Recibido: 4/1/2023 / Aceptado: 13/3/2023
doi: https://doi.org/10.26439/interfases2023.n017.6230

Optimización de planes de vuelo para múltiples drones en zonas de construcción

Alvaro Sotelo Vila
alvaro_10_12@hotmail.com
https://orcid.org/0000-0001-6154-6144
Universidad de Lima, Perú

Lourdes Ramírez Cerna
lramirec@ulima.edu.pe
https://orcid.org/0000-0002-7927-7875
Universidad de Lima, Perú

RESUMEN

El sector de la construcción ha encontrado en los drones una tecnología útil para la vigilancia y supervisión de obras, en especial desde la pandemia del COVID-19. Esta investigación propone el diseño de modelos de planificación de vuelo con el fin de optimizar su tiempo y velocidad. El objetivo es desarrollar un modelo que permita emplear múltiples drones para llevar a cabo tareas de supervisión en zonas de construcción. En este sentido, se presenta un modelo de programación dinámica y una metaheurística basada en algoritmo genético, ambos aplicados para la optimización de planes de vuelo con múltiples drones. Las propuestas implementadas en Python han sido probadas en 14 escenarios, incrementando gradualmente la complejidad. En todos ellos, el modelo basado en programación dinámica muestra mejoras significativas en el tiempo de planificación, obteniendo una diferencia promedio de 281,34 segundos o 4 minutos y 47 segundos, lo cual es un 98,01 % superior al algoritmo genético. Además, se observa una mejora considerable en las velocidades por segmento, lo cual se refleja en los resultados.

PALABRAS CLAVE: programación dinámica, planificador de vuelo, drones de vigilancia, algoritmo genético

FLIGHT PLAN OPTIMIZATION FOR MULTIPLE DRONES IN CONSTRUCTION SITES

ABSTRACT

The construction sector is searching for new technologies, such as drones, that prove helpful for the surveillance and supervision of construction sites, especially in pandemic situations. This research proposes designing flight planning models to optimize flight time and speed. The main objective is to develop a model that allows multiple drones to carry out supervision tasks in construction areas. In this regard, it presents a dynamic programming model and a metaheuristic based on genetic algorithms, both applied for optimizing flight plans with multiple drones. The development process involves planning the route that the drones will follow by formulating the problem with the corresponding parameters. The next step is to generate a model for the dynamic programming algorithm, which is then validated using a genetic algorithm. The proposals implemented in Python were tested in 14 scenarios, gradually increasing in complexity. The dynamic programming-based model significantly improves planning time in all scenarios, achieving an average difference of 281,34 seconds or 4 minutes and 47 seconds, 98,01 % better than the genetic algorithm. Additionally, there is a considerable improvement in segment speeds, as the results show. A paired test evaluated these advancements. The hypothesis is supported with a p-value of 0,0031 for time and 0,0071 for the gain obtained by the objective function in both cases. This confirms the superiority of the dynamic programming algorithm compared to the genetic algorithm.

KEYWORDS: dynamic programming, flight planner, surveillance drones, genetic algorithm

1. INTRODUCCIÓN

El sector de la construcción en el Perú es uno de los principales generadores de empleo anualmente, junto con la manufactura y la minería. Esto se debe a la gran inversión en infraestructura pública y bienes raíces privados. Así, en el 2014, este sector empleó a 916 000 trabajadores (Palomino et al., 2017). Por ello, es necesario supervisar constantemente las obras para garantizar el cumplimiento de las medidas de seguridad establecidas en el Decreto Supremo 011-2019-TR, publicado en el diario oficial *El Peruano* (2019), como el uso de cascos, lentes de seguridad, chalecos de alta visibilidad y zapatos de seguridad.

Una tendencia emergente para satisfacer esta necesidad es el uso de drones, cuya popularidad sigue en aumento. Albeaino y Gheisari (2021) la atribuyen a la profesionalización del conocimiento técnico requerido, a la implementación de regulaciones más estrictas que brindan confiabilidad y al incremento de políticas de seguridad. Estas últimas continúan su expansión debido a la creciente demanda de obras impulsada por la migración hacia la capital.

Los drones actualmente son utilizados por empresas de construcción en las labores de mapeos topográficos y el estudio de suelos, seguimiento de equipos, monitoreo del progreso de obra, vigilancia y seguridad, seguridad del personal e inspección de infraestructura, y fotografía (Balfour Beatty, 2017; Sando, 2021). De entre las mencionadas, el mapeo topográfico es una de las tareas más precisas para generar reportes de suelos y, por tanto, requiere de ayuda de programas y herramientas automatizadas, entre las cuales destacan los drones.

La pandemia ha sido otro impulsor del uso de drones en el sector construcción y sus planificadores, puesto que en ese momento se incrementó la cantidad de implementos necesarios para los obreros, lo cual demandaba una supervisión constante. Igualmente, era necesario mantener la medida de distanciamiento social de al menos 1,5 metros entre personas. Esto se debía a que, de acuerdo con las directrices del Ministerio de Vivienda, Construcción y Saneamiento (MVCS, 2020), si un trabajador se contagia o presenta síntomas compatibles con el COVID-19, debe ser enviado a casa o acudir a un centro de salud. La normativa tuvo un impacto negativo en la industria y generó la necesidad de buscar nuevas tecnologías emergentes para mitigar los costos adicionales, pues fue necesario contratar más personal para la supervisión de las obras.

Por otro lado, la comunidad científica muestra un creciente interés académico en la creación de planificadores de vuelo. Un ejemplo destacado es el estudio realizado por Yi y Sutrisna (2021), en el que se desarrolla un modelo dinámico para abordar el entorno en constante cambio de las zonas de construcción. Además, se plantea como trabajo futuro la creación de un modelo que permita la supervisión simultánea de múltiples drones. Asimismo, el artículo de Li et al. (2019) expone una serie de *frameworks* de agrupamiento jerárquico y propone como trabajo futuro la aplicación de este enfoque para el pilotaje de

drones en edificios o lugares más estrechos. En este sector, se utilizan también nuevas tecnologías, como los drones de vigilancia, para llevar a cabo proyectos de gran envergadura. De acuerdo con Fan y Saadeghvaziri (2019), se espera que en la próxima década este mercado genere una rentabilidad de 45 000 millones de dólares.

Por lo tanto, resulta imperativo desarrollar un *software* de planificación de vuelo que pueda abordar estos problemas. Aunque existen numerosos algoritmos y planificadores de rutas para diversos ámbitos, se requiere uno capaz de monitorear múltiples drones y abarcar un terreno más extenso. Por esta razón, esta investigación tiene como objetivo abordar la brecha existente en cuanto a un planificador de vuelo para múltiples drones en el sector de la construcción.

Este trabajo cobra relevancia en medio de los avances en tecnologías que permiten viajes complejos para vehículos como los UAV (vehículos aéreos no tripulados, por sus siglas en inglés). El Departamento de Ingeniería Aeroespacial de la Universidad de Maryland destaca el rápido progreso de tecnologías como la microelectrónica y la creciente demanda por desarrollar pilotos automáticos más pequeños y ligeros para permitir la operación de vehículos aéreos con mayor dinámica, tal como mencionan Hrishikeshavan y Chopra (2017).

Según Criado y Rodríguez Rubio (2015), también existe un impacto positivo en industrias y personas interesadas en el entretenimiento con drones. Además, el uso de drones fomenta un impacto ambiental positivo al descongestionar el tráfico en las ciudades y reemplazar métodos de entrega más tradicionales, lo cual reduce las emisiones de gases de efecto invernadero y mejora la eficiencia del transporte (Doole et al., 2020).

En este trabajo se presenta un modelo de programación dinámica y un algoritmo genético diseñados para su aplicación en zonas de construcción, con el fin de realizar funciones de supervisión. Estos algoritmos buscan optimizar las labores de supervisión en función del tiempo y la velocidad que cada dron requiere en diferentes partes del trayecto. Además, permiten la coordinación de varios drones simultáneamente y consideran otras restricciones necesarias para el correcto funcionamiento del vuelo. Como resultado de esta optimización del vuelo, se logra reducir costos como el consumo de combustible, el tiempo de vuelo y, en última instancia, los costos operativos.

El presente documento se estructura de la siguiente manera: la sección 1 es la introducción; la sección 2 abarca el estado del arte y las investigaciones previas relacionadas; la sección 3 describe la metodología utilizada; la sección 4 presenta los resultados obtenidos; la sección 5 discute los resultados previos; y, finalmente, la sección 6 presenta las conclusiones del estudio.

2. ESTADO DEL ARTE

En el estado del arte, se llevó a cabo una revisión exhaustiva de la literatura relacionada con los algoritmos de planificación de vuelo. En esta categoría, se encontraron numerosos estudios que emplean diversos tipos de algoritmos, los cuales fueron clasificados según la tipología propuesta por Hromkovič (2013) en algoritmos deterministas, heurísticos y de aproximación.

2.1 Deterministas

2.1.1 Branch and bound

En el artículo de Poikonen et al. (2019), se aborda un problema similar al planteado por Schermer et al. (2020), conocido como el problema del viajante de comercio con drones (TSP-D, por sus siglas en inglés), en el cual una serie de drones se encargan de distribuir pedidos desde un camión. Para resolver este problema, se propone un modelo basado en el método de *branch and bound*.

En este caso, se establecen rutas predefinidas hacia los puntos de entrega, similares a un grafo. El dron se dirige a los puntos marcados a partir de una raíz que contiene los dos puntos más cercanos al dron. Luego, se crean ramificaciones donde se copian los valores de la instancia anterior y se agrega el nuevo punto de entrega. El objetivo final es lograr que el dron visite todos los puntos de entrega minimizando el tiempo que requiere para ello. Con este fin, se selecciona la ramificación que cumpla con otras condiciones, como la capacidad de la batería del dron.

Comparando el TSP con el TSP-D, se obtiene que, para todos los escenarios, excepto cuando el número de instancias es 12, el valor objetivo se reduce en al menos un 30 %, al igual que el tiempo promedio de finalización, que se reduce en un 40 %. Un posible trabajo futuro consiste en mejorar el modelo al agregar múltiples puntos de inicio y fin, lo que implica el uso de múltiples camiones y clientes, a diferencia del enfoque presentado en este artículo, donde se establece una zona de partida y llegada para cada dron.

2.1.2 Programación lineal

El artículo de Wankmüller et al. (2020) presenta un enfoque para calcular las variables de vuelo necesarias para el despliegue de drones con desfibriladores en operaciones de rescate en los Alpes. El estudio propone un modelo de optimización lineal que determina qué dron es el más adecuado para responder a una solicitud de ayuda, basándose en la proximidad geográfica. Para ello, se consideran las velocidades de vuelo del dron en cada eje espacial, junto con la distancia, para calcular el tiempo estimado de llegada al destino, que es la función que se busca minimizar.

Además, se establecen restricciones como la capacidad de atender solo una solicitud a la vez y un número limitado de locales con una cantidad limitada de drones disponibles. Utilizando una configuración base de 36 drones, se lograron tiempos de viaje de 05:27 minutos en promedio a los sitios de los pacientes. Esto permitió que el 50 % reciba un desfibrilador externo automatizado (DEA) dentro de los 05:00 minutos, lo que se traduce en una tasa de supervivencia estimada que oscila entre el 50 % y el 70 % si se proporciona resucitación cardiopulmonar (RCP) de inmediato.

Por otro lado, en el artículo de Schermer et al. (2020) se propone un sistema que utiliza drones y camiones para resolver una variante del problema del viajante de comercio (*traveling salesman problem* [TSP]), llamada problema de ubicación de estaciones para el viajante con drones (*traveling salesman drone station location problem* [TSDSLP]). Este sistema combina un camión y un dron para realizar entregas.

Para resolver este problema, se emplea un modelo de programación lineal que determina la mejor ruta para el camión, que se desplaza por la ciudad hacia las estaciones de abastecimiento, mientras el dron realiza las entregas a los clientes. Se asume que los clientes son capaces de recibir los paquetes entregados por el dron. La función objetivo del modelo es minimizar el tiempo total del proceso de distribución y los costos asociados, teniendo en cuenta variables como la operatividad de las estaciones y la velocidad del camión, entre otros.

Los resultados muestran que, a medida que aumenta el número de estaciones, los costos disminuyen significativamente, con ahorros superiores al 50 % al usar tres estaciones en comparación con el 30 % obtenido con una sola estación. Además, el alcance de los drones en relación con el camión influye considerablemente en los costos. Al usar un alcance de 8 unidades, el ahorro se reduce a menos del 20 % al utilizar tres estaciones. Sin embargo, uno de los principales problemas señalados por los autores es que los solucionadores estándar solo pueden resolver instancias pequeñas del TSDSLP, lo que impide abordar escenarios más grandes, como un sitio de construcción o múltiples drones trabajando simultáneamente, a diferencia de lo presentado en este trabajo.

En el artículo de Nguyen et al. (2022), se plantea un problema similar basado en el TSP llamado problema de enrutamiento de vehículos paralelo con programación de drones de costo mínimo (*parallel drone scheduling vehicle routing problem* [PDSVRP]). En este sistema, que implica la interacción de camiones y drones para las entregas, los drones compiten con los camiones para realizar las entregas.

Para resolver este problema, se utiliza un modelo de programación entera con el objetivo de minimizar el costo total de entrega de los pedidos, específicamente el costo operativo. Se emplea el método *slack induction by string and sweep removals* (SISSRs) y se aplican restricciones adicionales relacionadas con la capacidad de los drones y camiones, y el nivel de combustible, entre otras. Se realizaron 10 iteraciones y se

mejoraron 7 soluciones encontradas en la literatura con la prueba de ejecución única, obteniendo un total de 26 nuevas soluciones.

En particular, se compararon los resultados con otros dos artículos y sus respectivos escenarios, y se observó una diferencia de hasta el 1,2 % en los valores objetivo. Sin embargo, también se hallaron escenarios en los que se consiguió la misma solución óptima, sin ninguna diferencia en los valores objetivo (brecha de 0 %).

2.1.3 Programación dinámica

El artículo de Yi y Sutrisna (2021) propone un modelo de programación de drones para la vigilancia efectiva de sitios de construcción. El enfoque desarrollado se basa en la programación dinámica, lo que permite resolver el problema de manera eficiente. Para aplicar este enfoque, se discretizaron los valores de la energía de la batería, el tiempo de vuelo y la velocidad de vuelo. Se planteó un caso de estudio utilizando las zonas de construcción y mantenimiento de la Universidad Massey para demostrar la aplicabilidad del modelo y algoritmo propuestos. Un trabajo futuro sugerido por los autores es adaptar este modelo para el uso de múltiples drones simultáneamente.

Para comparar los resultados de los diferentes casos, se utilizó una heurística y un modelo de programación dinámica que exploró todas las posibles combinaciones de velocidades, considerando condiciones como la velocidad máxima. Los resultados demostraron claramente la superioridad del método de programación dinámica propuesto, con una mejora máxima del 430 % en comparación con el algoritmo de programación heurística. En general, se observó una mejora promedio del 293 % en todos los escenarios. Sin embargo, como se mencionó anteriormente, este modelo no es adecuado para abordar el uso de varios drones, lo cual limita sus aplicaciones en términos de distancias y tiempos frente al modelo presentado en este trabajo.

Por su parte, el artículo de Bouman et al. (2018) aborda otro problema de TSP-D utilizando un modelo de programación dinámica. En este caso, se divide el problema en partes, donde la ruta del dron se considera el problema principal y la ruta del camión que transporta al dron se considera un subproblema. Las distancias y los puntos de entrega ya están predefinidos.

Para el problema de los drones, se utiliza un algoritmo que busca los valores óptimos del dron al calcular variables como el tiempo, la batería y la velocidad. El segundo problema consiste en hacer que el camión intercepte la ruta del dron, que ya fue definida en el subproblema anterior. Se debe considerar el tiempo de entrega y minimizarlo, con la restricción de seguir la ruta del dron. Se observa que, cuando solo hay 10 nodos para recorrer, la programación dinámica obtiene mejores resultados que la programación lineal frente a otro artículo utilizado para la comparación.

Sin embargo, a medida que aumenta el número de nodos, el tiempo de ejecución crece exponencialmente y muestra una diferencia de hasta 47 minutos en comparación con el artículo de referencia en el escenario sin camiones y 20 nodos, e incluso supera las 12 horas de procesamiento en el caso de 4 camiones y 20 nodos. En este artículo, también se plantea como trabajo futuro la inclusión de múltiples drones en el modelo, ya que debido a que comienzan y terminan en el mismo punto, esto debería ser factible, un aspecto que se aborda en el presente estudio.

2.2 Metaheurísticas

El artículo de Khan et al. (2021) se centra en la mejora del transporte médico a través de drones. Su objetivo es utilizar drones médicos para transportar equipo de primeros auxilios, evitando así los problemas de tráfico que pueden retrasar la atención médica. Para lograr esto, propone cuatro algoritmos diferentes: el problema de enrutamiento de vehículos capacitado (CVRP, por sus siglas en inglés), la optimización de enjambre de partículas (PSO, por sus siglas en inglés), la optimización de colonias de hormigas (ACO, por sus siglas en inglés) y el algoritmo genético (GA, por sus siglas en inglés).

El modelo matemático que los algoritmos utilizan para calcular la ruta óptima tenía como función objetivo minimizar el costo de movilizar el dron de un paciente a otro, es decir, la distancia. Además, se plantearon cuatro premisas para generar las restricciones:

1. Formular un conjunto de rutas con costos mínimos.
2. Cada dron tendrá asignada una ruta.
3. Cada paciente será visitado una sola vez.
4. Cada ruta comienza en el nodo 0 y termina en el nodo $n + 1$.

Se realizaron pruebas comparativas entre los distintos algoritmos, ingresando el modelo y los parámetros correspondientes. En términos de tiempo de planificación, se encontró que tanto el CVRP como el GA obtuvieron los mejores resultados. Específicamente, el CVRP demostró una mejora en el tiempo de ejecución de hasta 0,05 segundos en comparación con el GA en todos los escenarios analizados. Además, se logró obtener los mismos costos en todos los casos. Vale mencionar que la capacidad del vehículo se considera una restricción importante, ya que al reducirse a la mitad (de $q = 20$ a $q = 10$), se observó una disminución del 50 % en el tiempo de ejecución.

Por otro lado, el artículo de Fu et al. (2012) aborda un propósito diferente, enfocándose en el planificador de rutas de drones para evitar obstáculos, particularmente radares terrestres. En este caso, se utiliza un método basado en algoritmos genéticos y se emplea una función que maximiza la distancia con los obstáculos sin alejarse demasiado de los *waypoints*.

El algoritmo propuesto consta de tres etapas principales. En la primera, se calcula la distancia con los obstáculos; en la segunda, se calcula la distancia entre el *waypoint* previo y el posterior; y en la tercera, se divide la primera ecuación entre la segunda y se multiplica por un factor de proporción para obtener la función objetivo. Además, se proponen ciertos parámetros iniciales, como un número de generaciones de 1000 y un porcentaje de mutación del 5 %.

En los experimentos realizados, se observó que el número de generaciones necesario para obtener un resultado satisfactorio fue de 15 por obstáculo, lo que demuestra la eficacia del algoritmo para evitar obstáculos de manera efectiva.

2.3 Algoritmos de aproximación

En el trabajo de Lee y Cha (2020), se presenta un enfoque para optimizar la ruta en vuelos autónomos de drones de vigilancia utilizando el aprendizaje reforzado. El objetivo del estudio es permitir que el dron de vigilancia encuentre de manera autónoma la ruta óptima mediante el uso de un algoritmo de aprendizaje propuesto.

Para lograr este objetivo, se mejora el aprendizaje por refuerzo tradicional mediante la optimización de parámetros claves, como el coeficiente de tasa de aprendizaje, los criterios de convergencia y la detección de convergencia de errores para los procesos de políticas *greedy*. Estas políticas, también conocidas como aproximaciones simples, implican seleccionar aleatoriamente un valor dentro de un conjunto y ajustarlo a medida que se realizan iteraciones. La detección de convergencia de errores es un método de aprendizaje que proporciona retroalimentación sobre los valores generados por las políticas *greedy*.

En el trabajo se logra un resultado de simulación que consistió en 500 intentos con un solo dron de vigilancia en un área de cuadrícula desconocida. Se obtuvo una ganancia promedio de 500 con una desviación estándar de 2,82, utilizando un *learning rate* de 0,891. Este resultado se comparó con otro artículo, donde la principal diferencia radicaba en el número de iteraciones, con una ventaja de 69 147 iteraciones menos a favor del modelo propuesto.

2.4 Consideraciones finales

En esta sección, se destacan los trabajos futuros que se enfocan en mejorar la eficiencia de los algoritmos de planificación de rutas de drones. Por ejemplo, Bouman et al. (2018) plantean en el futuro la búsqueda de un algoritmo de programación dinámica con el objetivo de planificar rutas óptimas para múltiples drones, teniendo en cuenta restricciones de energía y tiempo de vuelo. De manera similar, Yi y Sutrisna (2021) también adoptan este enfoque en su artículo.

Por otro lado, en el artículo de Poikonen et al. (2019), se sugiere como trabajo futuro resolver el problema de la planificación de rutas de drones con múltiples puntos de lanzamiento y aterrizaje. Para abordar este desafío, los autores proponen utilizar una técnica de ramificación y acotamiento con el fin de encontrar soluciones óptimas.

En general, estos trabajos futuros tienen como objetivo mejorar la capacidad de los modelos de planificación de rutas al incrementar el número de drones y puntos de salida y llegada. En este sentido, el presente trabajo aborda ambos aspectos mediante la implementación de segmentos y la capacidad añadida al planificador de dividir los viajes y obtener una ruta para más de un dron.

3. METODOLOGÍA

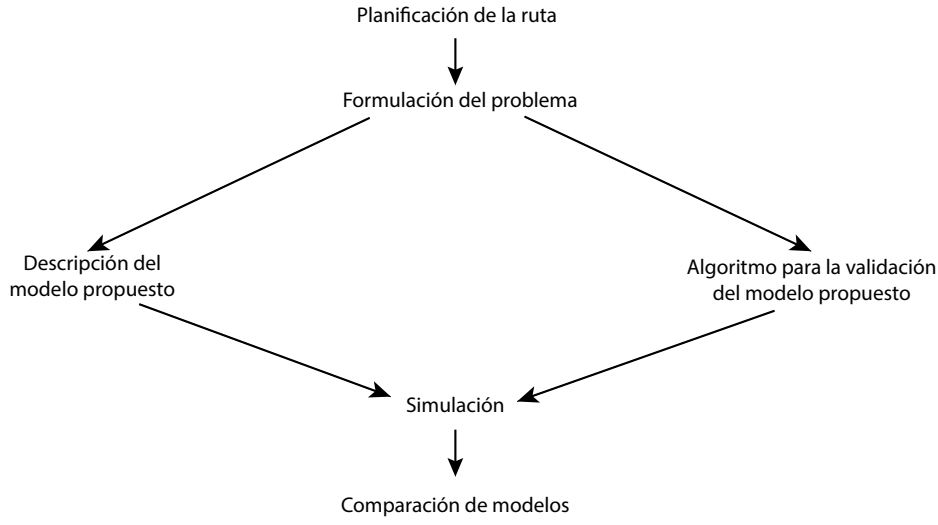
La metodología utilizada en este estudio se basa en los artículos de Yi y Sutrisna (2021) y de Khan et al. (2021), lo que implica un total de cinco etapas. En la primera, se realiza la planificación de la ruta sobre la cual se construirán los planes de vuelo, al mismo tiempo que se recopila información técnica sobre el vuelo de los drones, como la distancia mínima entre ellos, la velocidad, el consumo de energía, la aceleración y la velocidad mínima de viaje por segmento.

En la segunda etapa, se formula el problema identificando las variables y restricciones que se utilizarán para construir el modelo. Luego, en la tercera etapa, se desarrolla el modelo en sí. Dado que se hará una comparación entre algoritmos, en la siguiente etapa se describen los dos modelos propuestos: el algoritmo de programación dinámica y el algoritmo genético.

Finalmente, se añade una etapa en la que se implementan ambas propuestas en diferentes escenarios, con el objetivo de explicar posteriormente los resultados obtenidos. Todo este proceso se presenta de manera esquemática en la Figura 1.

Figura 1

Diagrama de flujo de la metodología propuesta



3.1 Planificación de la ruta

Para llevar a cabo la experimentación, se utilizó como modelo de construcción la zona en obras de la Universidad de Lima. Se seleccionó una ruta frecuente para la toma de videos con fines de supervisión de obra y fotogrametría. El mapa de esta zona se construyó utilizando la herramienta DroneDeploy, con una altitud de 80 metros y considerando la prevención del terreno. Se estableció una superposición frontal del 75 % y una superposición lateral del 70 %. El resultado de este proceso se muestra en la Figura 2, junto con las distancias correspondientes que se detallan en la Tabla 1.

En la Figura 2, se puede observar una ruta representada por líneas verdes paralelas, generadas mediante la herramienta mencionada anteriormente. Cada segmento de la ruta, denominado *k*, está compuesto por una línea vertical y otra que lo conecta con el siguiente segmento. Se han obtenido los primeros seis segmentos de la ruta, los cuales se detallan en la Tabla 1 y se grafican en la Figura 3. Con fines experimentales,

se han utilizado las velocidades mínimas descritas en el artículo de Yi y Sutrisna (2021).

Figura 2

Mapa de ruta de drones

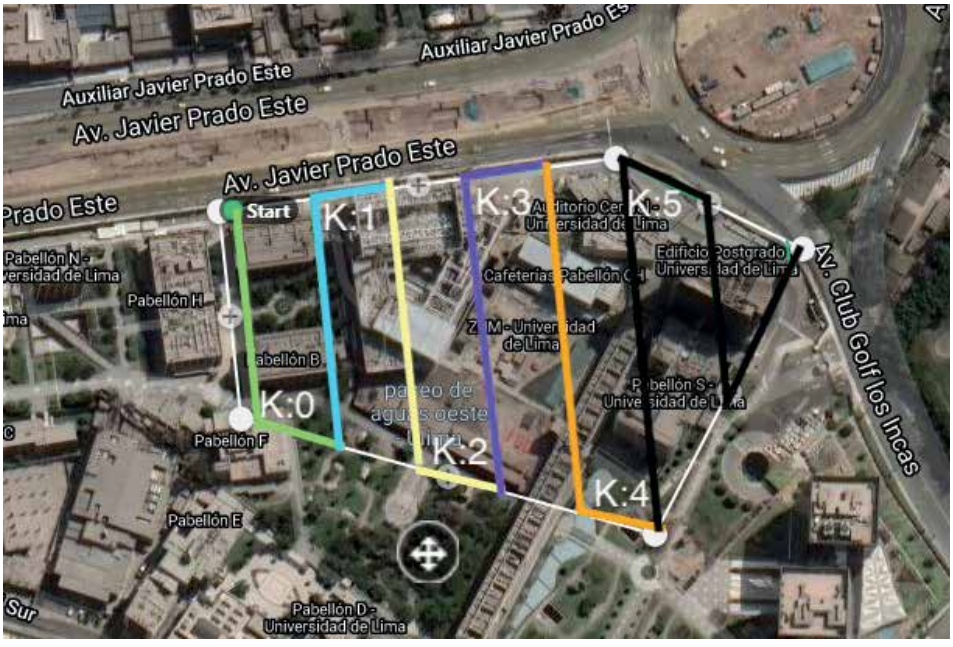
Tabla 1

Distancia por segmento

Nombre	Distancia (m)	Tiempo (min)
K = 0	106	428,62
K = 1	116	282,38
K = 2	136	250,48
K = 3	145	233,34
K = 4	157	196,16
K = 5	305	183,16

Figura 3

Detalle de mapa por segmento



3.2 Descripción del modelo propuesto

Para determinar las variables necesarias, se tomarán en consideración aquellas utilizadas por Yi y Sutrisna (2021), junto con el dron profesional de referencia, el Mavic 2 Pro DJI, de DJI (2020). Las variables se presentan en la Tabla 2.

El dron seleccionado posee tres velocidades de vuelo, y para la simulación se utilizará el modo P o normal, con una velocidad máxima de $V_{\max} = 8\ \mathrm{m/s}$. Además, tiene una batería con capacidad de $Q = 59{,}29\ \mathrm{Wh}$, una aceleración de $\alpha, \beta = 2\ \mathrm{m/s^2}$, y un consumo extra de energía de $\rho^1 = 5\ \mathrm{W}$ al acelerar y $\rho^1 = -2\,\mathrm{W}$ al desacelerar.

Tabla 2

Variables

Variable	Nombre
Q	Batería total
α	Aceleración
β	Desaceleración
T	Tiempo máximo
ρ^1	Consumo de energía cuando α o $\beta \neq 0$
ρ^2	Consumo de energía cuando α o $\beta = 0$
$V_{\text{max.}}$	Velocidad máxima
$V_{\text{mín.}}$	Velocidad mínima
t_k	Tiempo por segmento
p_k	Tiempo mínimo por segmento

El modelo que se va a usar está basado en el propuesto por Yi y Sutrisna (2021), cuya función de ganancia está en la ecuación 1, donde p_k es el tiempo mínimo que el dron debe pasar por una determinada superficie y t_k es el tiempo que realmente usa; eso quiere decir que lo que se busca es maximizar el tiempo sobrante que un dron tiene para recorrer el área, por tanto, pk no debe sobrepasar t_k, como se observa en la ecuación 3. Para abordar el problema, se divide el recorrido en segmentos $K = 0 \ldots k$ de medida l. La capacidad energética del dron está dada por $Q(\text{Wh})$, la velocidad es $V_{\text{mín.}} < v_k < V_{\text{máx.}}$ (como se observa en la ecuación 11); cuando el dron vuela de manera constante, la velocidad puede variar ligeramente, así que depende de la función $F(v)$; en esta primera prueba: $F(v) = v$ y la velocidad del primer segmento será la obtenida por la primera iteración del modelo, como en la ecuación 7. La aceleración es marcada por $\alpha(\text{m/s}^2)$ y la desaceleración por $\beta(\text{m/s}^2)$; esto será de utilidad junto con el gasto de energía por aceleración $\rho 1(\text{W})$ y $\rho 2(\text{W})$ por desaceleración. Además, existe un tiempo máximo mostrado por la ecuación 6 y uno inicial dado por la ecuación 4, mientras el tiempo de cada segmento está dado por las ecuaciones 2 y 5.

$$Ganancia\{-\infty, si\ t_k < p_k\ p_k * (t_k - p_k), si\ no \tag{1}$$

$$t_k\{\frac{v_k - v_{k-1}}{\alpha} + \frac{l_k - l_{k-1} - \frac{v_k^2 - v_{k-1}^2}{2\alpha}}{v_k}\ si\ v_k \geq v_{k-1}\ \frac{v_k - v_{k-1}}{\beta} + \frac{l_k - l_{k-1} - \frac{v_k^2 - v_{k-1}^2}{-2\beta}}{v_k}\ si\ no\ k = 1, \ldots, K \tag{2}$$

$$t_k \geq p_k, k = 1, \ldots, K \tag{3}$$

$$T_0 = 0 \tag{4}$$

$$T_k = T_{k-1} + t_k, k = 1, \dots, K \tag{5}$$

$$T_k \leq T \tag{6}$$

$$v_0 = v_k \tag{7}$$

La energía consumida viene siendo dada por q_k^1 en caso de aceleración (ecuación 8) o desaceleración, y por q_k^2 (ecuación 9) en velocidad constante, ambos en Wh. Estos no pueden consumir más batería de la que tiene el dron con la ecuación 10.

$$q_k^1 = \{\rho_1 \frac{v_k - vv-1}{\alpha}, si\ v_k \geq v_{k-1}\ \rho_2 \frac{v_k - vv-1}{-\beta}, si\ no\ k = 1, \dots, K \tag{8}$$

$$q_k^2 = \{F(v_k)(l_k - l_{k-1}\frac{(v_k)^2 - v_{k-1}^2}{2\alpha}), si\ v_k \geq v_{k-1}\ F(v_k)(l_k - l_{k-1}\frac{(v_k)^2 - v_{k-1}^2}{-2\beta}), si\ no\ k = 1, \dots, K \tag{9}$$

$$\sum_{k=1}^{k} \quad (q_k^1 + q_k^2) \leq Q \tag{10}$$

$$V_{mín.} \quad \leq v_k V_{máx.} \quad , k = 1, \dots, K \tag{11}$$

Estas restricciones solo aplican para un único dron, por lo que se deben agregar ciertas consideraciones para poder implementar más drones. Una de ellas es que la ruta deberá ser dividida en la cantidad de drones que se usarán y se debe mantener un área de seguridad para cada dron para evitar interferencias. Para solucionar este problema, se deben crear variables que denominan cuántos drones existen, siendo n el número de drones y k_n el número de segmentos que le corresponden a cada dron. Además, se añaden las variables *ta* y *la*, que representan el tiempo que consume cada dron por segmento y la distancia de cada dron por segmento, las cuales se definen respectivamente por las ecuaciones 12 y 13, y 14 y 15. Otra restricción es que las velocidades no pueden sobrepasar los límites de $V_{\text{máx.}}$ y $V_{\text{mín.}}$, parámetros que se muestran en la ecuación 11. Por último, la lógica para la restricción implica saber la distancia acumulada de los segmentos y el tiempo acumulado; en el caso de que el tiempo acumulado en un dron sea menor que el tiempo acumulado del siguiente dron, se comparan sus distancias acumuladas y se verifica que sean menores que *D*, la distancia mínima entre drones; esto se comprueba en la ecuación 16.

$$ta_{n,1} = t_{1+n\frac{K}{N}}, n = 1 \dots N \quad (12)$$

$$ta_{n,kn} = t_{1+n\frac{K}{N}} + ta_{n,kn-1}\,, n = 1 \dots N, kn = 2 \dots K/N \quad (13)$$

$$la_{n,1} = l_{1+n\frac{K}{N}}, n = 1 \dots N \quad (14)$$

$$la_{n,kn} = l_{1+n\frac{K}{N}} + la_{n,kn-1}\,, n = 1 \dots N, kn = 2 \dots K/N \quad (15)$$

$$la_{n,kn} - la_{n-1,kn} < D, si\ ta_{n,nk} > ta_{n-1,nk}\,, n = 2 \dots N, kn = 1 \dots K/N \quad (16)$$

Finalmente, las ecuaciones se resuelven utilizando la ecuación 1 y se obtiene el siguiente resultado:

$$V_k(v_0, Q_{k-1}, T_{k-1}, v_{k-1}) = máx._{v_k}\, p_k(t_k - p_k) \quad (17)$$

Al ser dos drones, se dividirán los segmentos a la mitad. El inicio del primero es en $k = 0$ y termina en $k < K/2$, y el segundo dron iniciaría en $k < K/2$ y terminaría en $k = K/2$. Esto deja el algoritmo propuesto en la Figura 4.

Figura 4

Algoritmo propuesto

Algoritmo 1 Algoritmo propuesto

Para cada v_0^1 *en* $V_{mín.}, \dots, V_{máx.}$ **hacer:**

Para cada k **en** $K/2,\dots,1$ **hacer:**

Si $k \in \{k-1, \dots, 1\}$ **entonces:**

Enumere todos los valores posibles de Q_{k-1}^1, t_{k-1}^1 y v_{k-1}^1, donde $Q_{k-1}^1 \in \{0, \dots, Q\}$, $t_{k-1}^1 \in \{0, \dots, T\}$ y $v_{k-1}^1 \in \{V_{mín.}, \dots, V_{máx.}\}$. Para cada $(v_0^1, Q_{k-1}^1, t_{k-1}^1, v_{k-1}^1)$, resuelva la ecuación 17.

Fin Si

Fin para cada

Fin para cada

Para cada v_0^2 *en* $V_{mín.}, \dots, V_{máx.}$ **hacer:**

Para cada k **en** $K/2,\dots,1$ **hacer:**

Si $k \in \{k-1, \dots, 1\}$ **entonces:**

Enumere todos los valores posibles de Q_{k-1}^2, t_{k-1}^2 y v_{k-1}^2, donde $Q_{k-1}^2 \in \{0, \dots, Q\}$, $t_{k-1}^2 \in \{0, \dots, T\}$ y $v_{k-1}^2 \in \{V_{mín.}, \dots, V_{máx.}\}$. Para cada $(v_0^2, Q_{k-1}^2, t_{k-1}^2, v_{k-1}^2)$, resuelva la ecuación17.

Fin Si

Fin para cada

Fin para cada

3.3 Algoritmo genético propuesto

El algoritmo genético está compuesto por las mismas condiciones que el algoritmo de programación dinámica, pero se utiliza la función de ganancia para obtener el *fitness*. Este se encuentra en el algoritmo de la Figura 5.

Crear población inicial: para la población inicial, se genera de manera aleatoria un conjunto de individuos que estén dentro del rango de velocidades *V*. El número de individuos se da por el tamaño_población. Estas velocidades se encuentran en el rango que permiten los drones profesionales Mavic DJI (2020).

Crear plan: una vez obtenida la población inicial de velocidades, se les aplicará las ecuaciones del modelo para obtener la ganancia, que posteriormente serán guardadas en un arreglo llamado *fitness*. Después, se verifican las restricciones y en caso de que no se cumpla alguna se reemplaza el valor del arreglo por 0. De esta manera, se asegura que toda la descendencia cumpla las restricciones. Por último, se promedian los valores del *fitness*.

Cruzamiento: se emparejan los individuos aptos tomándolos por pares. Aquellos que presenten un mayor valor en el arreglo tienen una mayor probabilidad de ser seleccionados. Posteriormente, se genera una descendencia combinando la mitad de las velocidades del primer padre con las del segundo.

Mutación: finalmente, se generará una mutación con una probabilidad de 0,05 similar al de Fu et al. (2012).

Se creó una función de parada que tiene como condición que el individuo con mayor ganancia sea por lo menos 90 % tan alto como el obtenido por el algoritmo de programación dinámica. Esto no es necesario para el modelo de programación dinámica porque, en ese caso, se produce una sola respuesta posible al final de las iteraciones, mientras que la respuesta del algoritmo genético puede variar dependiendo de los parámetros iniciales y de las generaciones que necesite (esto se corrige con la función de parada explicada anteriormente).

Es importante señalar que el algoritmo genético genera diferentes métricas en cada lanzamiento; por ello, se realizó una función que genera varias iteraciones para cada escenario. Posteriormente, se analizan los datos usando estadística descriptiva (media, máximo, mínimo y desviación estándar).

Figura 5

Algoritmo genético

Algoritmo 2 Algoritmo genético

```
fitness = [ ]
descendencia = [ ]
población←CrearPoblación(K, V, tamaño_población)
Mientras fitness < (maxGanDina * 0.999) hacer:
    Para cada v en población hacer:
        plan←CrearPlan(v)
        Si plan cumple las restricciones entonces:
            fitness← plan resolver 1
        Si no:
            fitness←0.0
        Fin si

        fitness←∑ fitness/tamaño_población

        Para cada i en tamaño_población/2 hacer:
            padres← Escoger 2 padres aleatorios
            padres = CrossPoint(padres)
            descendencia←padres
        Fin para cada
        Población = descendencia
        Mutar(población)
    Fin para cada
Fin
```

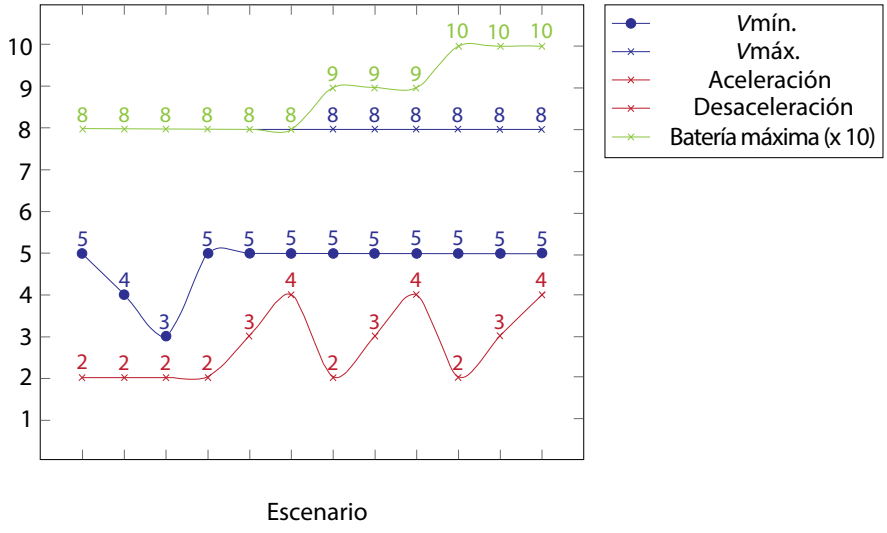
4. RESULTADOS

La implementación se hizo en Python usando un procesador AMD A10-9600P Radeon R5, 16 GB RAM, GPU: NVIDIA Rtx 3070. La complejidad de un escenario está basada en los siguientes factores: la velocidad de desplazamiento de un dron y el número de segmentos de un área geográfica. Para la prueba de concepto, se considera el campus universitario de la sede principal de la Universidad de Lima, ubicada en Surco. Esta área geográfica se ha distribuido en seis segmentos, como se observa en la Figura 2 y cuyas medidas están en la Tabla 1. Cabe destacar que esta distribución fue tomada considerando las características del dron Mavic 2 Pro. Entre las restricciones, se encuentran las de aceleración y desaceleración, las cuales son de 2 m/s^2; el consumo de batería, que va a razón de 5 Wh; y una batería de 80 Wh. También se usó como referencia la distancia mínima de la Tabla 1 para la distancia mínima entre dos drones, siendo esta de 106 metros. El número de posibles respuestas, en este caso, será una variación con repetición de velocidades con respecto a segmentos, o por sus siglas VR.

Debido a lo mencionado anteriormente, los escenarios irán aumentando en complejidad con el fin de evaluar el rendimiento de los modelos. Para ello pueden modificarse dos *inputs*: velocidad o segmentos. En esta ocasión, se modificó las velocidades: en el escenario 1, el rango de velocidades será de 4; en el 2, será de 5; y en el 3, será de 6, como se muestra en la Tabla 3. Esto hará que la complejidad aumente, ya que las posibles respuestas serán de 4VR6, 5VR6 y 6VR6. Además, se crearon más escenarios aumentando la batería, las aceleraciones y las desaceleraciones desde el escenario 4 al 12.

Figura 6

Escenarios



La Figura 6 muestra doce escenarios distintos en el eje *X*, cada uno de los cuales se caracteriza por una combinación única de variables utilizadas para evaluar el modelo. Los primeros tres escenarios se centran en la evaluación de los valores $V_{mín.}$ (velocidad mínima) y $V_{máx.}$ (velocidad máxima), mientras que los siguientes nueve escenarios abordan otras variables relevantes. En particular, los rangos de aceleración y desaceleración varían de 2 m/s^2 a 4 m/s^2, la $V_{mín.}$ oscila entre 3 m/s y 5 m/s, y la capacidad máxima de la batería del dron se sitúa entre 80 Wh y 100 Wh.

4.1 Casos con datos simulados en los algoritmos propuestos

Finalmente, se muestran los resultados de ambos algoritmos. Cabe recordar que la ganancia se define por la ecuación 1 del modelo.

Tabla 3

Resultados en el escenario 1

Programación dinámica	Ganancia	Tiempo				
	3473,75	0,02				
Segmento	1	2	3	4	5	6
Velocidades	8	5	5	5	6	5
Algoritmo genético	Ganancia	Tiempo				
		(máx.	mín.	med	desc)	
	3473,75	5,78	1,3	3,38	2	
Segmento	1	2	3	4	5	6
Velocidades	7	7	5	5	6	5

Para el escenario 1, mostrado en la Tabla 3, se consideró un rango de velocidades de 3. Se obtuvo 3473,75 de ganancia. En este caso, el número de posibles soluciones estaría definido por 4VR6, por lo que el modelo de programación dinámica logró 0,02 segundos, mientras que el algoritmo genético alcanzó 3,38, lo que da 3,36 segundos de diferencia.

Tabla 4

Resultados en el escenario 2

Programación dinámica	Ganancia	Tiempo				
	3481	0,07				
Segmento	1	2	3	4	5	6
Velocidades	8	4	4	4	8	4
Algoritmo genético	Ganancia	Tiempo				
		(máx.	mín.	med	desc)	
	3481	70,37	3,58	21,14	19	
Segmento	1	2	3	4	5	6
Velocidades	8	4	7	4	6	4

En la Tabla 4, se observan los resultados en el escenario 2, donde se consideró un rango de velocidades de 4. Se obtuvo el siguiente plan con una ganancia de 3481. En este caso, el número de posibles soluciones estaría definido por 5VR6, por lo que el modelo

de programación dinámica logró 0,07 segundos, mientras que el algoritmo genético alcanzó 21,14 segundos, dando 21,07 segundos de diferencia.

Tabla 5

Resultados en el escenario 3

Programación dinámica	Ganancia	Tiempo				
	3495,75	17,04				
Segmento	1	2	3	4	5	6
Velocidades	5	3	3	5	8	3
Algoritmo genético	Ganancia	Tiempo				
		(máx.	mín.	med	desc)	
	3482,25	1263,41	9,58	836,64	442	
Segmento	1	2	3	4	5	6
Velocidades	8	3	5	3	8	3

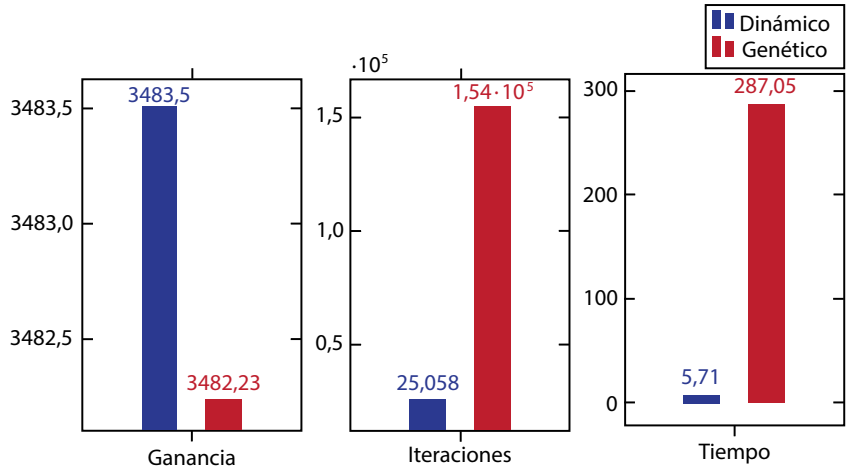
La Tabla 5 muestra el escenario 3, donde se consideró una velocidad máxima de 8 y una velocidad mínima de 3, dando como resultado de su resta un rango de velocidades de 5. De esta manera se obtuvo el siguiente plan con una ganancia de 3495,75 para el algoritmo de programación dinámica y 3482,25 para el algoritmo genético. En este caso, el número de posibles soluciones estaría definido por 6VR6, por lo que el modelo de programación dinámica logró 17,04 segundos, mientras que el genético alcanzó 836,64 segundos, con una diferencia de 819,6 segundos.

Se puede observar una mejora en cuanto a la diferencia de algoritmos, que puede llegar hasta varios minutos, además de que ambos obtuvieron resultados similares usando métodos distintos. Esto muestra que el algoritmo de programación dinámica posee una mayor eficiencia que el genético y calcula correctamente el resultado. Además, tal como se predijo, las velocidades afectan en gran medida al rendimiento de ambos algoritmos, pero mayormente al de programación dinámica. Esto puede ser solucionado con una mayor memorización o probando otros algoritmos.

Finalmente, se obtuvieron los promedios de todos los escenarios y se compararon en la Figura 7, donde se observa la mejora en tiempo y ganancia del algoritmo dinámico. Específicamente, el algoritmo genético obtiene una ganancia de 3462,52, mientras que el dinámico 3475,25 segundos. Esto es importante, ya que muestra una diferencia en el nivel de precisión que se traduce en un mayor tiempo de vuelo y, por tanto, mayor tiempo de supervisión. Además, existe una disminución del tiempo de ejecución del 98,01 %.

Figura 7

Comparación general



5. DISCUSIÓN

En esta sección, se obtuvieron los promedios de los resultados presentados en la Figura 7, los cuales revelan una diferencia de 281,34 segundos o 4 minutos y 47 segundos en cuanto al tiempo de ejecución. Esta variación se debe al incremento de genes en los escenarios, que resulta en un aumento en la cantidad de individuos posibles en las poblaciones. Además, el hecho de que el algoritmo genético genere estos individuos de manera aleatoria termina afectando negativamente el tiempo de ejecución.

Debido a que el algoritmo genético usa una función de parada, tiene un número de generaciones dinámico. Por tanto, aumenta el número de iteraciones, pero reduce el tiempo de ejecución.

Para reforzar este punto y tomando en cuenta que las variables que se usaron en la experimentación son las mismas para ambos algoritmos, se realiza un test pareado para determinar si existe una diferencia significativa en los resultados y el tiempo de ejecución de cada solución. Los datos utilizados para las pruebas son los resultados de ganancia y tiempo de ejecución arrojados por el algoritmo genético y el dinámico, las variables de entrada empleadas para dichos algoritmos se detallan en los escenarios del 4 al 12 de la Tabla 3.

En la Tabla 6, se observan las variaciones de los resultados en cuanto a tiempo de ejecución en la desviación estándar, así como la media para cada algoritmo. De igual manera, se aprecian en la Tabla 7 las ganancias obtenidas por cada uno. Se utilizó una

prueba de hipótesis de igualdad (=) para poder destacar que efectivamente existe una diferencia significativa entre ambos algoritmos y, posteriormente, se compararon sus medias para distinguir cuál obtuvo un mejor desempeño.

Tabla 6

Tiempo de ejecución

	N	Media	Desv. est.	*p*-valor
Dinámico	9	0,08	0,2	0,003
Genético	9	5,04	31,8	

Null hypothesis	$H_0: \mu_{difference} = 0$
Alternative hypothesis	$H_1: \mu_{difference} \neq 0$

Tabla 7

Beneficio

	N	Media	Desv. est.	*p*-valor
Dinámico	9	3463,82	10,83	0,0071
Genético	9	3464,15	11,29	

Null hypothesis	$H_0: \mu_{difference} = 0$
Alternative hypothesis	$H_1: \mu_{difference} \neq 0$

Volviendo a la Tabla 6, se tiene como hipótesis 1 que el tiempo de ejecución del genético – dinámico ≠ 0, en este caso, el *p*-valor es de 0,003, por lo que se aprueba esta hipótesis. Considerando que la media del algoritmo genético es inferior a la del dinámico, se concluye que su tiempo de ejecución es significativamente inferior. De igual manera, en la Tabla 8, se tiene como hipótesis 1 que el beneficio del dinámico – genético ≠ 0, en este caso, el *p*-valor es de 0,0071, por lo que se aprueba esta hipótesis. Y considerando que la media del algoritmo dinámico es superior a la del genético, se concluye que el dinámico tiene un beneficio significativamente superior al genético.

6. CONCLUSIONES

Al principio, en este trabajo se presentaron los distintos algoritmos usados para la creación de planificadores de vuelos, entre los que se destacaron aquellos de programación dinámica y algoritmo genético. Se optó por programación dinámica y se usó como base

el trabajo de Yi y Sutrisna (2021). A partir de ello se creó un algoritmo que funcione con una problemática en la que se requiera de múltiples drones.

Posteriormente, se hizo un levantamiento de información en el que se recolectaron las variables que se usarían en los distintos escenarios, junto con las especificaciones del dron que se empleó (Mavic 2 Pro). Finalmente, se implementó el algoritmo dinámico y el genético en Python para compararlos. Este último utilizó el mismo modelo y función objetivo.

En los resultados, se obtuvo una mejora considerable del algoritmo en programación dinámica con respecto al algoritmo genético, y se cumplieron todas las restricciones del modelo usando un número de permutaciones y recursos óptimos, como se observa en el apartado de la discusión.

De esta manera se consiguieron distintos planes para diferentes variables. Por ejemplo, en caso de usar dos drones con los parámetros del escenario 1 (explicados en la Figura 6), se programa al dron 1 con las siguientes velocidades: 8, 5, 5, para los primeros tres segmentos. El segundo dron se programa con las siguientes velocidades 5, 6, 5, para los últimos tres segmentos.

Además, existe una mejora con respecto al tiempo de ejecución y el beneficio obtenido por parte de la programación dinámica, lo que lo hace más preciso y más rápido. Sin embargo, el modelo aún puede ser mejorado en cuanto a su complejidad. Esto se debe a que sigue requiriendo de muchas permutaciones para su ejecución, sobre todo en modelos deterministas.

Por ello, algunos futuros trabajos podrían incluir utilizar metodologías de conjuntos de drones enjambre y la implementación de estos algoritmos en un entorno real de construcción. También pueden usarse otros modelos de drones relacionados con labores de ingeniería como nuevos escenarios, por ejemplo, el Matrice de DJI.

REFERENCIAS

Albeaino, G., & Gheisari, M. (2021). Trends, benefits, and barriers of unmanned aerial systems in the construction industry: A survey study in the United States. *Journal of Information Technology in Construction*, *26*, 84-111. https://doi.org/10.36680/j.itcon.2021.006

Balfour Beatty. (2017, 12 de mayo). *Flying into the future of bridge inspections*. https://www.balfourbeatty.com/news/flying-into-the-future-of-bridge-inspections/

Bouman, P., Agatz, N., & Schmidt, M. (2018). Dynamic programming approaches for the traveling salesman problem with drone. *Networks*, *72*(4), 528-542. https://doi.org/10.1002/net.21864

Criado, R. M., & Rodríguez Rubio, F. (2015). Autonomous path tracking control design for a comercial quadcopter. *IFAC-PapersOnLine*, *48*(9), 73-78. https://doi.org/10.1016/j.ifacol.2015.08.062

Decreto Supremo 011-2019-TR [Ministerio de Trabajo y Promoción del Empleo]. Decreto Supremo que aprueba el Reglamento de Seguridad y Salud en el Trabajo para el Sector Construcción. 11 de julio del 2019. Diario oficial *El Peruano*. https://cdn.www.gob.pe/uploads/document/file/341232/decreto-supremo-n-011-2019-tr-1787274-4.pdf?v=1562856062

DJI. (2020). *Mavic 2 Pro/Zoom. User Manual*. DJI.

Doole, M., Ellerbroek, J., & Hoekstra, J. (2020). Estimation of traffic density from drone-based delivery in very low level urban airspace. *Journal of Air Transport Management*, *88*, 101862. https://doi.org/10.1016/j.jairtraman.2020.101862

Fan, J., & Saadeghvaziri, M. A. (2019). Applications of drones in infrastructures: Challenges and opportunities. *World Academy of Science, Engineering and Technology International Journal of Mechanical and Mechatronics Engineering*, *13*(10), 649-655. https://doi.org/10.5281/zenodo.3566281

Fu, S.-Y., Han, L.-W., Tian, Y., & Yang, G.-S. (2012). Path planning for unmanned aerial vehicle based on genetic algorithm. En *2012 IEEE 11th International Conference on Cognitive Informatics and Cognitive Computing* (pp. 140-144). IEEE. https://doi.org/10.1109/ICCI-CC.2012.6311139

Hrishikeshavan, V., & Chopra, I. (2017). Refined lightweight inertial navigation system for micro air vehicle applications. *International Journal of Micro Air Vehicles*, *9*(2), 124-135. https://doi.org/10.1177/1756829316682534

Hromkovič, J. (2013). *Algorithmics for Hard Problems: Introduction to Combinatorial Optimization, Randomization, Approximation, and Heuristics*. Springer Science & Business Media.

Khan, S. I., Qadir, Z., Munawar, H. S., Nayak, S. R., Budati, A. K., Verma, K. D., & Prakash, D. (2021). UAVs path planning architecture for effective medical emergency response in future networks. *Physical Communication*, *47*, 101337. https://doi.org/10.1016/j.phycom.2021.101337

Lee, D., & Cha, D. (2020). Path optimization of a single surveillance drone based on reinforcementlearning. *International Journal of Mechanical Engineering and Robotics Research*, *9*(12), 1541-1547. https://doi.org/10.18178/ijmerr.9.12.1541-1547

Li, Y., Liu, H., Zheng, X., Han, Y., & Li, L. (2019). A top-bottom clustering algorithm based on crowd trajectories for small group classification. *IEEE Access*, *7*, 29679-29698. https://doi.org/10.1109/ACCESS.2019.2902310

Ministerio de Vivienda, Construcción y Saneamiento. (2020). *Lineamientos de prevención y control frente a la propagación del COVID-19 en la ejecución de obras de construcción*. https://cdn.www.gob.pe/uploads/document/file/671272/Lineamiento_de_Prevencion_y_Control_del_COVID-19_en_Obras_Construccion.pdf

Nguyen, M. A., Dang, G. T.-H., Hà, M. H., & Pham, M.-T. (2022). The min-cost parallel drone scheduling vehicle routing problem. *European Journal of Operational Research*, *299*(3), 910-930. https://doi.org/10.1016/j.ejor.2021.07.008

Palomino, J., Hennings, J., & Echevarría, V. (2017). Análisis macroeconómico del sector construcción en el Perú. *Quipukamayoc*, *25*(47), 95-101. https://doi.org/10.15381/quipu.v25i47.13807

Poikonen, S., Golden, B., & Wasil, E. A. (2019). A branch-and-bound approach to the traveling salesman problem with a drone. *INFORMS Journal on Computing*, *31*(2), 335-346. https://doi.org/10.1287/ijoc.2018.0826

Sando. (2021, 2 de agosto). Sando logra la habilitación para operar drones en espacios aéreos controlados, zonas urbanas y vuelos nocturnos. *Sando blog.* https://www.sando.com/es/drones-sando-aesa-operar-zonas-urbanas-nocturnas/

Schermer, D., Moeini, M., & Wendt, O. (2020). The traveling salesman drone station location problem. En H. Le Thi, H. Le & T. Pham Dinh (Eds.), *Optimization of Complex Systems: Theory, Models, Algorithms and Applications. WCGO 2019* (pp. 1129-1138). Springer. https://doi.org/10.1007/978-3-030-21803-4_111

Wankmüller, C., Truden, C., Korzen, C., Hungerländer, P., Kolesnik, E., & Reiner, G. (2020). Optimal allocation of defibrillator drones in mountainous regions. *OR Spectrum*, *42*(3), 785-814. https://doi.org/10.1007/s00291-020-00575-z

Yi, W., & Sutrisna, M. (2021). Drone scheduling for construction site surveillance. *Computer-Aided Civil and Infrastructure Engineering*, *36*(1), 3-13. https://doi.org/10.1111/mice.12593