j\} \in \mathscr{G} \tag{4.8}$$

$$\sum_{\{i,j\}\in\mathscr{N},i\neq j} (c_{ij|r}x_{ij|r} - c_{ij|p}x_{ij|p}) - \sigma_{rp} \leq 0 \qquad \forall \{r,p\} \in \mathscr{D}, r \neq p \tag{4.9}$$

$$\sum_{\{i,j\}\in\mathscr{N},i\neq j} (c_{ij|p}x_{ij|p} - c_{ij|r}x_{ij|r}) - \sigma_{rp} \leq 0 \qquad \forall \{r,p\} \in \mathscr{D}, r \neq p \tag{4.10}$$

$$\sum_{\{i,j\}\subset S,d\in\mathscr{D}} x_{ij|d} \leq |S| - 1 \qquad S \subset \mathscr{N}, 2 \leq |S| \leq n-2 \tag{4.11}$$

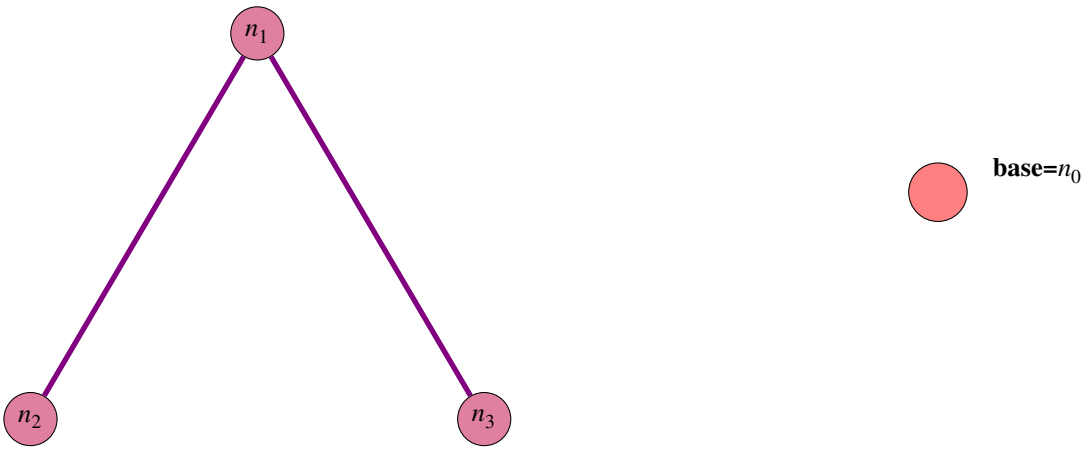


Figura 4.1 Mapa de ejemplificación, con 3 torres y base (n = 4 y m = 2).

Tanto en el caso genérico como en el ejemplo que se está utilizando tenemos los mismos dos datos de partida, el número de nodos y el número de UAVs (que equivale al número de rutas que se debe obtener). Con estos datos ya se puede definir el número de columnas de A_{eq} y A y filas de b_{eq} y b, que es el mismo valor en ambos casos y proviene de la longitud del vector x de la función objetivo 3.1 (los dos sumandos de la función 4.1). El primer sumando corresponde con el planteamiento clásico del TSP que persigue minimizar el tiempo de la suma de todas las rutas (como en la ecuación 2.1), por lo que los valores de $x_{ij|d}$ son los que se van a almacenar en el vector x, mientras que el segundo sumando está pensado para equilibrar el tiempo que requieren las distintas rutas (se explicarán más en detalle en la sección 4.5). Por tanto, son los valores de σ_{rp} los que se van a guardar en x. Así, la longitud total de x será la suma del tamaño de estos dos conjuntos de variables que pueden ser definidos con los datos del número de nodos y UAVs.

La longitud de x_{ij} será el número total de posibles caminos que tiene la misión. Al tener n nodos existen $\binom{n}{2} = \frac{n!}{(n-2)!2!}$ conexiones entre estos, y al considerar los dos posibles sentidos de cada conexión, el número

de caminos por cada UAV es $2\frac{n!}{(n-2)!2!} = n(n-1)$, quedando finalmente $c = mn(n-1)$ caminos para todos los UAVs. Por otro lado, el tamaño de σ_{rp} depende del número de UAVs ya que hay un σ_{rp} por cada pareja de UAVs, quedando $c_{UAV} = \binom{m}{2} = \frac{m!}{(m-2)!2!}$ como la longitud de σ_{rp}. Por tanto, la longitud de x será $mn(n-1) + \frac{m!}{(m-2)!2!}$.

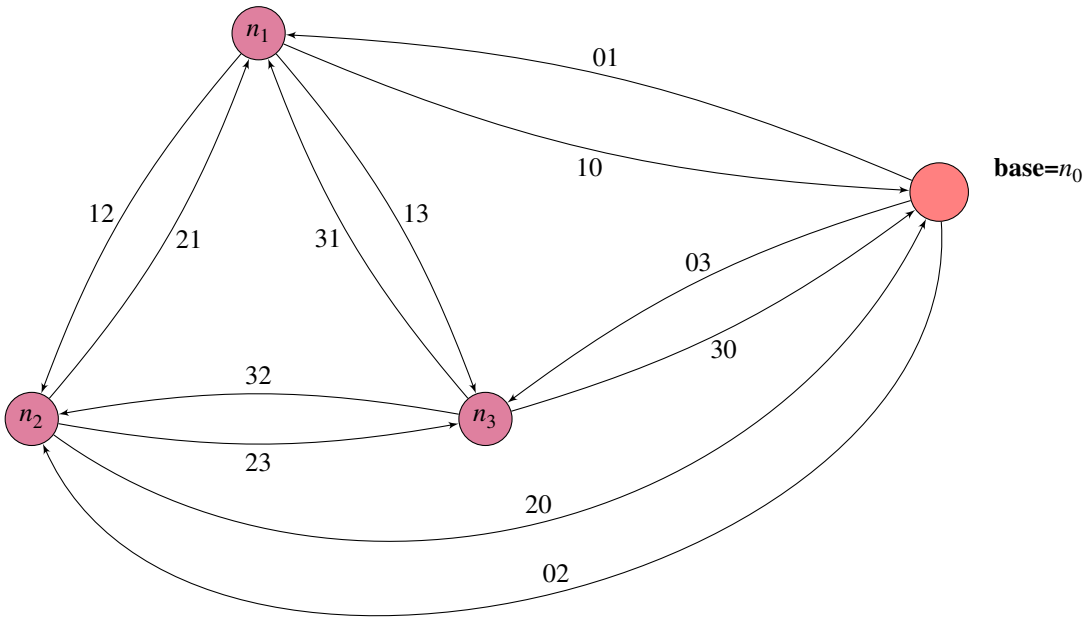


Figura 4.2 Caminos posibles para un solo UAV en el caso de la figura 4.1.

Para el caso de ejemplo se pueden ver todos los caminos para cada UAV en la figura 4.2, dando un total de 12 caminos que puede recorrer cada UAV y 24 caminos entre los 2 UAVs. Además, del segundo sumando quedaría una combinación posible, dando un total de 25 columnas a añadir a las matrices A y A_{eq}.

Aprovechando que los valores que se van a poner en las matrices A y A_{eq} van a seguir un patrón, se va a representar en las siguientes secciones cómo quedarían estas matrices para el ejemplo adoptado mediante dos tablas. Una primera tabla que agrupe las columnas en tres grupos: una por los caminos que le corresponden al UAV 1, otra por los caminos que le corresponden al UAV 2 y una tercera para el valor que deba tener la variable σ_{12}. En esta tabla se indicará que patrón seguirá cada fila de la matriz para cada uno de estos grupos. Por separado, la segunda tabla mostrará como debe de ser cada patrón, indicando el valor que asocia cada patrón a cada camino de un UAV o a σ_{12}.

4.1 Inicio y fin de cada ruta

Tabla 4.1 Estructura de matriz para entrada y salida de la base.

Secciones	caminos UAV1	caminos UAV2	σ
UAV1	F	0	0
UAV2	0	F	0

El primer requisito que se va a explicar es la obligatoriedad de que cada UAV empiece y termine en su propia base, el cual está realmente representado mediante dos restricciones: cada base debe tener un único camino que salga de ella (inicio) y cada base debe tener un único camino que llegue a ella (final). Para cumplirlas se deben satisfacer las ecuaciones 4.2 y 4.3, que representan una igualdad por lo que para adaptarlas a *intlinprog*

se deben plasmar mediante la ecuación 3.4 en las matrices A_{eq} y b_{eq}. Ambas restricciones se plasman en las matrices con la misma estructura para los UAVs, representado en la tabla 4.1, pero seleccionando distintos caminos para cada una de las condiciones, por lo que F puede ser dos vectores diferentes, F_{in} y F_{out}.

4.1.1 Camino inicial

Para que cada UAV inicie su recorrido desde la base hace falta un camino que salga de esta, representado en la restricción 4.3 que fuerza a que solo exista un camino de salida. Para que *intlinprog* pueda mantener esa restricción, la matriz A_{eq} deberá tener m filas, una para cada base (tal y como se muestra en la tabla 4.1), y en cada fila poner a 1 las columnas correspondientes a los caminos que salgan de esa base, mientras que el vector b_{eq} debe ser unitario. De esta forma solo uno de los caminos que sale de la base podría ser válido para la ruta final. Para el caso de ejemplo, la F_{out} de la tabla 4.1 sería la que aparece en la tabla 4.2.

Tabla 4.2 Fila F_{out} que restringe las salidas de las bases.

camino	0-1	0-2	0-3	1-0	1-2	1-3	2-0	2-1	2-3	3-0	3-1	3-2
F_{out}	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0

4.1.2 Camino final

Para cumplir la segunda restricción y conseguir que los UAVs terminen su ruta en la base sin volver a ella en medio del recorrido, se utiliza la restricción 4.2. Al adaptar esta restricción para *intlinprog*, el vector b_{eq} será exactamente igual al caso anterior (el vector unitario), al mismo tiempo que la matriz A_{eq} de este caso tendrá las mismas dimensiones que en el anterior, variando los elementos que valgan 1. Concretamente, en la fila del UAV 1 las columnas que estarán a 1 son aquellas que correspondan con los caminos que llegan a la base del UAV 1. Ejemplificando el caso de ejemplo, la fila F de la tabla 4.1 sería la representada en la tabla 4.3, F_{in}.

Tabla 4.3 Fila F_{in} que restringe las entradas a las bases.

camino	0-1	0-2	0-3	1-0	1-2	1-3	2-0	2-1	2-3	3-0	3-1	3-2
F_{in}	0	0	0	**1**	0	0	**1**	0	0	**1**	0	0

4.2 Entrada y salida de cada nodo

Un requisito básico en cualquier planificador basado en el TSP es que cuando se determina que un UAV entre en un nodo, también se le asocia la salida del mismo y viceversa, ya que si no se le pone esta restricción específicamente el planificador puede aportar soluciones imposibles, como puede ser forzar a un UAV a ir a un nodo del que nunca sale o al contrario, que haga que un UAV salga de un nodo al que nunca ha entrado.

En el caso de las bases, este tipo de problemas está solucionado con la restricción anterior, pero no es aplicable al resto de nodos ya que esas restricciones fuerzan al optimizador a que cada UAV pase por su base una vez. El caso de las torres es distinto, no se quiere forzar a que cada UAV pase por cada torre, si no que se pretende conseguir que si un UAV interacciona con una de las torres, sea con un camino de entrada y otro de salida. Es decir, que cada UAV entre a una torre tantas veces como salga de ella, como indica la restricción 4.4.

Para adaptar esta restricción, se debe añadir a la matriz A_{eq} (n-1)m filas, una por cada torre y cada UAV. En cada una se le dará el valor -1 a los caminos del UAV correspondiente que lleguen al nodo en concreto y el valor 1 para los caminos en los que el UAV salga de ese nodo, mientras que el vector b_{eq} será el vector nulo. De esta manera se consigue que cada vez que un UAV entre en un nodo se sume -1, para que la condición de que la suma sea igual a cero obligue al optimizador a utilizar un camino que salga del nodo (los cuales valdrían 1) y al contrario. Así se consigue que el optimizador utilice el mismo número de entradas y salidas para cada nodo y cada UAV.

En las tablas 4.4 y 4.5 podemos ver cómo se tendrían que construir las matrices A_{eq} para el ejemplo de la figura 5.1.

Tabla 4.4 Estructura matriz para restricción de entrada y salida del mismo nodo.

Secciones	caminos UAV1	caminos UAV2	σ
nodo 1 y UAV1	F^1_{torre}	0	0
nodo 2 y UAV1	F^2_{torre}	0	0
nodo 3 y UAV1	F^3_{torre}	0	0
nodo 1 y UAV2	0	F^1_{torre}	0
nodo 2 y UAV2	0	F^2_{torre}	0
nodo 3 y UAV2	0	F^3_{torre}	0

Tabla 4.5 Filas F_{torre} para restringir las entradas y salidas de cada torre.

camino	0-1	0-2	0-3	1-0	1-2	1-3	2-0	2-1	2-3	3-0	3-1	3-2
F^1_{torre}	0	0	0	0	**1**	**1**	0	**-1**	0	0	**-1**	0
F^2_{torre}	0	0	0	0	**-1**	0	0	**1**	**1**	0	0	**-1**
F^3_{torre}	0	0	0	0	0	**-1**	0	0	**-1**	0	**1**	**1**

4.3 Capacidad de la batería

Otro requisito importante para el planificador es que la capacidad de la batería sea suficiente para que cada UAV pueda realizar la ruta propuesta. Esta restricción viene definida por la ecuación 4.5 que presenta un nuevo parámetro, $E_{ij|d}$, el cual representa el porcentaje de batería consumido por el camino desde i a j para el UAV d. Para calcular el consumo de cada camino debemos hacer estimaciones del consumo de cada UAV que se basan en modelos energéticos dependientes de distintos parámetros del UAV y del efecto del viento.

4.3.1 Modelos energéticos

A la hora de realizar las estimaciones de consumo se han utilizado cuatro modelos que estiman la potencia que requiere un UAV para realizar cada uno de los cuatro movimientos puros (ascender, descender, avanzar y vuelo a punto fijo) especificándole una serie de características propia de cada UAV y otros datos específicos del movimiento que se va a realizar. Gracias al usar estos modelos parametrizados, se puede adaptar el planificador a sistemas multi-UAV heterogéneos y con especificaciones particularizadas para cada misión. La descripción de estos modelos de consumo está fuera del alcance de este trabajo.

Los parámetros propios del UAV son datos que caracterizan la estructura y aerodinámica de la nave, como:

- Masa total
- Número de rotores
- Número de hélices de cada rotor
- Radio del rotor
- Cuerda del rotor
- Coeficiente de sustentación
- Coeficiente de resistencia
- Factor de potencia inducido
- Eficiencia energética

mientras que los datos de los movimientos son datos que se pueden decidir específicamente para cada misión y no dependen de manera estricta del tipo de UAV que se vaya a utilizar. Estos pueden ser:

- **Velocidad de inspección**, velocidad a la que sobrevolará la línea eléctrica.
- **Velocidad de navegación**, velocidad típica a la que se desplazará cuando no está en inspección.
- **Velocidad de ascenso y descenso**, velocidad a la que se eleva y desciende cada UAV.
- **Velocidad respecto al viento**.
- **Capacidad de batería**, que depende del número de celdas y que puede variar por cada UAV.

- **Densidad del aire**, se supone constante para regiones amplias. Afecta al consumo energético.
- **Gravedad**, se toma un valor constante para la Tierra de 9.81 m/s^2.

4.3.2 Efecto del viento

Uno de los elementos que más puede afectar al consumo energético de los UAVs es el viento, ya que los modelos previamente mencionados necesitan la velocidad respecto al viento para estimar el consumo que tienen. Esto se debe a que las aeronaves se mantienen en el aire gracias al efecto de la sustentación, que se basa en usar un perfil alar (ver figura 4.3) para generar una fuerza de sustentación suficiente que contrarreste el efecto de la gravedad al mover el aire. Al final, la velocidad de ese aire va a verse afectada por la velocidad del viento. En consecuencia, la fuerza de sustentación es dependiente de la velocidad del viento.

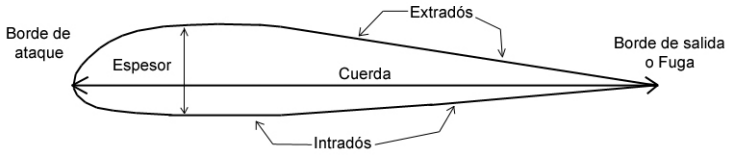


Figura 4.3 Perfil alar.

Los UAVs de ala fija utilizan esta geometría en las propias alas para obtener esa fuerza al avanzar, creando esta sustentación con el aire que circula alrededor de la ala. Mientras que en los multi-rotores, el perfil alar se encuentra en cada una de las hélices de los rotores, de manera que cada rotor genera una fuerza de sustentación, consiguiendo la estabilidad y la maniobrabilidad que caracterizan a los multi-rotores, al tener como mínimo cuatro puntos de sustentación distintos.

Además, en la figura 4.4 tenemos la curva de consumo típico de un UAV, en la que se puede observar que existe una velocidad respecto al aire óptima en la que se consigue una mayor distancia de vuelo. Normalmente interesa trabajar cerca de esta velocidad, aunque no siempre es posible porque el viento no es algo controlable y puede actuar de manera totalmente opuesta a lo que convendría para la misión o porque la velocidad a la que queremos navegar no es esa.

Además, en la figura 4.4 solo se tiene en cuenta el modulo de dicho velocidad, a pesar de que las velocidades son magnitudes vectoriales y por tanto la resta de la velocidad respecto al suelo y respecto al aire da un vector. Por lo cual, se necesita únicamente el módulo de esa velocidad, que se obtiene de la ecuación 4.12, la cual procede de aplicar el teorema del coseno al escenario de la figura 4.5.

$$v = \sqrt{v_s^2 + v_a^2 - 2v_s v_a cos(\alpha)} \tag{4.12}$$

Donde v es la velocidad del UAV respecto al aire, v_s la velocidad el UAV respecto al suelo, v_a la velocidad del viento, y α el ángulo entre ambos.

4.3.3 Obtención de matrices de restricción

Partiendo del consumo que requiera cada camino y de la capacidad de la batería (Q) se obtiene el porcentaje de batería que requiere cada camino ($E_{ij|d}$). Al ser la restricción 4.5 una restricción de desigualdad se debe utilizar la matriz A, añadiéndole m filas (una por cada UAV), y dándole a cada elemento el valor del porcentaje de ese camino ($E_{ij|d}$), consiguiendo que la suma de cada fila dé la cantidad de batería utilizada en la ruta. Para hacer que la restricción se cumpla, el vector b debe ser un vector con todos los elementos a 1. De tal manera que la suma de los porcentajes de batería utilizada en cada ruta pueda ser como mucho una carga completa de la batería. Aunque realmente no se suele utilizar la capacidad real de la batería que se coloca en el UAV, sino que suele usar un 80 % de la batería real, con el fin de corregir pequeños errores en la estimación de los costes o para poder hacer frente antes otras posibles adversidades. Con una matriz con la estructura similar a las de las tablas 4.6 y 4.7, conseguimos que cada ruta nunca pueda consumir más que una batería completa.

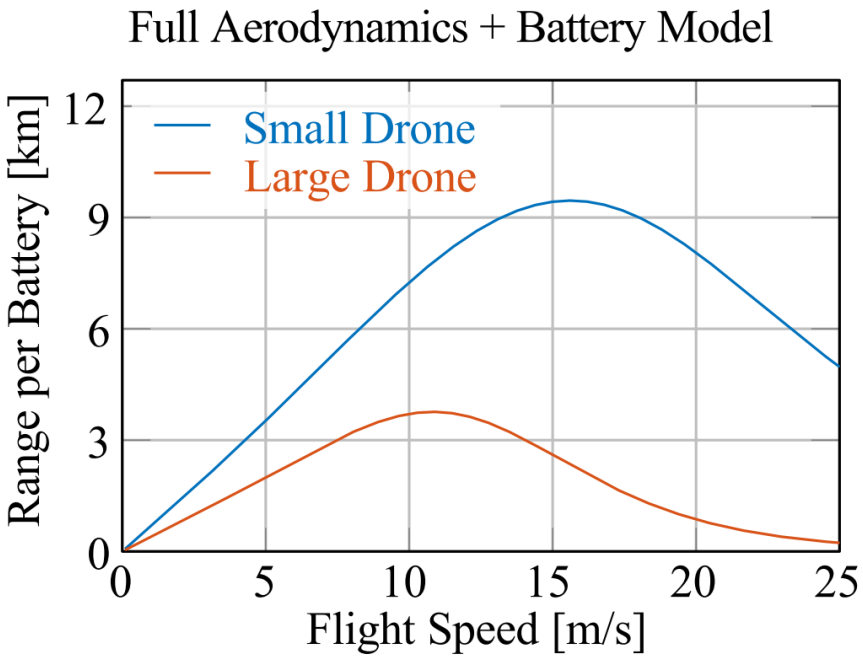


Figura 4.4 Curva de consumo de batería de multi-rotores [13].

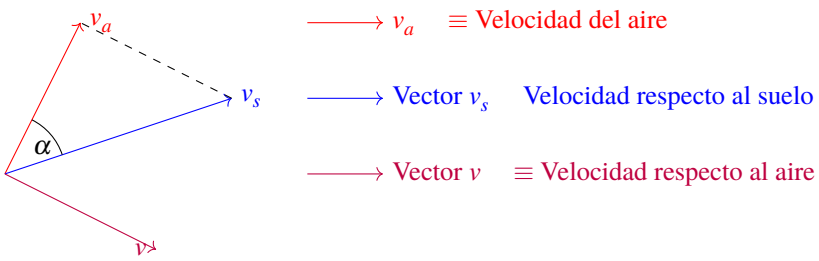


Figura 4.5 Diagrama vectorial de velocidades.

Tabla 4.6 Estructura matriz para restricción de batería.

Secciones	caminos UAV1	caminos UAV2	σ
ruta 1	F_Q^1	0	0
ruta 2	0	F_Q^2	0

Tabla 4.7 Filas F_Q^i para restringir el límite de la batería.

camino	0-1	0-2	0-3	1-0	1-2	1-3	2-0	2-1	2-3	3-0	3-1	3-2
F_Q^1	$E_{01\|1}$	$E_{02\|1}$	$E_{03\|1}$	$E_{10\|1}$	$E_{12\|1}$	$E_{13\|1}$	$E_{20\|1}$	$E_{21\|1}$	$E_{23\|1}$	$E_{30\|1}$	$E_{31\|1}$	$E_{32\|1}$
F_Q^2	$E_{01\|2}$	$E_{02\|2}$	$E_{03\|2}$	$E_{10\|2}$	$E_{12\|2}$	$E_{13\|2}$	$E_{20\|2}$	$E_{21\|2}$	$E_{23\|2}$	$E_{30\|2}$	$E_{31\|2}$	$E_{32\|2}$

4.4 Inspección de las líneas eléctricas

La motivación principal para realizar este planificador es centrarlo en la inspección y mantenimiento de líneas eléctricas, ya que hasta el momento ninguna de las restricciones previamente explicadas asegura que cada línea a inspeccionar sea sobrevolada. Para ello se toman los segmentos de línea de tensión que deben ser inspeccionados como caminos que deben ser recorridos forzosamente durante la misión. Las restricciones que aseguran esto son 4.6, 4.7 y 4.8, debido a que una única restricción no puede asegurar la correcta y rápida inspección de todas las líneas que se desean revisar ya que se debe tener en cuenta varios aspectos.

En primer lugar, el UAV que inspeccione la línea no es importante a priori como se ve en la ecuación 4.6,

por lo cual cada vez que un UAV sobrevuele una de las líneas a inspeccionar, se deberá tener en consideración de manera conjunta en la misma fila de la matriz de restricción a diseñar.

En segundo lugar, tal y como están planteados los caminos existen dos posibilidades por cada UAV de recorrer la misma línea entre el nodo *i* y el *j*: el camino *i* a *j* y el camino *j* a *i*. Los cuales, para esta restricción no se diferencian, por lo que a la matriz de restricción necesaria se le debe añadir el mismo número de filas que de caminos con líneas a inspeccionar. En cada una de estas filas se deberán activar los elementos respectivos al camino a inspeccionar, sin tener en consideración el sentido del recorrido.

Sin embargo, al funcionar la ecuación como 3.3 un límite inferior para el vector *b* no se puede utilizar de la misma manera que se ha utilizado hasta el momento, ya que esta restricción marca un límite superior para el vector *b*. Para utilizarlo como límite inferior, se debe invertir la inecuación de *intlinprog* 3.3:

$$Ax \leq b \tag{4.13}$$

$$-Ax \geq -b \tag{4.14}$$

Aplicado a las matrices, para invertir la restricción basta con cambiar los signos a los valores originales. Esto implica que los elementos a activar en las matrices *A* deben tener el valor -1, mientras que el vector *b* debe pasar de ser un vector unitario positivo a un vector unitario negativo. De esta manera se implementa el límite inferior requerido para esta restricción. En el ejemplo que se está viendo, la matriz *A* resultante quedará como en las tablas 4.8 y 4.9.

Tabla 4.8 Estructura de la matriz para inspección de líneas.

Secciones	caminos UAV1	caminos UAV2	σ
conexión 1	$F^1_{segmento}$	$F^1_{segmento}$	0
conexión 2	$F^2_{segmento}$	$F^2_{segmento}$	0

Tabla 4.9 Filas $F^i_{segmento}$ para inspeccionar las líneas.

camino	0-1	0-2	0-3	1-0	1-2	1-3	2-0	2-1	2-3	3-0	3-1	3-2
$F^1_{segmento}$	0	0	0	0	**-1**	0	0	**-1**	0	0	0	0
$F^2_{segmento}$	0	0	0	0	0	**-1**	0	0	0	0	**-1**	0

Se debe tener en consideración que este límite no restringe el número máximo de veces que se debe pasar por cada uno de los caminos de inspección, lo que puede suponer un perjuicio para el optimizador debido a que *intlinprog* trabaja de forma iterativa y podría suceder que en alguna de las iteraciones pase varias veces por el mismo camino. Esto es algo improductivo, puesto que el pasar varias veces por el mismo camino no aporta nada, únicamente puede aportar para visitar una torre que esté lejos de las demás y solamente está cerca de una torre que del resto, por lo que para llegar a ella el recorrido más corto sea ir y volver por el mismo camino.

Para intentar limitar esto y que *intlinprog* no realice iteraciones en las que no va a conseguir el objetivo, se añade una restricción que limita cada camino en cada dirección un máximo de uno sola vez (4.7 y 4.8), posibilitando recorrer solamente un camino de ida y otro de vuelta. Para construir esta matriz, se aplica la restricción 4.13 para generar el límite superior. En este caso, a la matriz *A* se le deberán añadir 2 veces el número de líneas a inspeccionar, una por cada pareja de filas y cada una de estas filas será para cada una de las direcciones.

De esta manera, para el camino de *i* a *j* habrá dos filas, una para el camino de *i* a *j* y otra para el camino de *j* a *i*. En el primero se activarán todos los elementos del camino de *i* a *j* para todos los UAVs, mientras que en el segundo se activarán los caminos de todos los UAVs que vayan de *j* a *i*. A la vez, el vector *b* será un vector unitario, de tal forma que se fuerza a que cada camino solo pueda ser recorrido una única vez en cada dirección entre todos los UAVs.

4.5 Reducción del tiempo de la misión

Al trabajar con múltiples UAVs, la misión va a tener múltiples rutas que completar y que se pueden ejecutar de manera simultánea, por lo que si queremos minimizar el tiempo que la misión está en curso, debemos

Tabla 4.10 Estructura de la matriz para inspección de líneas.

Secciones	caminos UAV1	caminos UAV2	σ
conexión 1.1	$F^{11}_{segmento}$	$F^{11}_{segmento}$	0
conexión 1.2	$F^{12}_{segmento}$	$F^{12}_{segmento}$	0
conexión 2.1	$F^{21}_{segmento}$	$F^{21}_{segmento}$	0
conexión 2.2	$F^{22}_{segmento}$	$F^{22}_{segmento}$	0

Tabla 4.11 Filas $F^{ij}_{segmento}$ para inspeccionar las líneas.

camino	0-1	0-2	0-3	1-0	1-2	1-3	2-0	2-1	2-3	3-0	3-1	3-2
$F^{11}_{segmento}$	0	0	0	0	**1**	0	0	0	0	0	0	0
$F^{12}_{segmento}$	0	0	0	0	0	0	0	**1**	0	0	0	0
$F^{21}_{segmento}$	0	0	0	0	0	**1**	0	0	0	0	0	0
$F^{22}_{segmento}$	0	0	0	0	0	0	0	0	0	0	**1**	0

minimizar el tiempo de la ruta que requiera más tiempo.

Esta idea se representa en el planificador en el segundo sumando de la función objetivo 4.1 [17], donde σ_{rp} quiere representar la diferencia de tiempo entre cada una de las rutas y $c_{ij|d}$ al tiempo de cada camino. De manera que las rutas se van a diseñar para que sus duraciones sean lo más igualadas posibles, que conjunto al primer término de la función 4.1 (que mantiene la idea del TSP de minimizar todas las rutas), se consigue minimizar la ruta que tarde más.

Para definir σ_{rp} se deben utilizar las ecuaciones 4.9 y 4.10. Esto se debe a que ambas definen a σ_{rp} haciendo de sus límites superior e inferior. Si reescribimos las ecuaciones, despejando σ_{rp}:

$$\sum_{\{i,j\}\in\mathcal{N},i\neq j} (c_{ij|r}x_{ij|r} - c_{ij|p}x_{ij|p}) \geq \sigma_{rp} \tag{4.15}$$

$$\sum_{\{i,j\}\in\mathcal{N},i\neq j} (c_{ij|p}x_{ij|p} - c_{ij|r}x_{ij|r}) \geq \sigma_{rp} \tag{4.16}$$

Ambas ecuaciones cubren los dos casos posibles:

- La ruta del UAV r es más larga que la ruta de p. En este caso, σ_{rp} lo limitará 4.15.
- La ruta del UAV p es más larga que la ruta de r. En este caso, σ_{rp} lo limitará 4.16.

De esta forma, al adaptarlo a *intlinprog* se necesita usar la restricción de desigualdad 3.3 con las matrices A y b. Para cumplir ambas restricciones es necesario añadir a la matriz A 2 filas por cada σ_{rp}, de forma que cada σ_{rp} tenga una fila para la restricción 4.9 y otra para 4.10. En el ejemplo planteado, como solo existe σ_{12}, solo habría que añadir 2 filas a la matriz A, quedando las filas que se ven en las matrices 4.12 y 4.13.

Tabla 4.12 Estructura de la matriz para igualar tiempos de rutas.

Secciones	caminos UAV1	caminos UAV2	σ
σ^1_{12}	F^1_σ	-F^2_σ	1
σ^2_{12}	-F^1_σ	F^2_σ	1

Tabla 4.13 Filas F para igualar tiempos de líneas.

camino	0-1	0-2	0-3	1-0	1-2	1-3	2-0	2-1	2-3	3-0	3-1	3-2
F^1_σ	$c_{01\|1}$	$c_{02\|1}$	$c_{03\|1}$	$c_{10\|1}$	$c_{12\|1}$	$c_{13\|1}$	$c_{20\|1}$	$c_{21\|1}$	$c_{23\|1}$	$c_{30\|1}$	$c_{31\|1}$	$c_{32\|1}$
F^2_σ	$c_{01\|2}$	$c_{02\|2}$	$c_{03\|2}$	$c_{10\|2}$	$c_{12\|2}$	$c_{13\|2}$	$c_{20\|2}$	$c_{21\|2}$	$c_{23\|2}$	$c_{30\|2}$	$c_{31\|2}$	$c_{32\|2}$

4.6 Minimizar las rutas no críticas

A través de los resultados del apartado anterior, se consigue un conjunto de rutas que hará que la misión en conjunto dure lo menos posible. Sin embargo, este enfoque solo asegura que la ruta más larga se recorra en el menor tiempo, siendo esta la ruta crítica de la misión. El resto de rutas obtenidas no tienen que ser

necesariamente las óptimas para los segmentos de líneas que corresponden inspeccionar a cada UAV. Por ello, tras resolver el problema de minimizar la ruta más lenta, se realiza una segunda etapa de optimización de cada una del resto de rutas, las rutas no críticas.

Para esta etapa de optimización se toma una a una cada una de las rutas no críticas y se optimizan siguiendo el planteamiento clásico del TSP, cambiando la restricción de tener que pasar por cada nodo por tener que pasar por todos los segmentos de línea eléctrica a inspeccionar. De este modo se consigue que cada una de las rutas sea óptima para los segmentos que debe inspeccionar. Con esto no se minimiza el tiempo de ejecución de la misión, ya que depende de la ruta crítica, pero sí se mejora el consumo energético de la misma, debido a que se pretende reducir el tiempo de ejecución de la mayoría de las rutas. Además, este problema debería ser resuelto mucho más rápidos porque se aplica a un único UAV y a un número reducido de torres.

4.7 Estaciones de recarga

Un aspecto que se ha intentado tener en cuenta es la posibilidad de recargar la batería durante la misión mediante al utilización de bases de recarga. Esto hubiera permitido realizar ruta más largas ya que la restricción de la batería no hubiese sido un impedimento tan restrictivo como si lo es en el escenario actual. Sin embargo, esta posibilidad presenta muchas dificultades y hace escalar el tiempo de resolución del problema de manera vertiginosa, además de suponer bastantes cambios en la implementación del planificador.

En primer lugar, implica un cambio en el calculo del tiempo, debido a que al tiempo que se tarda de cualquier nodo a una base de recarga, se le debería añadir el tiempo que tarda en recargar y del tiempo de ascenso y despegue para alcanzar al base de recarga, lo que supone añadir un módulo que distinga en cada camino si se llega o se parte de una base de recarga para que se tenga en cuenta estos tiempos. En el mismo sentido, un módulo parecido debería utilizarse para el cálculo del coste, ya que el cálculo del coste de ascender y aterrizar es igual, pero requieren otras funciones de estimación de energía utilizada.

En segundo lugar, esto tiene implicaciones en las restricciones que definen el planificador, reformulando la ecuación 4.5, dejándola como se ve en la ecuación 4.17:

$$\sum_{i \notin \{\mathscr{R},0\}} \; \underset{or}{\sum_{j \notin \{\mathscr{R},0\}}} c_{ij|d} x_{ij|d} \leq \sum_{i \in \{\mathscr{R},0\}} \; \underset{or}{\sum_{j \in \{\mathscr{R},0\}}} x_{ij|d} Q_d / 2 \qquad \forall d \in \mathscr{D} \tag{4.17}$$

Con esta restricción se establece que el coste energético de todos los caminos de un cada ruta debe ser menor que el el consumo de batería (Q) que se va a tener en la ruta. Para contabilizar la cantidad de cargas completa de la batería que se va a usar en la ruta, se toman todos los caminos de llegada o partida a una base de recarga (conjunto $\mathscr{R}$) o a la base de inicio y por cada uno de estos caminos, se suma la mitad de la carga completa de la batería. De esta manera la batería que se tiene al inicio está integrado con la salida y la llegada a la base y cada una de las recargas, se tienen en cuenta con la llegada y la salida (2 caminos) a la base. Como ya se ha explicado previamente, estos cambios suponen un aumento inmenso de las posibilidades para resolver el problema, lo que se manifiesta en un tiempo demasiado elevado para el tipo de aplicaciones a los que está enfocado este proyecto. Por lo tanto, se optó por no introducirlo en la versión final del planificador.

4.8 Matrices resultantes

Una vez definidas todas estas restricciones y sus matrices, para que *intlinprog* pueda procesarlas, se deben unir todas. Para ello, se juntan todas las filas de cada una, separando las matrices A_{eq} y b_{eq} y las matrices A y b. Obteniendo finalmente las matrices de la tabla 4.14 para las restricciones que tengan que cumplir una desigualdad, mientras que están las matrices de la tabla 4.15 para las restricciones que deban cumplir igualdad.

Tabla 4.14 Matrices A y b completas para el caso de ejemplo.

A			b
caminos UAV1	caminos UAV2	σ	
F_Q^1	0	0	1
0	F_Q^2	0	1
0	0	0	1
$F_{segmento}^1$	$F_{segmento}^1$	0	-1
$F_{segmento}^2$	$F_{segmento}^2$	0	-1
$F_{segmento}^{11}$	$F_{segmento}^{11}$	0	1
$F_{segmento}^{12}$	$F_{segmento}^{12}$	0	1
$F_{segmento}^{21}$	$F_{segmento}^{21}$	0	1
$F_{segmento}^{22}$	$F_{segmento}^{22}$	0	1
F_σ^1	$-F_\sigma^2$	1	0
$-F_\sigma^1$	F_σ^2	1	0

Tabla 4.15 Matrices A_eq y b_eq completas para el caso de ejemplo.

A_{eq}			b_{eq}
caminos UAV1	caminos UAV2	σ	
F_{out}	0	0	1
0	F_{out}	0	1
0	0	0	1
F_{in}	0	0	1
0	F_{in}	0	1
0	0	0	1
F_{torre}^1	0	0	0
F_{torre}^2	0	0	0
F_{torre}^3	0	0	0
0	F_{torre}^1	0	0
0	F_{torre}^2	0	0
0	F_{torre}^3	0	0

5 Resultados

En el capítulo anterior se ha visto cómo se forman las matrices necesarias para darle las restricciones al planificador. A lo largo de este capítulo se van a aplicar esas restricciones a un mapa que representa un tendido eléctrico con 11 torres conectadas como se muestra en la figura 5.1, con la finalidad de comprobar el funcionamiento y los resultados que devuelve el planificador.

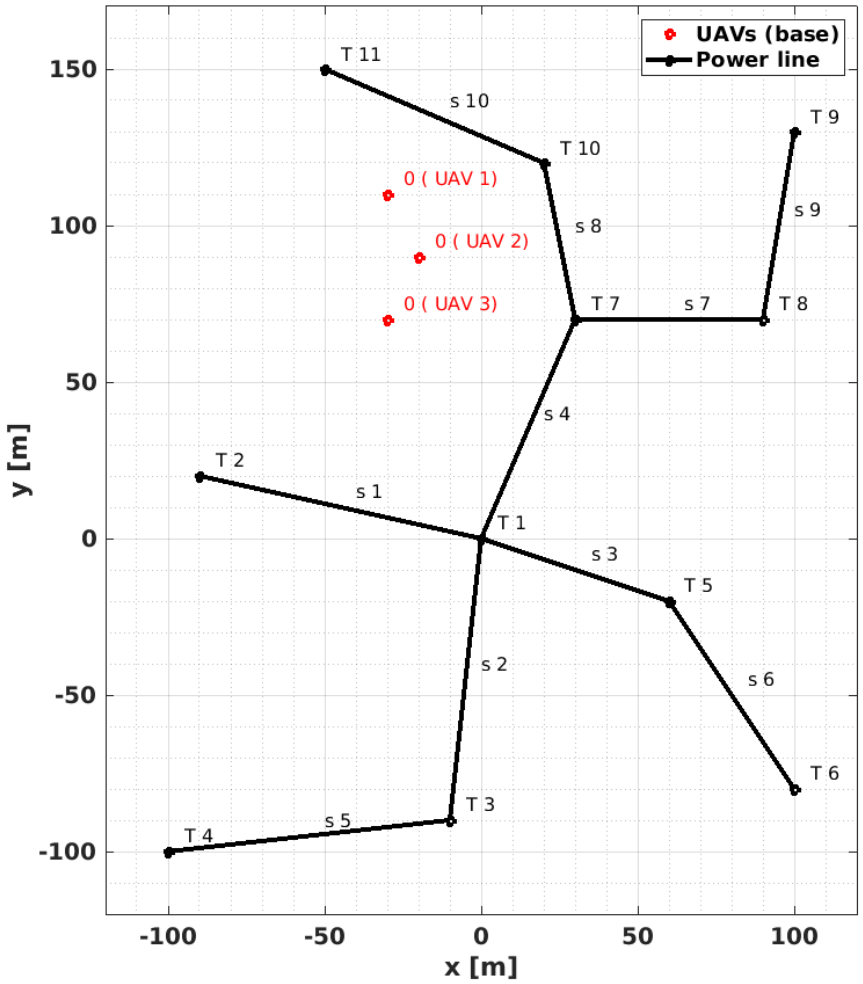


Figura 5.1 Mapa de pruebas.

5.1 Solución principal

Con la finalidad de hacer una comparación justa de los distintos aspectos que se han tenido en cuenta durante el desarrollo del planificador, es necesario tener una solución como punto de partida para la comparación. Para esto se ha determinado que el escenario principal sea el mapa propuesto con viento nulo y con 3 UAVs para la inspección, operando todos ellos a la misma velocidad de inspección. En este caso, el planificador da como resultado las rutas que se muestran en la figura 5.2 y podemos ver algunos datos relevantes en la tabla 5.1.

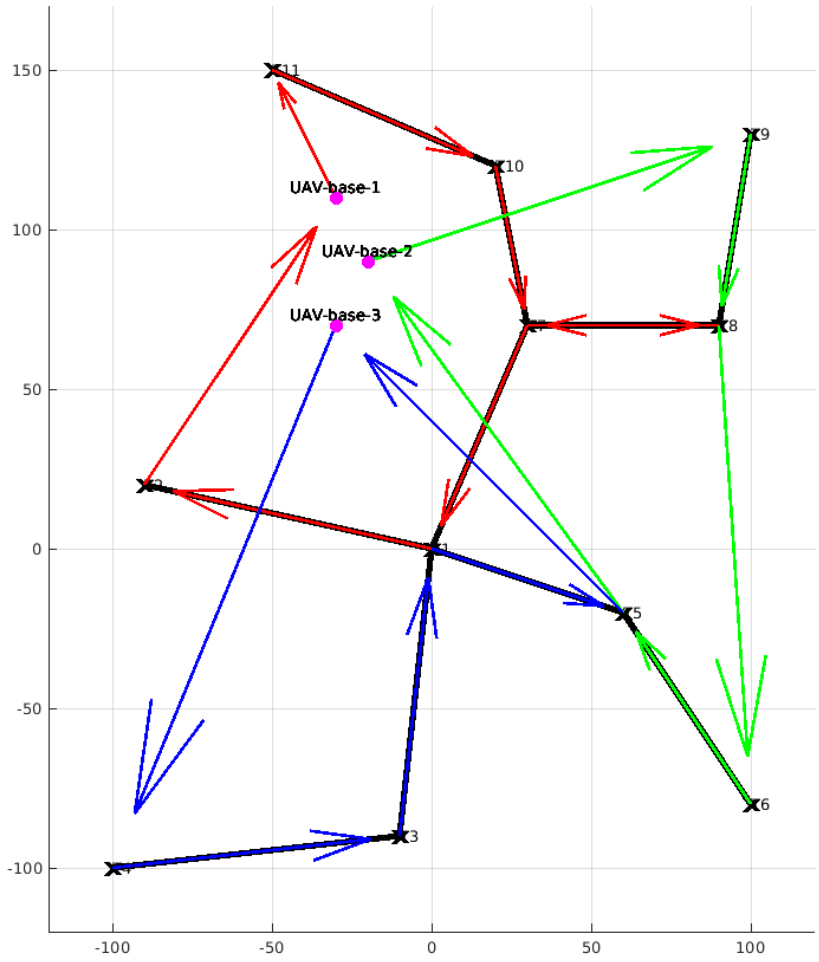


Figura 5.2 Solución principal.

Tabla 5.1 Resultados escenario principal.

	UAV 1	UAV 2	UAV 3
Tiempo	113.68 s	109.16 s	111.10 s
Batería	58.21 %	55.95 %	56.92 %

Tiempo para minimizar ruta crítica	22 min 32.719 s
Tiempo para minimizar rutas no críticas	0.1112 s
Iteraciones *intlinprog*	9

5.2 No optimización rutas no críticas

El primer elemento del que estudiaremos su efecto va a ser la segunda etapa de optimización, la cual se implementó con el objetivo de conseguir optimizar el tiempo de las rutas no críticas e intentar disminuir el consumo energético.

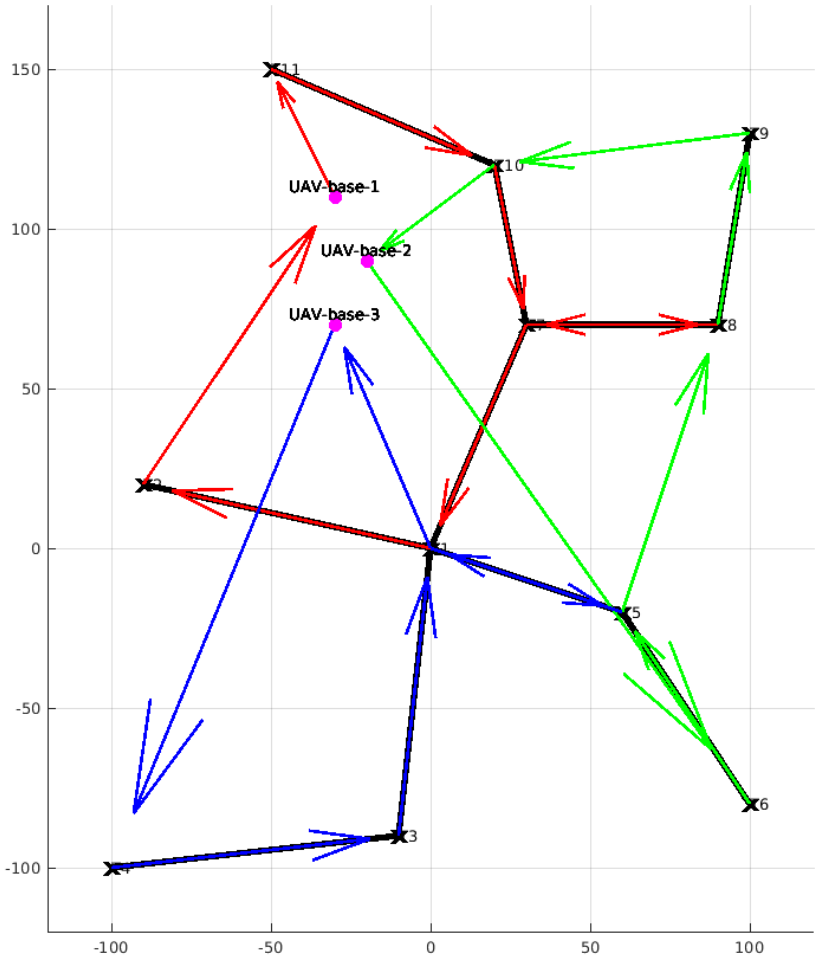


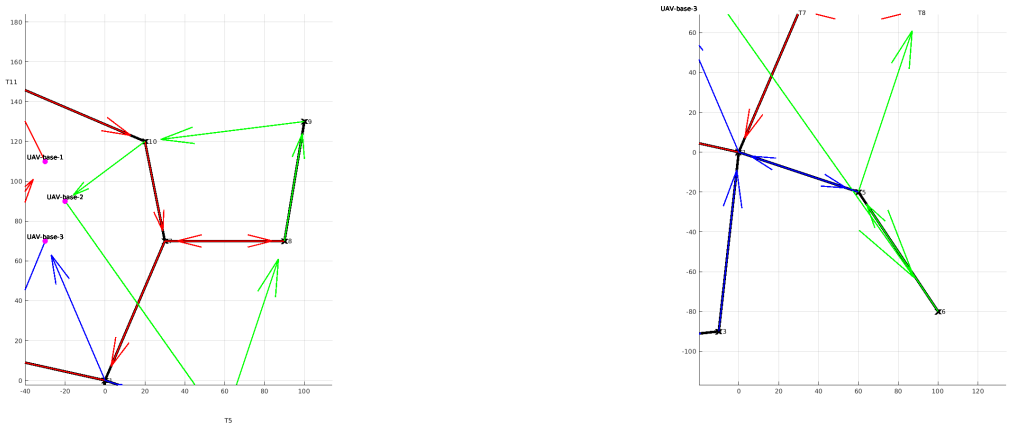
Figura 5.3 Solución sin etapa de optimización de rutas no críticas.

La solución que se obtiene sin esta segunda etapa se puede ver en la figura 5.3, que es bastante parecida a la solución del escenario 1. Sin embargo, hay rutas que toman caminos que se podrían mejorar, como son los casos que se muestran en la figura 5.4. En el caso de la subfigura 5.4a, se ve que al pasar por el nodo 4 no aporta nada respecto a las restricciones que tiene el planificador, al igual que en la subfigura 5.4b, el pasar la segunda vez por el nodo 5 tampoco aporta nada ni es requerido por ninguna de las restricciones.

Tabla 5.2 Resultados sin etapa de optimización de rutas no críticas.

	UAV 1	UAV 2	UAV 3
Tiempo	113.68 s	113.30 s	113.52 s
Batería	58.21 %	58.02 %	58.13 %

Tiempo para minimizar ruta crítica	22 min 50 s
Iteraciones *intlinprog*	9



(a) (b)

Figura 5.4 Recorridos no óptimos.

Además, comparando los datos de las tablas 5.2 y 5.1, se observa que la ruta crítica (la que más tiempo requiere) sigue siendo la ruta del UAV 1, mientras que las otras 2 se observa que minimizan su tiempo de ejecución en la solución principal. Esto se debe a que la segunda etapa de optimización elimina caminos no óptimos como pueden ser los que aparecen en la figura 5.4.

5.3 Efecto del viento

El consumo de batería de los UAVs es muy dependiente de la meteorología, por lo que el viento es uno de los elementos que más podría afectar al consumo energético de los UAVs, como ya se comentó en la sección 4.3.2. Para poder estudiar el efecto del viento en la planificación de la ruta, se tomará el escenario original y se le aplicarán diversos vientos, variando tanto dirección como la velocidad, planteando 4 escenarios:

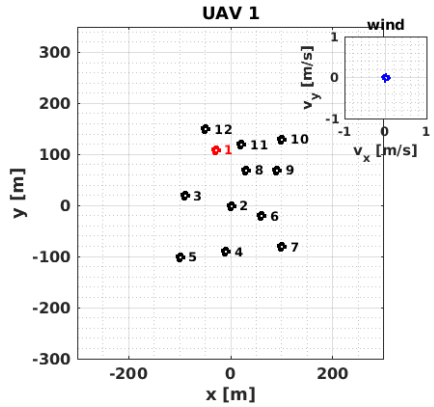
- Viento Norte a $10m/s$.
- Viento Sur a $10m/s$.
- Viento Noroeste a $10m/s$.
- Viento Norte a $11.5m/s$.

Con la finalidad de realizar una comparación se van a utilizar los mapas de consumo, que son unas figuras que representan el porcentaje de batería que consume el UAV en el recorrido desde el nodo i al nodo j. En la figura 5.5b podemos ver el caso del escenario original, con viento nulo, que presenta una simetría debido a este hecho y a que todas las torres están a la misma altura. Así, el coste de ir del nodo i al j debería ser igual que el de ir del nodo j al i. Por otro lado, en la figura 5.5d se puede ver que cada camino consume entre el 10 % y el 40 % de la batería, por lo que no hay ningún camino que el UAV no pueda realizar al menos dos veces.

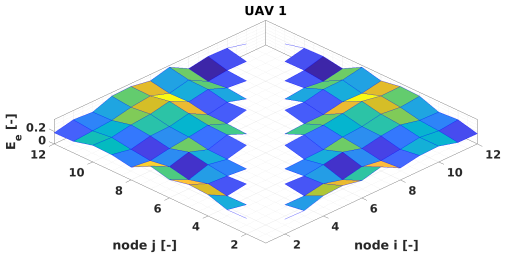
5.3.1 Viento de 10m/s Norte

En este escenario se puede ver el efecto de un viento fuerte con dirección Norte, siendo la velocidad de inspección de todos los UAVs de $5m/s$ y la del viento de $10m/s$. Con esta diferencia se espera ver los efectos que puede causar el viento. Al aplicar este viento, se pueden ver varios efectos en el consumo. En primer lugar, en las figuras 5.6b y 5.6c se puede observar que se pierde la simetría que existía en el caso sin viento. Algo esperable, ya que al existir viento, este podría ayudar en ciertos casos y perjudicar en otros. En segundo lugar, en la figura 5.6d se puede observar como el consumo en general ha subido, haciendo que el peor de los posibles caminos consuma más del doble que en el caso anterior, pudiendo perjudicar a la ruta mediante la restricción de la batería. Si se miran los datos de la tabla 5.3 se confirma esto, ya que los consumos energéticos de los 3 UAVs en este escenario suben respecto a la solución principal.

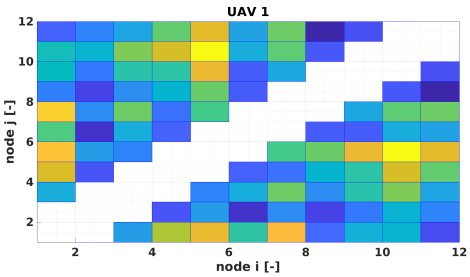
Para este escenario, el planificador nos proporciona las rutas de la figura 5.7, las cuales varían ligeramente con respecto a la solución principal. La ruta crítica permanece igual, mientras que las otras dos varían, ya



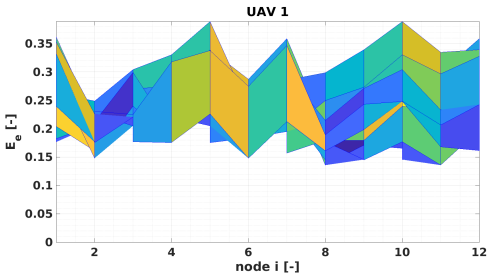
(a) Mapa de torres de la misión original.



(b) Mapa de consumo de misión original sin viento (Perspectiva).



(c) Mapa de consumo de misión original sin viento (Planta).



(d) Mapa de consumo de misión original sin viento (Perfil).

Figura 5.5 Misión sin viento.

Tabla 5.3 Resultados para viento de 10m/s Norte.

	UAV 1	UAV 2	UAV 3
Tiempo	113.68 s	113.04s	88.22 s
Batería	88.79 %	81.74 %	62.73 %

Tiempo para minimizar ruta crítica	12 min 33 s
Iteraciones *intlinprog*	6

que el segmento desde la torre 1 a 5 ha pasado de ser inspeccionado por el UAV 3 a ser inspeccionado por el UAV 2.

5.3.2 Viento de 10 m/s Sur

Este escenario propone un viento con la misma velocidad que el anterior pero en sentido contrario, con la idea de observar como podría afectar la dirección del viento en los resultados del planificador. En las figuras 5.8 se puede ver que los valores del mapa de consumo son parecidos a los que se han obtenido con dirección Norte pero con una simetría respecto al eje donde $i = j$. Esta se puede observar con mayor claridad al comparar la figura 5.8c y la figura 5.6c.

Aparte, viendo las rutas proporcionadas para este escenario en la figura 5.9, se ve que se obtienen las mismas soluciones que con viento Norte y por tanto, en la tabla 5.4 se ve que todas las rutas requieren el mismo tiempo y el tiempo de ejecución del planificador y el porcentaje de batería utilizada son similares.

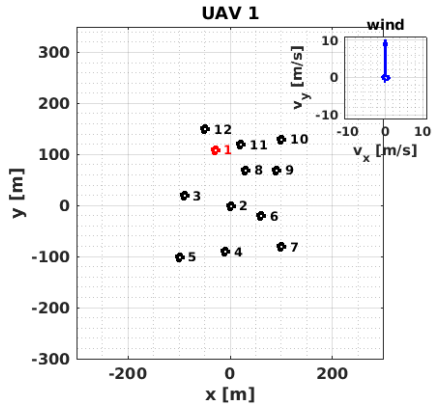
Tabla 5.4 Resultados para viento de 10m/s Sur.

	UAV 1	UAV 2	UAV 3
Tiempo	113.68 s	113.04s	88.22 s
Batería	84.84 %	85.78 %	64.17 %

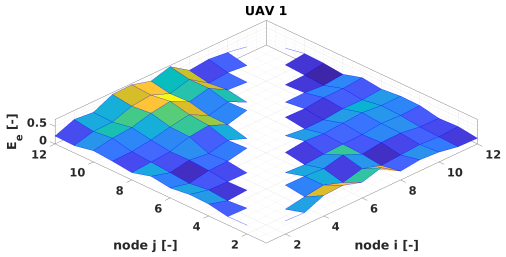
Tiempo para minimizar ruta crítica	18 min 28 s
Tiempo para minimizar ruta no críticas	0.074022 s
Iteraciones *intlinprog*	6

5.3.3 Viento de 10m/s Noroeste

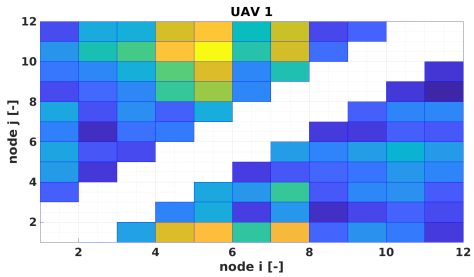
Este escenario se plantea para poder observar como afecta el viento con una dirección distinta, manteniendo la velocidad del viento con el fin de facilitar la comparación.



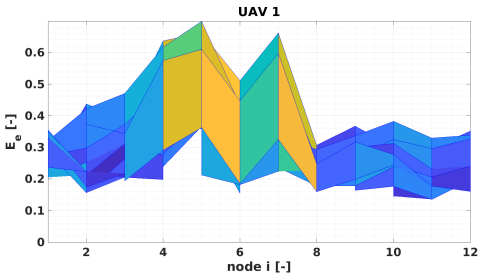
(a) Mapa de torres de la misión original con velocidad del viento.



(b) Mapa de consumo de misión original con viento Norte $10m/s$ (Perspectiva).



(c) Mapa de consumo de misión original con viento Norte $10m/s$ (Planta).



(d) Mapa de consumo de misión original con viento Norte $10m/s$ (Perfil).

Figura 5.6 Misión con viento Norte $10m/s$.

En la figura 5.10c se puede observar que la simetría que existía entre el viento Norte y Sur, aquí no está presente. Algo esperable, ya que esa simetría podría deberse a la simetría de la dirección de los vientos en los dos casos anteriores. Además, en 5.6d se puede observar cómo aumenta el consumo ligeramente, mientras que con viento dirección Sur se mantenía igual que con viento dirección Norte. No obstante, si se comparan los consumos energéticos totales de los casos anteriores (tablas 5.3 y 5.4) con los consumos energéticos de este caso (mostrados en la tabla 5.5), no se aprecia ninguna diferencia importante.

Tabla 5.5 Resultados para viento de 10m/s Noroeste.

	UAV 1	UAV 2	UAV 3
Tiempo	113.68 s	113.04s	88.22 s
Batería	87.53 %	81.32 %	60.0194 %

Tiempo para minimizar ruta crítica	11 min 7 s
Tiempo para minimizar ruta no críticas	0.07000 s
Iteraciones *intlinprog*	6

5.3.4 Viento de 11.5 m/s Norte

Con este caso se pretende forzar las baterías al caso más extremo que puedan soportar. En la tabla 5.6 se observa que las baterías quedarían casi agotadas tras estas rutas, llegando el UAV 3 a consumir el 99.30 % de su capacidad. Esto es consistente con lo que se puede ver en la figura 5.12d, ya que hay algunos caminos que llegan a consumir el 80 % de la batería. Al no poder hacer tantas combinaciones de caminos por el consumo energético, las rutas que proporciona el planificador en la figura 5.13 no se asemeja a los resultados anteriores, que si tenían cierta similitud entre ellas.

Tabla 5.6 Resultados para viento de 11.5m/s Norte.

	UAV 1	UAV 2	UAV 3
Tiempo	107.65 s	109.16s	113.80 s
Batería	94.57 %	90.38 %	99.30 %

Tiempo para minimizar ruta crítica	28 min 48 s
Tiempo para minimizar ruta no críticas	0.085594 s
Iteraciones *intlinprog*	11

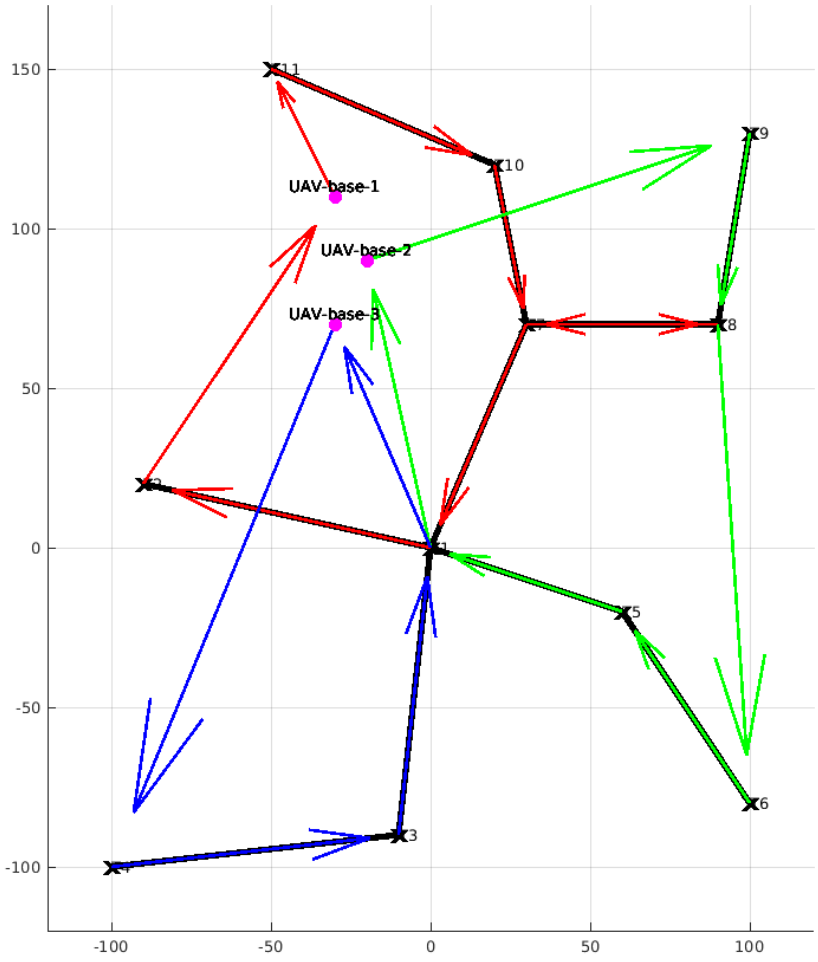


Figura 5.7 Rutas para viento de 10m/s Norte.

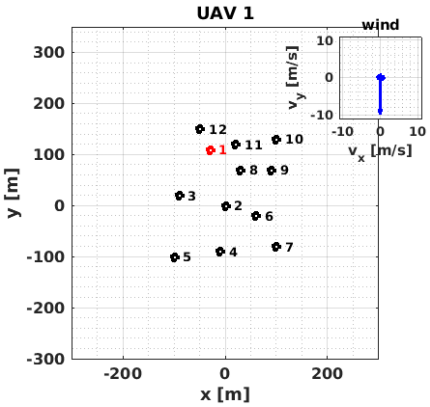
5.4 Efecto del número de UAVs

Otro de los aspectos fundamentales de que se han tenido en cuenta para este planificador es la posibilidad de poder cambiar el número de UAVs que van a estar disponibles en la misión. En la solución principal, se toman 3 UAVs y para realizar una comparación se va a utilizar el mismo escenario sin viento, pero con 2 UAVs. En la tabla 5.7 se observa que el tiempo necesario para resolver la planificación ha disminuido drásticamente de aproximadamente 15 minutos a los 30 segundos al utilizar un UAV menos. A la vez, cada UAV tarda más que en los otros casos, gastando también más batería. Lo cual era esperable, porque ahora cada UAV va a tener que inspeccionar más segmentos que al utilizar 3 UAVs, como se ve en la figura 5.14.

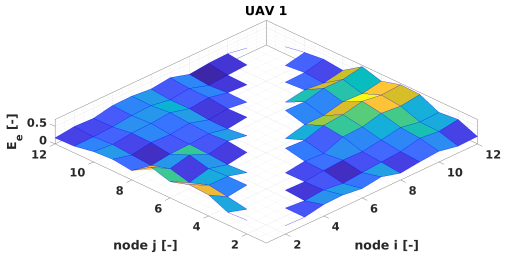
Tabla 5.7 Resultados para misión con 2 UAVs.

	UAV 1	UAV 2
Tiempo	150.44 s	150.41 s
Batería	76.55 %	76.53 %

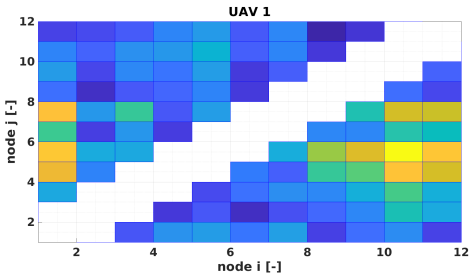
Número de caminos posibles	220
Tiempo para minimizar ruta crítica	32.471 s
Tiempo para minimizar rutas no críticas	0.026781 s
Iteraciones *intlinprog*	9



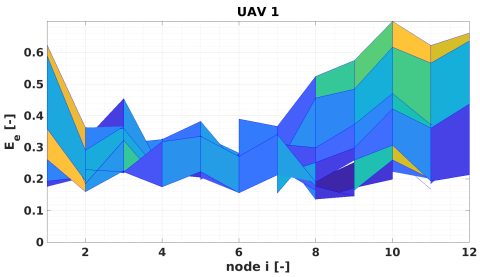
(a) Mapa de torres de la misión original con velocidad del viento.



(b) Mapa de consumo de misión original con viento Sur $10m/s$ (Perspectiva).



(c) Mapa de consumo de misión original con viento Sur $10m/s$ (Planta).



(d) Mapa de consumo de misión original con viento Sur $10m/s$ (Perfil).

Figura 5.8 Misión con viento Sur $10m/s$.

5.5 Validación en entorno real

En esta sección se pretende explicar el proceso completo que implicaría la generación de las rutas para las misiones y utilizar el planificador en un entorno real. Para ello se necesitan distintos recursos que se irán adaptando y consiguiendo para las necesidades del proyecto.

5.5.1 ATLAS

Lo primero que se necesita es un lugar donde se puedan realizar las pruebas, para lo que vamos a utilizar los mapas de las líneas eléctricas de ATLAS, donde ya se han realizado pruebas con anterioridad en los proyectos AERIAL-CORE y RESISTO. ATLAS [2] es un centro de pruebas de vuelo especializado para pruebas con UAVs que cuenta con grandes espacios para la realización de pruebas aéreas, tanto pistas como una gran zona de espacio aéreo segregado para realizar distintas pruebas en condiciones seguras, como se puede ver en la figura 5.15. Por tanto, es un buen entorno en el que realizar experimentos para la validación del planificador.

Para estas pruebas se va seleccionar un tendido eléctrico dentro del espacio aéreo del ATLAS y cerca de su pista de aterrizaje, como se puede ver en la figura 5.16.

5.5.2 Preparación del mapa

Una vez seleccionado los segmentos de tendido eléctrico a revisar, es necesario obtener las coordenadas de los mismos. Para esto, la herramienta más utilizada es Google Earth Pro, a través de la cual se puede seleccionar sobre vista de satélite los puntos donde están las torres y las bases de los UAVs que se van a utilizar, para extraerlo posteriormente en un fichero *.kml legible, de donde se obtienen las coordenadas de estos puntos.

Estas coordenadas no están preparadas para su uso en el planificador debido a dos motivos:

- **La altura** no se puede extraer, ya que Google Earth Pro no facilita este dato al guardar las ubicaciones. Por tanto, no se podría saber cómo varían las alturas, lo que afectaría al cálculo de la distancia entre

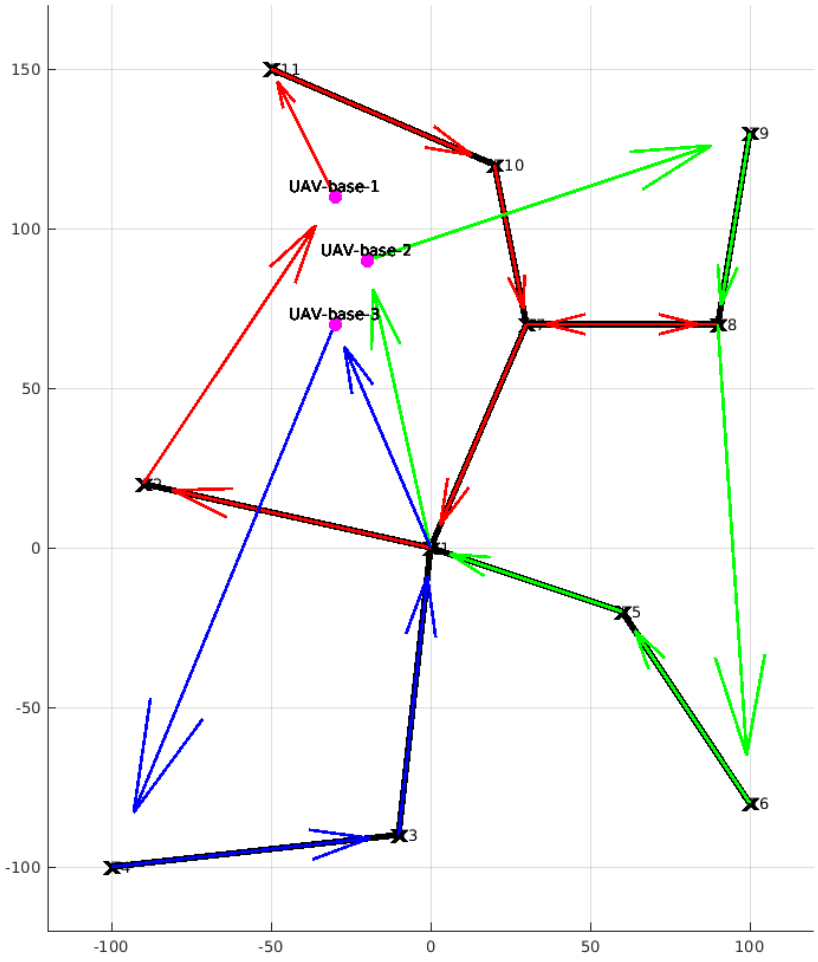


Figura 5.9 Rutas para viento de 10m/s Sur.

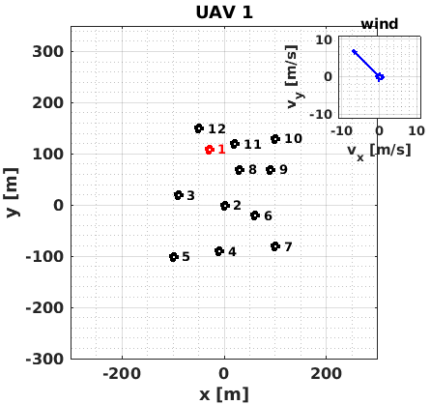
dos torres, además de al cálculo de tiempo de camino y de la energía a utilizar. Para solucionarlo, se requiere utilizar la API explicada en la sección 3.4 para obtener la altura de los puntos respecto al nivel del mar.

- **Las coordenadas** extraídas de Google Earth Pro a través de ficheros *.kml vienen dadas en el sistema de coordenadas geográficas, pensado para representar la forma geodésica de la Tierra. Sin embargo, bajo este sistema de coordenadas no se puede aplicar el planificador, ya que este trabaja con distancias euclídeas. Por lo tanto, hace falta una transformación a coordenadas UTM, para lo que es necesario saber en la región en la que se encuentra ATLAS. Al encontrarse en Villacarrillo (Jaén), se puede ver que se encuentra en la zona 30S en el mapa 3.3.

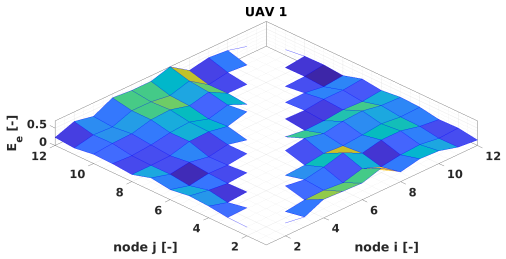
Una vez adaptadas las coordenadas, ya se pueden dar al planificador para que realice la planificación.

5.5.3 Resultados

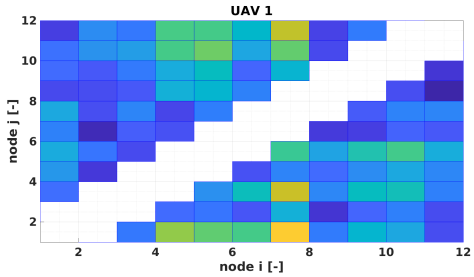
Las rutas proporcionadas se pueden ver en la figura 5.17 y, con los datos de la tabla 5.8, vemos que ante un caso real y con menos ramas que en el caso de ejemplo, el planificador da soluciones con mucha mayor rapidez, ya que ha tardado menos tiempo y ha requerido una única iteración, a pesar de tener más torres.



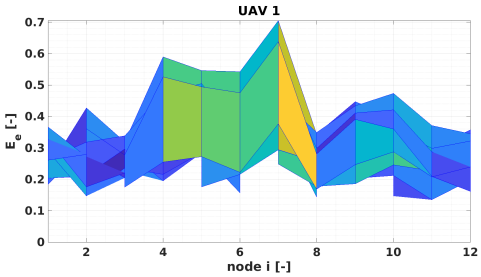
(a) Mapa de torres de la misión original con velocidad del viento.



(b) Mapa de consumo de misión original con viento Noroeste $10m/s$ (Perspectiva).



(c) Mapa de consumo de misión original con viento Noroeste $10m/s$ (Planta).



(d) Mapa de consumo de misión original con viento Noroeste $10m/s$ (Perfil).

Figura 5.10 Misión con viento Noroeste $10m/s$.

Tabla 5.8 Resultados para validación en entorno real.

	UAV 1	UAV 2
Tiempo	261.0258 s	261.0283 s
Batería	47.74 %	44.11 %

Tiempo para minimizar ruta crítica	4.081s
Tiempo para minimizar rutas no críticas	0.03297 s
Iteraciones *intlinprog*	1

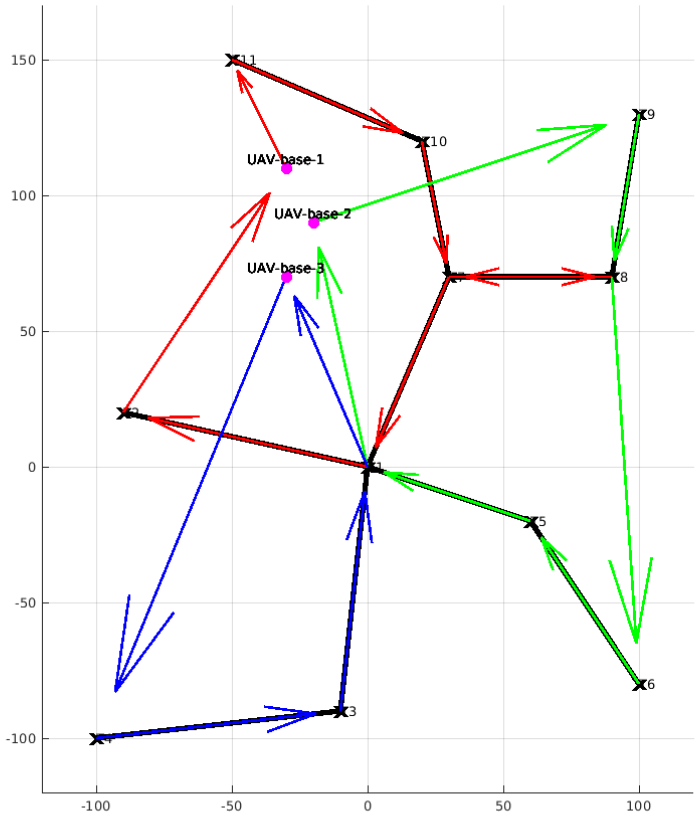
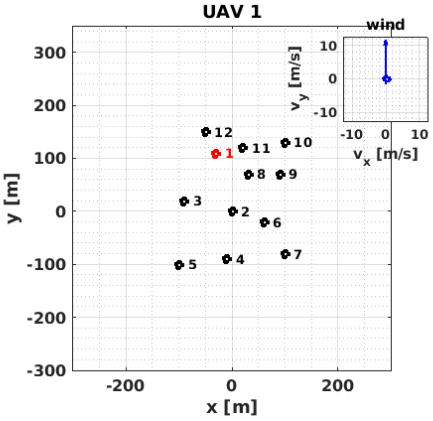
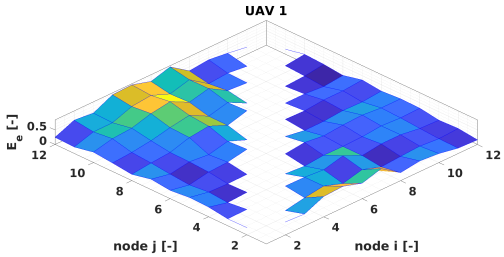


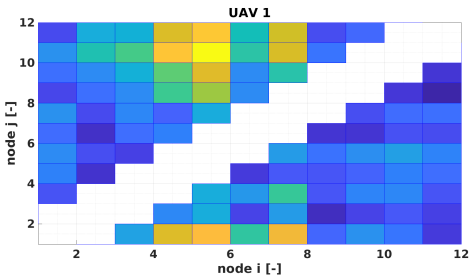
Figura 5.11 Rutas para viento de 10m/s Noroeste.



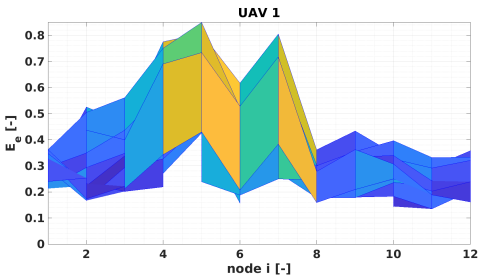
(a) Mapa de torres de la misión original con velocidad del viento.



(b) Mapa de consumo de misión original con viento Noroeste $11.5 m/s$ (Perspectiva).



(c) Mapa de consumo de misión original con viento Noroeste $11.5 m/s$ (Planta).



(d) Mapa de consumo de misión original con viento Noroeste $5 m/s$ (Perfil).

Figura 5.12 Misión con viento Noroeste $11.5 m/s$.

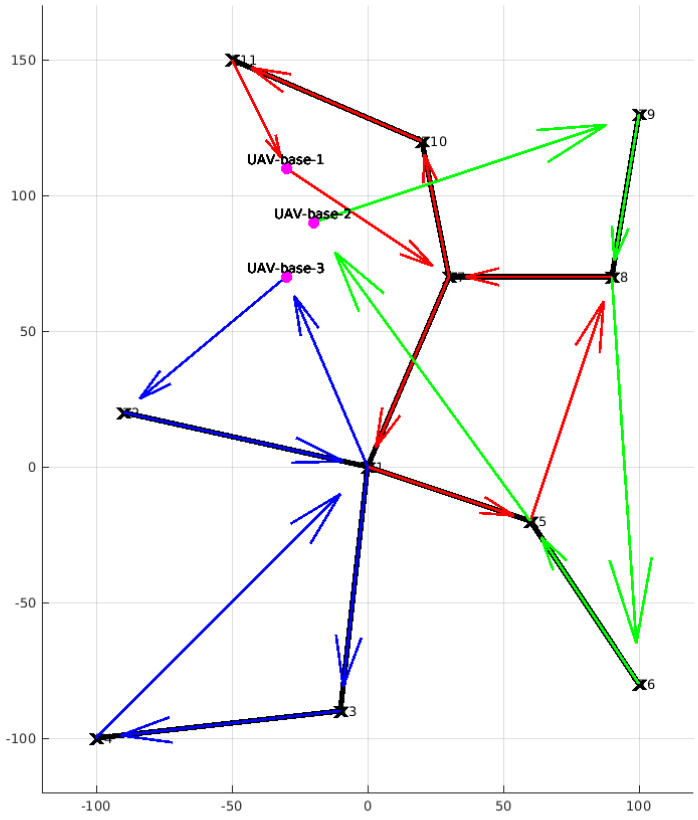


Figura 5.13 Rutas para viento de 11.5m/s Norte.

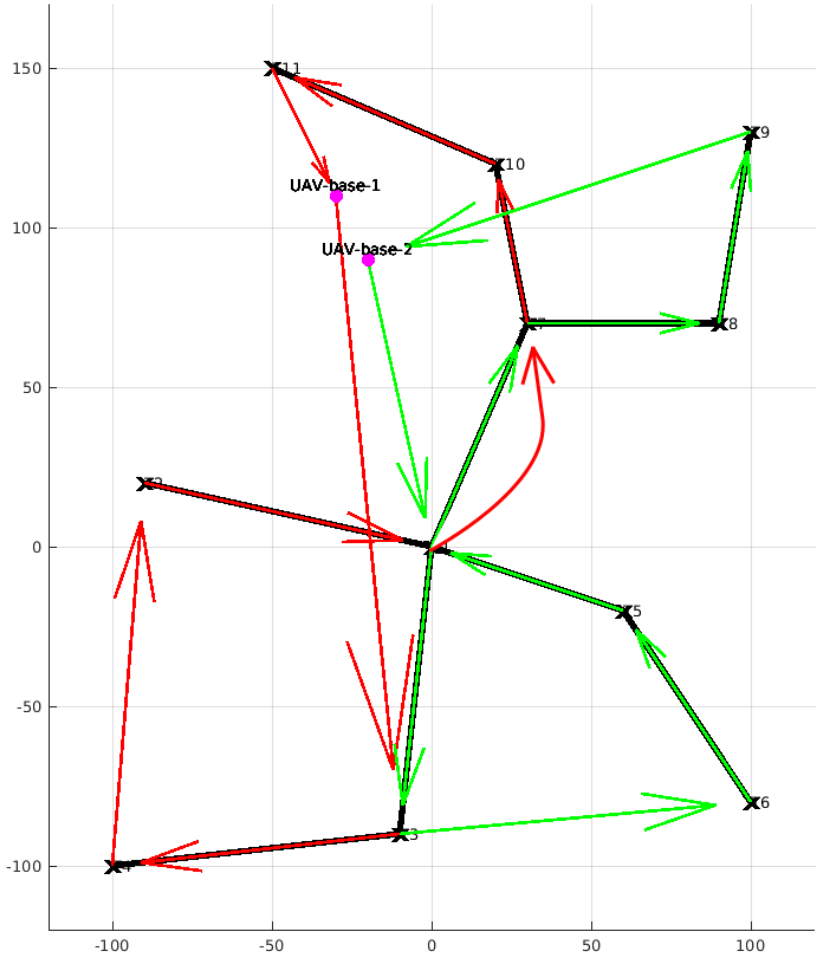


Figura 5.14 Rutas para 2 UAVs.

Figura 5.15 Zona aérea especial de ATLAS.

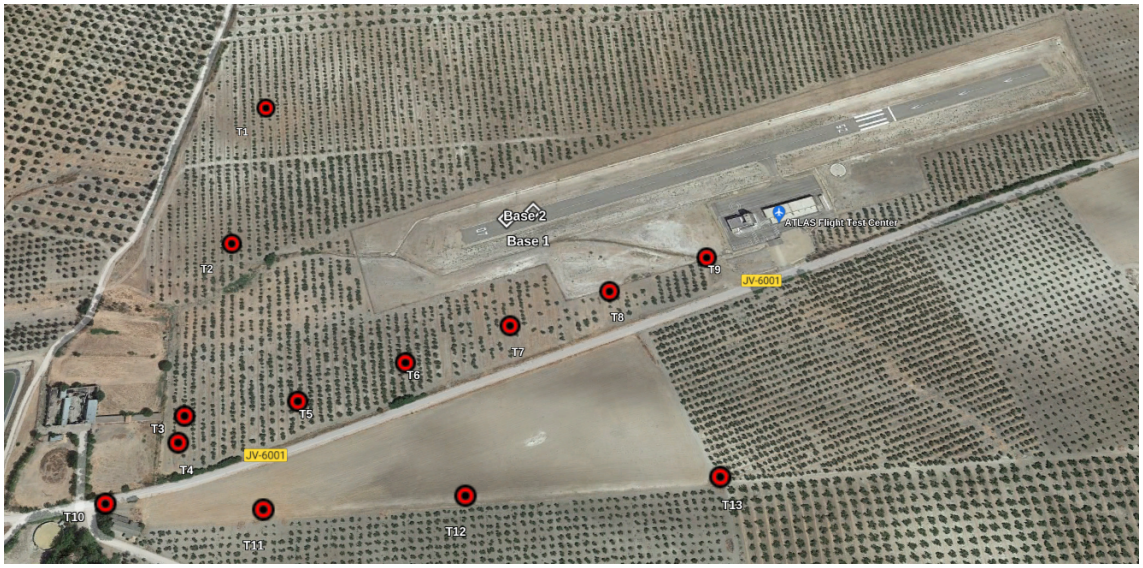


Figura 5.16 Mapa de tendido eléctrico en ATLAS (torres de la línea).

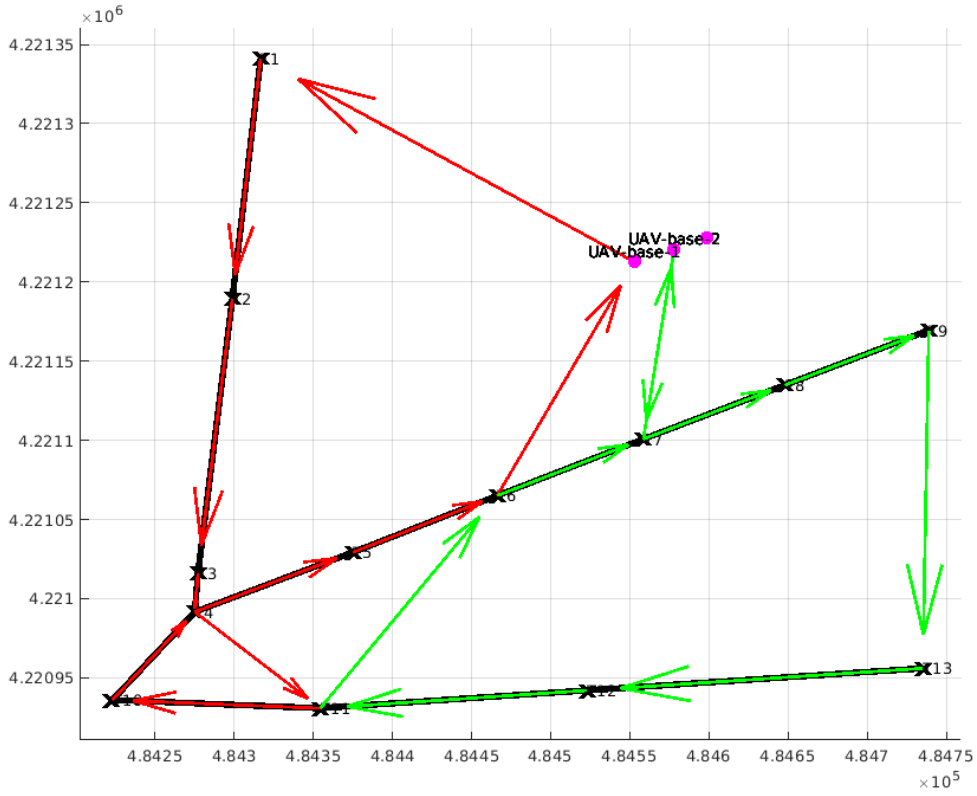


Figura 5.17 Rutas para misión en ATLAS.

6 Conclusiones y Trabajo Futuro

En el capítulo anterior se vieron y comentaron diferentes resultados del planificador ante un mapa preparado para probar cada una de las partes que tiene el planificador y cómo estas pueden llegar a afectar al propio desempeño. Además, se realiza una planificación real para la inspección de una parte de la línea eléctrica cerca al centro de vuelo ATLAS. Durante este capítulo se pretende profundizar más en qué indican esos resultados y plantear posibles mejoras partiendo de esas conclusiones.

En primer lugar, tenemos los resultados del planificador sin la segunda etapa de optimización. Teniendo en cuenta los datos de las tablas 5.1 y 5.2 se observa que para los UAVs 2 y 3 se reduce el tiempo de las rutas en algunos segundos y el consumo de batería baja también. Ante esto, la segunda etapa de optimización ha demostrado ser beneficiosa ya que reduce los tiempo y el consumo energético a un coste de cómputo de milésimas de segundo.

En segundo lugar, se ha querido tener en consideración el efecto del viento, analizando varios casos de vientos fuertes. Se han probado tanto viento en diferentes direcciones como con diferentes velocidades y se ha visto que el viento es un elemento importante y que se debe tener en cuenta. Puesto que en los experimentos del capítulo anterior se ha visto cómo el consumo de la batería pasaba de ser en torno al 40 % cuando no había viento a cercanos al 70 % con un viento de $10 m/s$ y al aumentar el viento ligeramente a 11.5 m/s, se llegaba a consumir el 99 % de la batería. Además, este incremento de batería se ha producido a pesar de haber buscado siempre el mejor recorrido posible, por lo que las rutas que se podrían hacer sin viento, pueden llegar a no ser la mejor ruta posible, e incluso llegar a ser inviables a partir de que exista un determinado viento. Esto le ha sucedido a la solución que proponía el planificador para la solución principal, que al aplicar el viento ya no era la mejor opción y el planificador proporcionaba otra acorde al viento.

En tercer lugar se ha probado a inspeccionar el mismo mapa con 2 UAVs en vez de 3, por lo que la misión pasa de tardar cerca de los 113 s a 150 s. No aumenta ni un minuto de misión, aunque el tiempo de cómputo sí varía significativamente. En los experimentos con 3 UAVs, el planificador siempre había necesitado entorno a los 20 minutos para resolverlo, pero al utilizar solo 2 UAVs, el planificador ha tardado cerca de 30s. Por tanto, se consigue una reducción de tiempo muy importante, que puede deberse a que, al encontrarnos ante una variante del TSP, es un problema NP-hard, y al eliminar un UAV se reduce en gran medida el tamaño de las matrices para el optimizador, así como el número de posibles combinaciones. El número de UAVs afecta al tamaño de estas matrices mediante dos vías:

1. El número de caminos posibles es directamente proporcionar al número de UAVs, como se demostró en el capítulo 3.
2. La cantidad de rutas a equilibrar con las restricciones 4.9 y 4.10.

Otro parámetro que afecta fuertemente al tiempo de computo requerido por el planificador para resolver el problema, es añadir el uso de estaciones de recarga, como ya se comentó anteriormente. Esto no solo añade más columnas en las matrices de restricciones, sino que también añade nuevas restricciones y una gran aumento de los posibles caminos. Si se utilizasen estas estaciones, la ruta no estaría limitada a una única batería, si no que se podrían utilizar tantas baterías como visitas se hagan a una estación de recarga. Esto aumenta tanto las posibles rutas, que conlleva un aumento inconmensurable del tiempo de computo, por lo que se decidió no integrarlo en el planificador final.

Con el fin se seguir mejorando el planificador, en futuros proyectos se debería seguir investigando en formas de agilizar la optimización, como puede ser la agrupación de torres, integrando varias en un único

nodo, para reducir el tamaño de las matrices que debe usar el optimizador, siguiendo la heurística de "Cluster First, Route Second" [21], o mediante nuevas restricciones que limiten las posibilidades que deba eliminar el optimizador.

Índice de Figuras

Índice de Tablas

Bibliografía

[1] *About the project - PILOTING H2020 Project — piloting-project.eu*, https://piloting-project.eu/about-the-project/, [Accessed 06-Jul-2023].

[2] *ATLAS | Air Trafic Laboratory for Advanced unmanned Systems — atlascenter.aero*, https://www.atlascenter.aero/en/, [Accessed 04-Jul-2023].

[3] *Copernicus Land Monitoring Service - EU-DEM — eea.europa.eu*, https://www.eea.europa.eu/data-and-maps/data/copernicus-land-monitoring-service-eu-dem, [Accessed 30-Jun-2023].

[4] *EHang | UAM - Passenger Transportation and Logistics — ehang.com*, https://www.ehang.com/uam/, [Accessed 29-Jun-2023].

[5] *GRVC | Robotics, Vision and Control Research Laboratory — grvc.us.es*, https://grvc.us.es/, [Accessed 05-Jul-2023].

[6] *HYFLIERS Project — oulu.fi*, https://www.oulu.fi/hyfliers/#overview, [Accessed 06-Jul-2023].

[7] *Objectives | AEROARMS — aeroarms-project.eu*, https://aeroarms-project.eu/objectives/, [Accessed 06-Jul-2023].

[8] *Open Topo Data — opentopodata.org*, https://www.opentopodata.org/#public-api, [Accessed 06-Jul-2023].

[9] *Proyecto RESISTO — edistribucion.com*, https://www.edistribucion.com/es/innovacion-nuevas-tecnologias/Proyecto_RESISTO.html, [Accessed 03-Jul-2023].

[10] Aerial-Core, *Objectives*, https://aerial-core.eu/objectives/, [Accessed 30-May-2023].

[11] Ed Alvarado, *237 Ways Drone Applications Revolutionize Business — droneii.com*, https://droneii.com/237-ways-drone-applications-revolutionize-business, [Accessed 29-Jun-2023].

[12] David L. Applegate, Robert E. Bixby, Vašek Chvátal, and William J. Cook, *The traveling salesman problem*, Princeton University Press, Princeton, 2007.

[13] Leonard Bauersfeld and Davide Scaramuzza, *Range, endurance, and optimal speed estimates for multicopters*, IEEE Robot. Autom. Lett. **7** (2022), no. 2, 2953–2960.

[14] DJI, *DJI - Official Website — dji.com*, https://www.dji.com/es, [Accessed 29-Jun-2023].

[15] US Air Force, *MQ-9 Reaper — af.mil*, https://www.af.mil/About-Us/Fact-Sheets/Display/Article/104470/mq-9-reaper/, [Accessed 29-Jun-2023].

[16] Tuned Global, *3 ways drones are used to enhance music eventséxperiences — blog.tunedglobal.com*, https://blog.tunedglobal.com/3-ways-drones-are-used-to-enhance-music-events-experiences, [Accessed 29-Jun-2023].

[17] Álvaro Caballero Gómez, *Motion planning with dynamics and environment awareness for aerial robotic manipulation in inspection and maintenance*, Ph.D. thesis, Universidad de Sevilla, 2022.

[18] GRVC, *GRIFFIN | GRVC — grvc.us.es*, *https://grvc.us.es/newweb/griffin/*, [Accessed 29-Jun-2023].

[19] Kiev Kelvin, *The Evolution of Drone Use in the Film Industry [2023] - Kiev Kelvin — kievkelvin.com*, *https://kievkelvin.com/blog/drone-use-in-the-film-industry/*, [Accessed 29-Jun-2023].

[20] Gilbert Laporte, *What you should know about the vehicle routing problem*, Nav. Res. Logist. **54** (2007), no. 8, 811–819 (en).

[21] Frantisek Nekovar, Jan Faigl, and Martin Saska, *Multi-tour set traveling salesman problem in planning power transmission line inspection*, IEEE Robot. Autom. Lett. **6** (2021), no. 4, 6196–6203.

[22] OpenWeatherMap.org, *Weather API - OpenWeatherMap — openweathermap.org*, *https://openweathermap.org/api*, [Accessed 06-Jul-2023].

[23] Amazon Staff, *Amazon reveals the new design for Prime Air's delivery drone—here's your first look — aboutamazon.com*, *https://www.aboutamazon.com/news/transportation/amazon-prime-air-delivery-drone-reveal-photos*, [Accessed 29-Jun-2023].

[24] Alejandro Suarez, Rafael Salmoral, Ambar Garofano-Soldado, Guillermo Heredia, and Anibal Ollero, *Aerial device delivery for power line inspection and maintenance*, 2022 International Conference on Unmanned Aircraft Systems (ICUAS), 2022, pp. 30–38.

[25] Mugin UAV, *Mugin-5 Pro 5000mm VTOL UAV Platform with 8 Motor Mounts x2013; Mugin UAV — muginuav.com*, *https://www.muginuav.com/product/mugin-5-pro-5000mm-vtol-uav-platform-8-motor-mounts/*, [Accessed 29-Jun-2023].