Efficient Drone Path Planning through Strategic Launch Pad Positioning

G. Gasteratos [1], and I. Karydis [1]
[1] Ionian University, Dept. of Informatics, 49132 Kerkyra, Greece
Tel.: + 30 26610 87423
E-mail: ggasteratos@ionio.gr, karydis@ionio.gr

Summary: Efficient drone path planning is critical for optimizing the performance of unmanned aerial systems, particularly in applications requiring extensive coverage and precision. This study explores the impact of strategic launch pad positioning on drone path planning, emphasizing its role in minimizing energy consumption and improving operational efficiency. Utilizing the multiple Traveling Salesman Problem (mTSP) algorithm, the research investigates how repositioning the launch pad can influence the assignment and configuration of drone routes. Results demonstrate that optimizing the launch pad location significantly reduces redundant travel distances, and enhances overall mission performance. This approach underscores the importance of adaptive deployment strategies in the design of efficient, energy-aware drone systems.

Keywords: Path planning optimization, Drone path planning, Starting Point Selection, mTSP.

1. Introduction

Path planning in drones is an essential aspect in unmanned aerial vehicle (UAV) operations, ensuring that the drone flies efficiently while meeting the mission requirements. Applications range from disaster relief and agricultural monitoring to logistics and surveillance, where effective path planning directly contributes to operational success by optimizing resource utilization, such as battery life and time [1].

As drone technology advances, path planning has become an interdisciplinary research area by incorporating robotics, artificial intelligence (AI), mathematics, and GIS [2, 3]. Nonetheless, the main focus still remains on calculating the best routes from start to destination, taking into account mission-specific factors. From delivery of medical supplies to disaster-stricken areas to real-time crop monitoring, path planning is key to improving efficiency and reducing costs in UAV operations.

1.1. Motivation & Contribution

The rapid growth of drone applications presents both opportunities and challenges in the realm of path planning. Modern applications demand innovative solutions that can handle increasing operational complexities, such as multi-drone coordination, real-time environmental adaptability, and energy efficiency. These challenges underscore the need for robust path-planning algorithms that not only optimize routes but also account for constraints like terrain, weather, and communication reliability.

One critical motivator for path-planning optimization is energy efficiency. Drones have limited battery capacity, which directly affects their operational range. Path-planning algorithms that minimize travel distance and energy consumption can significantly extend mission durations and expand the scope of applications. This article contributes to this ongoing effort by consolidating the latest findings and methodologies in drone path planning optimization. More specifically, the key contributions of this work can be summarized as follows:

- Presents the significant role of strategic launch pad positioning in enhancing the efficiency of drone path planning.
- Introduces a timely solution for identifying a near-optimal launch pad for a flight plan.
- Enhances the framework presented in [1] with the addition of two algorithms that aid in the prediction of the optimal launch pad.

The structure of the paper is as follows: Section 2 reviews relevant literature on drone path planning. Section 3 introduces the proposed framework, while Section 4 outlines the experimental setup, presents the results, and provides an evaluation of the findings. The paper concludes in Section 5.

2. Background and related work

Recent research on drone path planning has focused on various innovative methodologies to enhance the operational efficiency of UAVs in complex and dynamic environments. As the operational landscape of drones becomes increasingly intricate, researchers have developed a range of strategies to address the challenges associated with path planning, including obstacle avoidance, energy efficiency, and environmental adaptability.

One significant area of advancement is the development of algorithms that optimize path planning under various constraints. Fan et al. [4] introduce a path planning method that uses Dubins paths to ensure smooth turns making sure to avoid restricted areas and directional constraints, thus improving mission efficiency for flight plans with long distances. Similarly, Xiong et al. [5] introduce a hybrid approach combining Improved Symbiotic Organisms Search (ISOS) with Sine-Cosine Particle Swarm Optimization (SCPSO) methods, enhancing the precision and stability of path planning in three-dimensional environments. These methods highlight the importance of adapting path planning algorithms to the specific operational contexts of drones.

Moreover, the integration of advanced technologies such as the Internet of Drones (IoD) [6] has been explored to optimize UAV path planning further. Shirabayashi [7] discusses the implications of IoD on path planning strategies, emphasizing the need for mathematical models that can accommodate the complexities introduced by interconnected drone networks. This integration not only facilitates better communication among drones but also enhances their ability to navigate dynamically changing environments.

The consideration of environmental factors is also critical in recent studies. Jones et al. [8] provide a comprehensive survey on the impact of environmental complexity on UAV path planning, identifying key challenges and proposing future research directions. The survey therein underscores the necessity for path-planning algorithms that can adapt to real-time changes in the environment, ensuring safe and efficient drone operations.

The application of swarm intelligence techniques has emerged as a promising avenue for enhancing drone path planning. For example, Wu et al. [9] propose a swarm-based 4D path planning method tailored for urban environments,

which addresses the complexities associated with multiple drones operating simultaneously. This approach not only improves flight safety but also optimizes the overall efficiency of drone operations in densely populated areas.

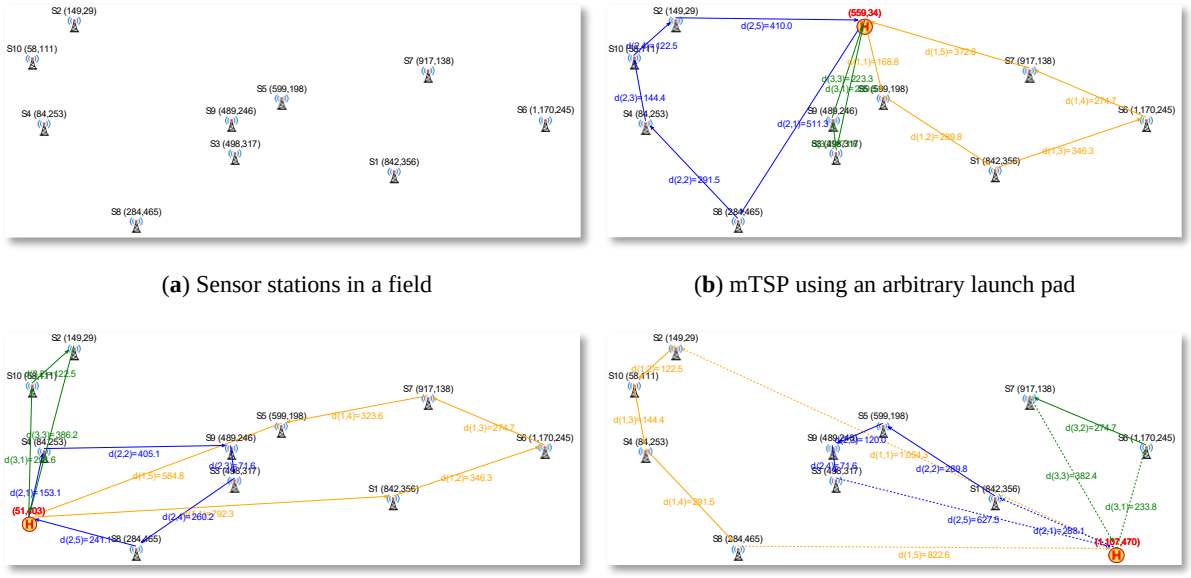
In addition, energy efficiency remains a pivotal concern in drone path planning. Diller's [10] research emphasizes the trade-off between speed and energy consumption, advocating for path-planning approaches that account for energy constraints while maximizing operational speed. This focus on energy-aware planning is crucial for extending the operational range of drones, particularly in applications such as surveillance and agricultural monitoring.

In the same area, Gasteratos and Karydis [1] proposed a path planning optimization technique whereby the starting point distances can be reduced leading to better energy management and operational efficiency. In their research show that when the launch pad is relocated, it minimizes the distance that drones must travel between the launch pad and their first station, as well as the distance from their last station back to the launch pad. This reduction directly translates into lower energy consumption and shorter flight times, enhancing the overall efficiency of the operation.

3. Proposed method

The impact of launch pad repositioning as described by Gasteratos and Karydis in their work [1], goes beyond improving the initial and final segments of each drone's route. The Multiple Traveling Salesman Problem (mTSP) [11, 12] algorithm, which is designed to minimize the total travel distance across multiple drones, calculates the optimal routes based on a predefined starting point, the launch pad, which is common to all drones. Changing the launch pad to a new location and running the algorithm again will generate entirely different route configurations. For instance, relocating the launch pad closer to a cluster of sensor stations may lead the algorithm to reassign some stations to different drones, resulting in shorter and more efficient routes overall.

This flexibility in route optimization highlights a critical aspect of the mTSP algorithm: it continuously seeks to minimize the total travel distance by exploring alternative route structures. When the launch pad is repositioned, the algorithm evaluates how the new location affects the travel distances between the sensor stations and adjusts the routes accordingly, as shown in (Fig. 1). This process not only reduces the travel distance for individual drones, but also balances the workload among the drones more effectively, preventing any single drone from being overburdened with a disproportionately long route. This means that repositioning the launch pad introduces an opportunity for the algorithm to explore different route combinations that may not have been considered optimal under the previous configuration.



(**a**) Sensor stations in a field

(**b**) mTSP using an arbitrary launch pad

(**c**) mTSP using a different launch pad

(**d**) mTSP using a different launch pad

Fig. 1. Changing the launch pad position.

These findings emphasize that the launch pad's location is not merely a logistical decision but a critical variable in the optimization process. Repositioning allows the algorithm to explore new route combinations, often resulting in better performance by minimizing redundant travel distances. This flexibility underscores the role of adaptive deployment strategies in enhancing operational efficiency for drone-based data collection systems. In light of these considerations, our research endeavors to explore methodologies for determining the most advantageous launch pad location for a specified flight plan. As far as we know, no existing research has addressed this particular challenge.

3.1. Establishing the ground truth

The methodology developed in this study must be validated against the ground truth to ensure measurable and reliable results. The initial step in this process is to establish the ground truth for the scenarios under consideration. This involves identifying a methodology that consistently identifies the optimal starting point for the area of interest, while temporarily disregarding the computational time required to achieve this outcome. The worst-case scenario in terms of time complexity is the application of the brute-force method, which involves evaluating every potential point in the area to determine the optimal starting point. Although this method is computationally expensive, it guarantees the identification of the best starting point.

For instance, consider a scenario where sensor stations are distributed across a field, as illustrated in (Fig. 1a), with dimensions of 1496 units in length and 571 units in width. The brute-force approach would require running the mTSP algorithm 1496 × 571 = 854,216 times to evaluate each point in the 2D Euclidean space where the sensor stations are located. Despite the significant computational expense, this method is crucial as it ensures the determination of the optimal starting point, leading to the minimal total traveled distance when the mTSP is applied.

Using equations (Eq. 2) and (Eq. 3) as defined by Cheikhrouhou and Khoufi in their work [12], the optimal starting point can be defined as the one with the minimum total traveled distance after applying mTSP in the form of:

$$OptimalStartingPoint = \min\left(TotalDistance_j\right) \quad (1)$$

whereby the *Total Distance* for *x* drones in a flight plan for a particular starting point *j* is defined as:

$$TotalDistance_j = \sum_{i=1}^{x} D\left(Route_{U_i}\right)_j \quad (2)$$

and the *Route* distance *D* of a drone U_i is defined as the total distance traveled by the drone, starting from its initial point *H*, visiting the assigned ground stations $S_{i_1}, S_{i_2}, \ldots, S_{i_n}$ sequentially in the given order, and then returning to *H*.

$$D\left(Route_{U_i}\right)_j = D\left(H, S_{i_1}\right) + \sum_{k=1}^{n-1} D\left(S_{i_k}, S_{i_{k+1}}\right) + D\left(S_{i_n}, H\right) \quad (3)$$

3.1. The implementation of GA with mTSP

In this study, we address the challenge of finding the optimal starting point in a sensor network distribution, where the computational cost of using brute-force methods can be prohibitive. The brute-force approach guarantees the optimal solution by evaluating every potential starting point, leading to a time complexity proportional to the number of possible evaluations (hundreds of thousands). While this method ensures accuracy, its computational expense grows significantly with larger problem sizes, making it inefficient for large-scale applications. Therefore, it is essential to explore alternative methods that can provide optimal or near-optimal solutions with significantly lower computational cost.

A promising approach to solve this problem efficiently is the use of genetic algorithms (GAs). GAs are heuristic search methods inspired by natural selection and the principles of evolution, which have been successfully applied to a wide variety of combinatorial optimization problems [13, 14]. The central advantage of GAs is their ability to explore large and complex solution spaces without exhaustively evaluating every possible solution, making them computationally feasible even for problems with large numbers of potential solutions [15].

To implement this approach, we enhanced the mTSP modeling framework described in [1] and introduced two additional techniques to identify the optimal starting point: brute-force and GA-based methods. The brute-force approach was straightforward to implement, as it involved reusing existing functionality. To implement the GA, we mapped its key characteristics to the mTSP problem, detailed next, and utilized the GeneticSharp library [16], which makes easier the development of applications using genetic algorithms.

A Genetic Algorithm requires several key features to function effectively [17]. First, it needs a method to represent potential solutions to the problem (the ***chromosome*** in GA terms). We implemented chromosome, in a way that encapsulates all the necessary information for evaluating a solution, such as the coordinates X and Y of the starting point.

Second, a mechanism to ***initialize a population*** of these solutions was implemented. This population serves as the starting point for the evolutionary process and should ideally cover a diverse range of possibilities to ensure a broad exploration of the solution space.

Another essential feature is the ***fitness function*** that quantifies the quality of each solution. This function provides a measurable way to compare solutions, guiding the algorithm toward better outcomes. This was implemented by utilizing the total distance calculated by mTSP.

The GA also needs a method to select solutions for reproduction. For this, the elite selection [18] was used that determines which solutions are more likely to contribute their "genes" to the next generation, favoring those with higher fitness. This ***selection process*** helps to preserve the best solutions found so far and prevents the GA from losing valuable genetic material during the evolutionary process.

To create new solutions, the algorithm relies on ***crossover*** and ***mutation*** operators [19]. Crossover combines parts of two parent solutions to produce offspring, facilitating the exchange of beneficial traits. Uniform crossover was used in this implementation that utilizes a fixed mixing ratio between two parents. Similarly, mutation introduces small random changes to individual solutions, helping the algorithm explore new areas of the solution space and maintain diversity. For this, the uniform mutation was used that replaces the value of the chosen gene with a uniform random value.

Finally, the algorithm requires a ***termination condition*** to decide when to stop the evolutionary process. As such, a simple yet effective termination condition was used to generate initial results that requires further investigation in future works. That is, the algorithm stops after an absolute threshold of one hundred generations.

4. Experimental evaluation

4.1. The Setup

As mentioned in the previous section the mTSP modeling framework in [1] was enhanced to include the brute-force and GA-based methods using the .NET framework with C#. The hardware specifications used for the tests were as follows: 512 GB of memory, an AMD® Ryzen Threadripper Pro 5955WX processor with 16 cores and 32 threads, operating at 4.0 GHz, and four NVIDIA GA102GL [RTX A5000] graphics cards.

To facilitate a more straightforward comparison between brute-force and GA, points outside the bounding box encompassing all sensor stations were excluded as these points contribute to increased total traveled distances rather than reduced.

A total of 80 representative flight plans, each corresponding to a unique scenario, were developed.

These scenarios varied in the number of stations, which were set at 5, 10, 15, 20, 25, 45, 60, or 100, and incorporated between 1 and 10 drones. Each flight plan was repeated nine additional times, for a total of 10 iterations per scenario, to account for the stochastic nature of the random station locations. Consequently, the study included a total of 80 scenarios x 10 iterations = 800 tests.

4.2. The Results

A total of 800 tests were conducted, each executed twice: once using brute-force and once using the GA. For each test, measurements were recorded for both the brute-force and GA approaches, specifically focusing on the total distance traveled and the time required to identify the optimal starting point.

In order to test the effectiveness and efficiency of the proposed methodology, this work utilizes the following two metrics: the Performance *degree* (Eq. 4) of the GA approach in relation to the ground truth provided by the brute-force method, and the *TimeRatio* (Eq. 5) of the GA approach in relation to the ground truth provided by the brute-force method. Accordingly, *TimeRatio* values close to 0 indicate the superiority of the GA approach, values close to 1 indicate the equivalence of the GA and brute-force approaches, while values greater than 1 show the superiority of the brute-force approach over the GA.

$$Performance = 1 - \left(1 - \frac{\text{Brute Force Distance}}{\text{GA Distance}}\right) \quad (4)$$

$$TimeRatio = \frac{\text{Genetic Algorithm Time}}{\text{Brute Force Time}} \quad (5)$$

Based on the results obtained, as shown in (Fig. 2) and (Fig. 3), the GA achieves accuracy comparable to brute-force while requiring significantly less computational time. The X-axis for both graphs shows how many ground sensor stations are used in each scenario.

More specifically in (Fig. 2) we see that the total traveled distance generated when GA is used to predict a starting point is a near match of that of the brute-force ranging from 100% to 97%.

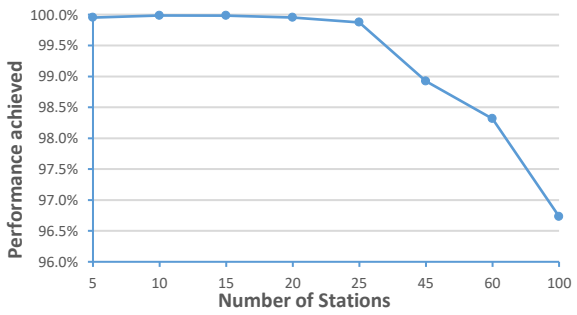


Fig. 2. The extent to which the GA approximated the brute-force's results.

Additionally, as shown in (Fig. 3), the computational cost of the Genetic Algorithm is significantly lower than that of the brute-force method, with *TimeRatio* of GA over brute-force methods being in the range of [0.00004490, 0.00021891] indicating thus the clear superiority of the GA approach, as previously discussed.

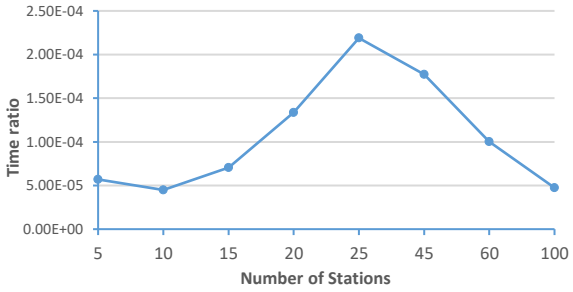


Fig. 3: Time ratio indicating GA's speed advantage over brute-force

In a qualitative approach of the evaluation of our experimentation we have to note that while the brute-force method guarantees the exact optimal solution, its computational cost becomes prohibitive for large-scale problems. In contrast, the Genetic Algorithm:

- Provides a near-optimal solution with accuracy levels exceeding 97%.
- Drastically reduces computation time, achieving results up to 4 orders of magnitude faster than the brute-force.

This balance of accuracy and efficiency makes the GA an ideal choice for large-scale sensor network optimization problems, where real-time or near-real-time decision-making is crucial.

A quick view of the findings is summarized in Table 1.

Table 1. Summary of Findings

Metric	Brute-Force	Genetic Algorithm	Advantage (GA)
Accuracy (Distance)	Exact solution	97% to 100%	Near-optimal
Computation Time	Prohibitively High	Significantly lower	4 orders of magnitude faster

5. Conclusions

Drone path planning constitutes a critical function within the domain of UAV operations, facilitating efficient navigation while simultaneously satisfying mission-specific objectives. Given the diverse

applications of UAVs, path planning exerts a significant influence on operational efficacy. This is achieved through the optimization of flight distance, consequently minimizing resource consumption, including battery life and operational time.

This study has demonstrated the significant role of strategic launch pad positioning in enhancing the efficiency of drone path planning. By utilizing the mTSP algorithm, we show that repositioning the launch pad can lead to substantial reductions in redundant travel distances, ultimately improving overall mission performance. These findings underscore the critical importance of adaptive deployment strategies in the design of energy-aware drone systems.

Future research will focus on fine-tuning the internal parameters of the Genetic Algorithm to enhance its performance. Additionally, exploring alternative methods for predicting the optimal starting point will be pursued to further improve the overall results.

Acknowledgements

Funded by the European Union for the project REMARKABLE (GA101086387). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Executive Agency (REA). Neither the European Union nor the granting authority can be held responsible for them.

References

[1]. G. Gasteratos, I. Karydis, Path planning optimization for multiple drones: Repositioning the starting point, IFIP International Conference on Artificial Intelligence Applications and Innovations, *Springer*, 2024, pp. 211–223.

[2]. M. M. Quamar, B. Al-Ramadan, et al., Advancements and applications of drone-integrated geographic information system technology - a review Remote Sensing, Vol. 15, Issue 20, 2023.

[3]. R. A. Saeed, M. Omri, et al., Optimal path planning for drones based on swarm intelligence algorithm Neural Computing and Applications, Vol. 34, Issue 12, 2022, pp. 10133–10155.

[4]. X. Fan, A path-planning method for UAV swarm under multiple environ-mental threats Drones, Vol. 8, 2024, p. 171.

[5]. T. Xiong, H. Li, et al., A hybrid improved symbiotic organisms search and sine–cosine particle swarm optimization method for drone 3d path planning Drones, Vol. 7, Issue 10, 2023.

[6]. P. Boccadoro, D. Striccoli, et al., An extensive survey on the internet of drones Ad Hoc Networks, Vol. 122, 2021, p. 102600

[7]. J. V. Shirabayashi, L. B. Ruiz, Toward UAV path planning problem optimization considering the internet of drones, *IEEE Access*, Vol. 11, 2023, pp. 136825–136854.

[8]. M. Jones, S. Djahel, et al., Path-planning for unmanned aerial vehicles with environment complexity considerations: a survey, *Acm Computing Surveys*, Vol. 55, 2023, pp. 1–39.

[9]. Y. Wu, K. Low, et al., Swarm-based 4d path planning for drone operations, in *Urban Environments*, *IEEE Transactions on Vehicular Technology*, Vol. 70, 2021, pp. 7464–7479.

[10]. J. Diller, Q. Han, Energy-aware drone path finding with a fixed-trajectory ground vehicle, 2023.

[11]. T. Bektaş, The multiple traveling salesman problem: an overview of formulations and solution procedures, *Omega-international Journal of Management Science*, Vol. 34, 2006, pp. 209–219.

[12]. O. Cheikhrouhou, I. Khoufi, A comprehensive survey on the multi-ple travelling salesman problem: Applications, approaches and taxonomy, *Computer Science Review*, Vol. 40, 2021, p. 100369.

[13]. S. N. Sivanandam, S. N. Deepa, Introduction to Genetic Algorithms. *Springer*, 2008.

[14]. A. Lambora, K. Gupta, et al., Genetic algorithm - a literature review, in *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, Faridabad, India, 2019, pp. 380–384.

[15]. S. Katoch, S. S. Chauhan, et al., A review on genetic algorithm: past, present, and future, *Multimedia tools and applications*, Vol. 80, 2021, pp. 8091–8126.

[16]. C# genetic algorithm library, https://github.com/giac omelli/GeneticSharp.

[17]. D. Whitley, A genetic algorithm tutorial Statistics and Computing, Vol. 4, Issue 2, 1994, pp. 65–85.

[18]. T. Back, Selective pressure in evolutionary algorithms: a characterization of selection mechanisms, in *Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence*, Orlando (FL), USA, Vol. 1, 1994, pp. 57–62.

[19]. A. E. Eiben, J. E. Smith, Introduction to Evolutionary Computing. *Springer*, 2015.