

## Article

# Strategic Launch Pad Positioning: Optimizing Drone Path Planning Through Genetic Algorithms <sup>†</sup>

Gregory Gasteratos <sup>\*,‡</sup>  and Ioannis Karydis <sup>‡</sup> 

Department of Informatics, Ionian University, 49132 Kerkyra, Greece; karydis@ionio.gr

\* Correspondence: ggasteratos@ionio.gr

<sup>†</sup> This article is a revised and expanded version of a paper entitled “Efficient Drone Path Planning through Strategic Launch Pad Positioning”, which was presented at “1st International Conference on Drones and Unmanned Systems (DAUS’ 2025)”, Granada, Spain, 19–21 February 2025.

<sup>‡</sup> These authors contributed equally to this work.

## Abstract

Multi-drone operations face significant efficiency challenges when launch pad locations are predetermined without optimization, leading to suboptimal route configurations and increased travel distances. This research addresses launch pad positioning as a continuous planar location-routing problem (PLRP), developing a genetic algorithm framework integrated with multiple Traveling Salesman Problem (mTSP) solvers to optimize launch pad coordinates within operational areas. The methodology was evaluated through extensive experimentation involving over 17 million test executions across varying problem complexities and compared against brute-force optimization, Particle Swarm Optimization (PSO), and simulated annealing (SA) approaches. The results demonstrate that the genetic algorithm achieves 97–100% solution accuracy relative to exhaustive search methods while reducing computational requirements by four orders of magnitude, requiring an average of 527 iterations compared to 30,000 for PSO and 1000 for SA. Smart initialization strategies and adaptive termination criteria provide additional performance enhancements, reducing computational effort by 94% while maintaining 98.8% solution quality. Statistical validation confirms systematic improvements across all tested scenarios. This research establishes a validated methodological framework for continuous launch pad optimization in UAV operations, providing practical insights for real-world applications where both solution quality and computational efficiency are critical operational factors while acknowledging the simplified energy model limitations that warrant future research into more complex operational dynamics.

**Keywords:** path planning optimization; drone path planning; starting point selection; launch pad positioning; mTSP; genetic algorithms; PLRP; location-routing problem



Academic Editor: Heming Jia

Received: 3 September 2025

Revised: 1 October 2025

Accepted: 10 October 2025

Published: 14 October 2025

**Citation:** Gasteratos, G.; Karydis, I.

Strategic Launch Pad Positioning: Optimizing Drone Path Planning Through Genetic Algorithms.

*Information* **2025**, *16*, 897. <https://doi.org/10.3390/info16100897>

**Copyright:** © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 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 [1]. 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 [2].

As drone technology advances, path planning has become an interdisciplinary research area by incorporating robotics, artificial intelligence (AI), mathematics, and GISs [3,4]. Nonetheless, the main focus still remains on calculating the best routes from start to

destination, taking into account mission-specific factors. From the 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.

### *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 [1,5]. To address this challenge, we establish a computational framework using travel distance minimization as the optimization objective, acknowledging that comprehensive energy optimization would require integration of aspects such as wind dynamics, terrain elevation, battery characteristics, and flight maneuvers as discussed in Section 4.5. This work 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:

- Formulates UAV launch pad positioning as a planar location-routing problem and demonstrates genetic algorithm effectiveness in continuous space optimization for multi-drone operations.
- Provides a comprehensive systematic comparison of a genetic algorithm against both exhaustive search methods and alternative metaheuristics (PSO, SA) for optimizing launch pad location in UAV applications, establishing the GA's superior computational efficiency despite marginally lower accuracy compared to PSO.
- Introduces and evaluates smart initialization and adaptive termination techniques for genetic algorithms in the LRP context, demonstrating modest but consistent improvements in convergence characteristics while maintaining solution quality.
- Provides a validated methodological framework with empirical benchmarks enabling practitioners to select appropriate optimization approaches based on accuracy requirements versus computational constraints.
- Enhances the framework presented in [2] with multiple algorithmic approaches that aid in the prediction of optimal launch pad positioning.
- Establishes computational benchmarks and parameter optimization guidelines for metaheuristic applications in continuous location-routing problems, contributing to the broader UAV optimization literature.

The structure of the paper is as follows: Section 2 reviews the 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. With the increasingly intricate operational landscape of drones, researchers have developed

a range of strategies to address the challenges associated with path planning, including obstacle avoidance, energy efficiency, and environmental adaptability.

### 2.1. Planar Location-Routing Problems in UAV Applications

The foundational work of Drezner and Wesolowsky [6–8] established the theoretical and methodological foundations for location-routing problems in continuous space, introducing trajectory-based optimization methods and formal integration of location and routing considerations that became cornerstones of modern planar location-routing theory. Their pioneering research demonstrated that treating location and routing decisions as integrated optimization problems rather than sequential subproblems yields significantly superior solutions.

The problem addressed in this study falls within the broader framework of planar location-routing problems, which integrate two interrelated optimization tasks: determining the optimal placement of facilities in the continuous plane and designing efficient routes from those facilities to serve geographically dispersed locations. Unlike discrete location-routing problems that restrict facility locations to predetermined candidate sites, a planar LRP allows facilities to be positioned anywhere within a continuous two-dimensional space, creating a nonlinear optimization challenge that captures the true interdependency between facility positioning and route optimization.

In the context of UAV operations, this translates to jointly optimizing launch pad positioning and flight path planning, where the launch pad coordinates (x,y) can be positioned anywhere within the operational area rather than being constrained to existing infrastructure locations. This continuous formulation is particularly relevant for drone applications where temporary or mobile launch platforms can be deployed flexibly based on mission requirements.

The foundational work of Salhi and Rand [9] established that treating location and routing decisions independently can lead to significantly suboptimal solutions, demonstrating the critical importance of joint optimization approaches. Their alternating heuristic method, which iteratively solves Weber-type location subproblems and vehicle routing subproblems, forms the theoretical foundation for many contemporary PLRP solution methodologies. Subsequent research by Nagy and Salhi [10] provided comprehensive surveys of location-routing variants, highlighting that the single-facility continuous PLRP remains one of the most studied configurations, often addressed through Weiszfeld-routing alternation methods or metaheuristic approaches.

### 2.2. UAV Path Planning Optimization Methodologies

One significant area of advancement is the development of algorithms that optimize path planning under various constraints. Fan et al. [11] introduced 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. [12] introduced 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) [13] has been explored to optimize UAV path planning further. Shirabayashi [14] 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.

### *2.3. Environmental Adaptability and Swarm Intelligence*

The consideration of environmental factors is also critical in recent studies. Jones et al. [15] 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. [16] 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.

### *2.4. Green Drone Delivery and Energy-Aware Routing*

Recent research has increasingly focused on minimizing energy consumption rather than simply optimizing distance or time in drone delivery systems. Lu et al. [17] formulated the green drone multi-package delivery routing problem (GDMPDRP), establishing a mixed-integer linear programming model where total energy consumption serves as the primary objective function. Their work demonstrates a critical insight: the shortest delivery path does not necessarily consume the lowest energy as energy consumption is influenced by both distance and payload weight variations throughout the route. They developed two hybrid algorithms—ACWS-SCIP (combining Advanced Clarke and Wright Saving with the SCIP exact solver) and ACWS-ICSA (integrating an Improved Crow Search Algorithm)—achieving near-optimal solutions while reducing computational time by over 95% compared to traditional genetic algorithms and particle swarm optimization.

While Lu et al.'s work provides a robust framework for energy-aware routing optimization with explicit modeling of payload-dependent battery drain, their problem formulation assumes fixed launch pad locations as given parameters. This constraint represents a significant limitation in real-world deployment scenarios where temporary or mobile launch platforms can be positioned flexibly. Our research addresses this gap by treating launch pad coordinates as continuous decision variables rather than fixed parameters, thereby extending the energy-aware optimization framework to encompass both facility location and vehicle routing decisions simultaneously, capturing the full flexibility available in UAV deployment while maintaining computational tractability through metaheuristic approaches.

### *2.5. Energy-Aware Planning and Launch Pad Optimization*

In addition, energy efficiency remains a pivotal concern in drone path planning. Diller's [18] 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 [2] proposed a path planning optimization technique whereby the starting point distances can be reduced, leading to better energy management and operational efficiency. Their research shows 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.

## 2.6. Metaheuristic Approaches to PLRP

Recent comprehensive surveys of truck–drone routing problems reveal that metaheuristic algorithms dominate the solution landscape, accounting for 43% of all solution methodologies [19]. Within metaheuristics, neighborhood search methods are most prevalent (51%), followed by evolutionary approaches (23%) and swarm intelligence methods (17%). Genetic algorithms represent the dominant evolutionary technique, comprising 71% of evolutionary approaches employed [19]. This distribution pattern closely aligns with the planar LRP domain, where genetic algorithms and variable neighborhood search have proven most effective for continuous location-routing optimization.

The success of genetic algorithms in truck–drone applications stems from their ability to handle the dual optimization challenge of vehicle routing and facility positioning through chromosome encodings that represent both route sequences and continuous coordinates [19]. This parallel directly supports our application of genetic algorithms to UAV launch pad positioning, where similar dual optimization challenges exist.

Variable Neighborhood Search (VNS) approaches, as explored by Mladenović and Brimberg [20], offer systematic mechanisms for escaping local optima in continuous location-allocation problems. These methodologies systematically explore neighborhoods of increasing complexity, providing robust frameworks that can be adapted to UAV deployment scenarios where both location and routing decisions must be optimized simultaneously.

A recent survey by Mara et al. [21] highlights that continuous (planar) variants of the location-routing problem (PLRP) remain a niche area (only 2%) within the broader LRP research landscape. These models are typically addressed using metaheuristic or hybrid heuristic frameworks as exact solution methods remain limited to small-scale instances due to the mixed-integer nonlinear programming complexity introduced by Euclidean distance calculations in continuous space.

## 2.7. Research Gap and Contribution Positioning

While existing research has made significant advances in both discrete UAV path planning and general planar location-routing problems, several critical gaps remain unaddressed. Most UAV path planning research, including recent energy-aware formulations [17], assumes fixed launch locations, while planar LRP studies typically focus on generic vehicle routing without considering the unique energy consumption characteristics and operational constraints of unmanned aerial systems.

Lu et al. [17] demonstrate that energy-optimal drone routes differ systematically from distance-optimal routes due to payload–weight interactions, establishing the importance of energy-aware objectives in drone delivery optimization. However, their formulation treats the launch pad location as an exogenous parameter, limiting the solution space to routing decisions alone. In practice, launch pad positioning significantly affects both the initial/final flight segments and the overall route structure determined by the mTSP algorithm as suboptimal launch pad placement can force longer routes or inefficient customer assignments even when routing is optimized.

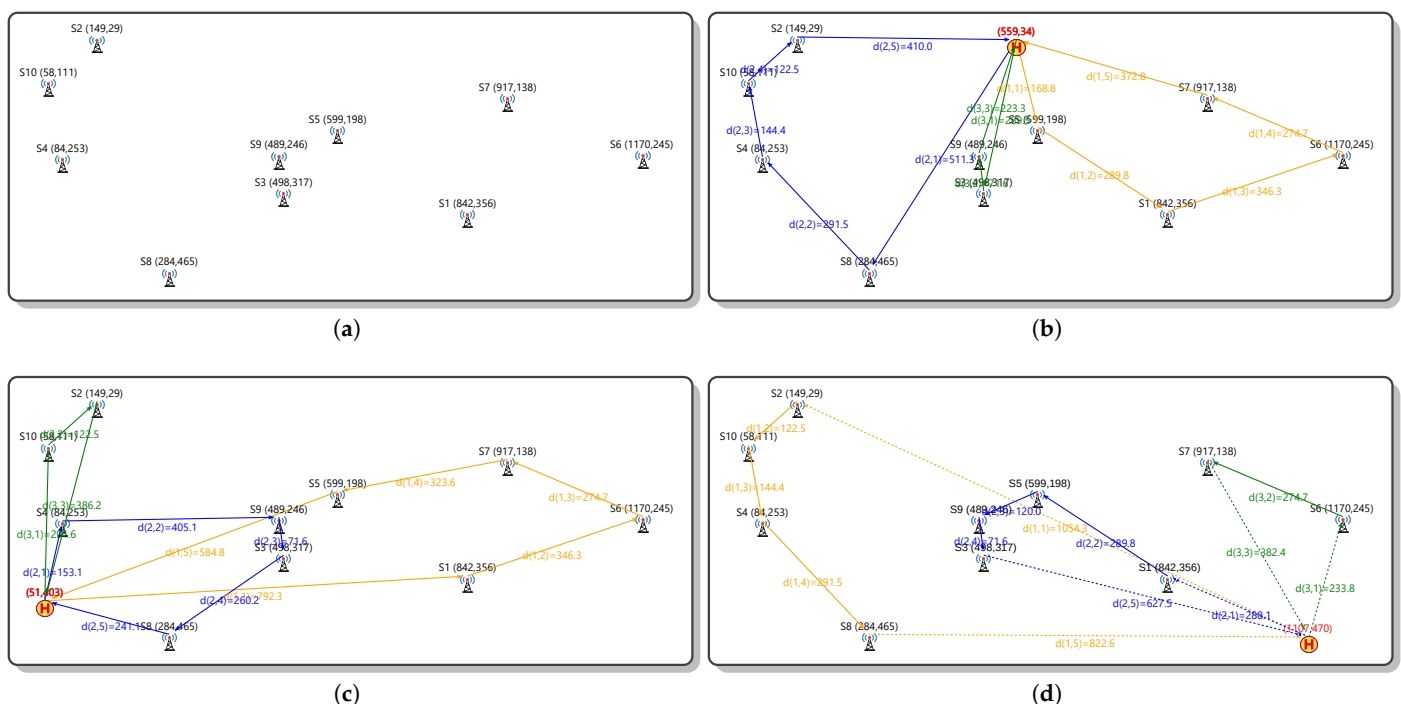
Our work bridges this gap by formulating UAV launch pad positioning as a planar location-routing problem and demonstrating that genetic algorithm-based approaches can efficiently solve this continuous optimization challenge. By allowing launch pad coordinates to vary continuously within the operational area rather than being restricted to predetermined locations, our methodology captures the full flexibility available in UAV deployment scenarios while maintaining computational tractability through metaheuristic

optimization techniques. This represents a natural extension of the energy-aware routing framework established by Lu et al., addressing the coupled optimization of both facility location and route assignment for multi-drone operations.

### 3. Proposed Method

The impact of launch pad repositioning as described by Gasteratos and Karydis in their work [2] goes beyond improving the initial and final segments of each drone's route. The mTSP [22,23] 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 Figure 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.



**Figure 1.** Changing the launch pad position. (a) Ten sensor stations (S1.. S10) in a field. (b) Applying mTSP on (a) using an arbitrary launch pad and 3 routes. The distances  $d(i,j)$  represent the sequential path segments for each drone  $i$ , where  $d(i,1)$  is the distance from the launch pad to the first assigned station,  $d(i,j)$  for  $j > 1$  represents the distance between consecutive stations, and the final segment shows the return distance to the launch pad. (c) Scenario (b) is repeated using a different launch pad, showing how repositioning affects route distances and assignments. (d) Similar repetition of (b) but



with a completely different launch pad location, further demonstrating the impact of launch pad positioning on total traveled distances. Different colored lines denote planned drone trajectories.

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 operations. In light of these considerations, our research endeavors to explore methodologies for determining the most advantageous launch pad location for a specified flight plan.

While the depot location within VRP formulations is well studied as discussed in Section 2, many such methods either (a) restrict depot candidates to a small discrete set, (b) ignore the combinatorial routing subproblem (using a linear-assignment surrogate), or (c) assume simplistic energy models (e.g., Euclidean distance without UAV dynamics). In contrast, our algorithm (i) searches over every point in the convex hull of sensor stations (continuous domain), (ii) solves an exact or near-exact mTSP (in the lower level) to capture true combinatorial routing costs, and (iii) introduces an adaptive termination rule for on-mission feasibility. To our knowledge, no prior work addresses all three simultaneously for UAVs.

### 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 Figure 1a, with dimensions of 1496 units in length and 571 units in width. The brute-force approach would require running the mTSP algorithm  $1496 \times 571 = 854,216$  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 (2) and (3) as defined by Cheikhrouhou and Khoufi in their work [23], the optimal starting point can be defined as the one with the minimum total traveled distance after applying mTSP in the form of

$$\text{Optimal Starting Point} = \arg \min_j (\text{TotalDistance}_j) \quad (1)$$

whereby the total distance for  $x$  drones in a flight plan for a particular starting point  $j$  is defined as

$$\text{TotalDistance}_j = \sum_{i=1}^x D(\text{Route}_{U_i})_j \quad (2)$$

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  $S_{i_1}, S_{i_2}, \dots, S_{i_n}$  sequentially in the given order, and then returning to  $H$ .

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

### 3.2. 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.

#### 3.2.1. Rationale for Genetic Algorithm Selection

We chose genetic algorithms as our optimization approach for several compelling reasons that distinguish them from other metaheuristic alternatives such as PSO, Ant Colony Optimization (ACO), and SA [19]. The selection of the GA represents a strategic decision based on both theoretical considerations and practical implementation advantages.

The primary motivation for selecting the GA lies in its natural modularity with existing mTSP fitness evaluation frameworks. Genetic algorithms inherently allow us to encode the launch pad coordinates as real-valued genes within a chromosome structure, enabling direct integration with our established mTSP solver for fitness computation. This approach maintains clean separation between the continuous location optimization and discrete routing evaluation, where each candidate chromosome representing launch pad coordinates can be directly passed to the mTSP algorithm to determine total travel distance. In contrast, alternative metaheuristics such as PSO [24] or Differential Evolution [25] would necessitate additional translation mechanisms to map continuous particle positions into discrete route assignments, potentially requiring ranking heuristics or nearest-neighbor approaches that introduce computational overhead and approximation errors.

The literature precedence further supports our choice of genetic algorithms for integrated location-routing problems. Comprehensive studies by Kang et al. [26] in their exact algorithm for heterogeneous drone-truck routing problems have demonstrated the effectiveness of GA-mTSP hybrid approaches in similar combinatorial optimization contexts. Additionally, Luo et al.'s [27] survey of truck-drone routing problems and Fangs et al.'s [28] recent advances in multi-traveling salesman algorithms for UAV swarms have consistently validated the performance of genetic algorithm frameworks when coupled with mTSP solvers. These established methodologies provide a proven template that facilitates direct comparison against brute-force ground truth while ensuring reproducible results.

Genetic algorithms offer an inherent balance between exploration and exploitation that proves particularly valuable for combinatorial problems like mTSP coupled with continuous location variables. The crossover operations enable exploration of new solution regions by combining beneficial traits from different parent solutions, while mutation operators maintain population diversity and prevent premature convergence. This balanced approach has demonstrated consistent success in handling the dual nature of our optimization problem, where both continuous coordinate optimization and discrete routing assignments must be simultaneously optimized.



### 3.2.2. The Approach

To implement this approach, we enhanced the mTSP modeling framework described in [2] 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 [29], which facilitates the development of applications using genetic algorithms.

A genetic algorithm requires several key features to function effectively [30]. First, it needs a method to represent potential solutions to the problem (the chromosome in GA terms). We implemented the 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. The population initialization in our implementation employs an adaptive sizing strategy with bounds set between 50 and 100 chromosomes. This approach, supported by the GeneticSharp library, begins with a minimum population of 50 individuals and can dynamically expand up to 100 based on diversity metrics and convergence behavior. The adaptive mechanism balances computational efficiency (smaller populations converge faster) against solution space exploration (larger populations maintain diversity).

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 [31] was used, which 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.

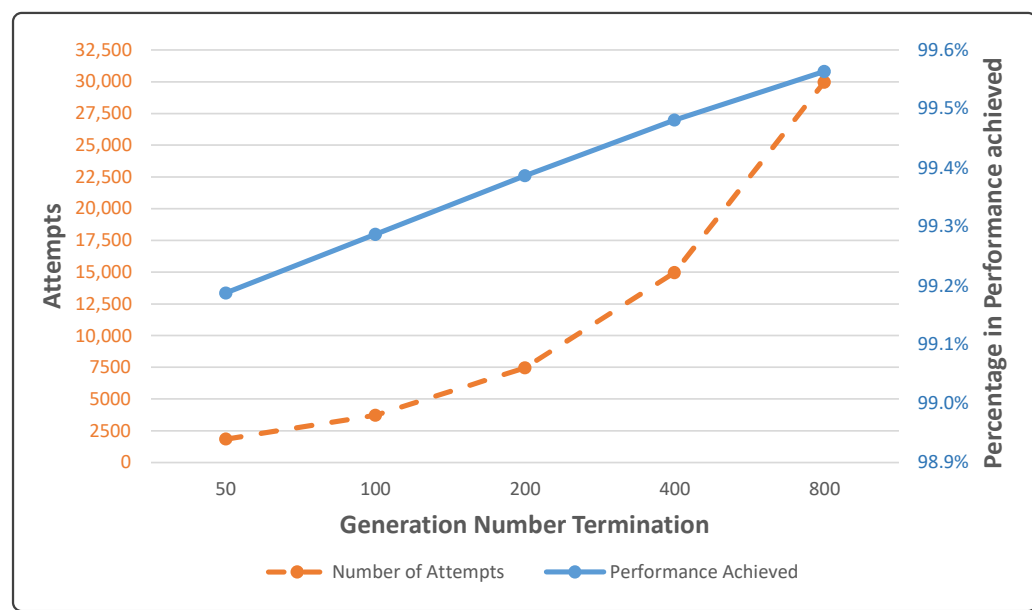
To create new solutions, the algorithm relies on crossover and mutation operators [32]. Crossover combines parts of two parent solutions to produce offspring, facilitating the exchange of beneficial traits. Uniform crossover, which utilizes a fixed mixing ratio between two parents, was used in this implementation. 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, which replaces the value of the chosen gene with a uniform random value, was used.

Finally, the algorithm requires a termination condition to decide when to stop the evolutionary process. Setting this parameter involves balancing computational efficiency against solution quality. When the algorithm terminates prematurely, potential improvements remain unexplored; conversely, excessive iterations often yield diminishing returns while consuming valuable computational resources.

To identify an optimal termination point for the number of generations, we conducted sensitivity testing (40,000 tests) across multiple threshold values (50, 100, 200, 400 and 800). Analysis of the resulting data (Figure 2) revealed a consistent pattern: higher generation counts produced incremental performance improvements, with accuracy metrics rising from 99.2% at 50 generations to 99.6% at 800 generations. However, the computational cost increased substantially and non-linearly, with attempt counts rising from 1843 at 50 generations to 29,963 at 800 generations.

The relationship between performance gain and computational expenditure demonstrated clear diminishing returns beyond 100 generations. At this threshold, the algorithm

achieved 99.3% accuracy while requiring only 3713 attempts, representing an optimal balance between solution quality and computational efficiency. Based on these findings, we implemented a straightforward termination condition that stops the evolutionary process after completing 100 generations, regardless of other convergence indicators.



**Figure 2.** Generation number–termination condition–sensitivity analysis (40,000 tests). number of attempts (left axis, dashed line); performance achieved (right axis, solid line).

The implementation is as follows: Algorithm 1 shows the setup that is used within the *FindOptimalStartingPoint* method to initialize the GeneticSharp library and then get the result by calling the *GA.Start()* method in line 9.

---

#### Algorithm 1 FindOptimalStartingPoint

---

**Require:** List of coordinates  $P$ , Number of routes  $R$

**Ensure:** Optimal starting coordinate, total distance

```

1: fitness_function  $\leftarrow$  new MtspFitness( $P$ ,  $R$ )
2: selection_proccess  $\leftarrow$  EliteSelection()
3: crossover_operator  $\leftarrow$  UniformCrossover()
4: mutation_operator  $\leftarrow$  UniformMutation(true)
5: adam_chromosome  $\leftarrow$  new MtspChromosome()
6: initial_population  $\leftarrow$  new Population(adam_chromosome)
7: GA  $\leftarrow$  new GeneticAlgorithm(initial_population, fitness_function, selection_proccess,
   crossover_operator, mutation_operator)
8: GA.Termination  $\leftarrow$  GenerationNumberTermination(100)
9: GA.Start()
10: bestChromosome  $\leftarrow$  GA.BestChromosome
11:  $x \leftarrow$  bestChromosome.GetGene(0).Value
12:  $y \leftarrow$  bestChromosome.GetGene(1).Value
13: return MtspGeneticResult:
    • BestPoint  $\leftarrow$  ( $x$ ,  $y$ )
    • BestDistance  $\leftarrow$   $-$ bestChromosome.Fitness.Value

```

---

Algorithm 2 shows the evolutionary process of obtaining the optimal starting point. The algorithm iteratively evolves a population through selection, crossover, and mutation, evaluating fitness at each step to identify the best solution.

A summary of the GA's parameter settings is provided in Table 1.

Figure 3 provides an overview of the entire process. The outer loop generates and refines potential launch pad locations through a genetic algorithm (GA). For each candidate location, the inner loop determines the most efficient drone routes by solving the mTSP, calculating the total travel distance. The fitness of each candidate is evaluated based on this total distance, with the goal of minimizing it. Finally, the algorithm outputs the optimal launch pad location along with its corresponding minimum total distance.

---

**Algorithm 2** Genetic algorithm for optimal starting point in MTSP

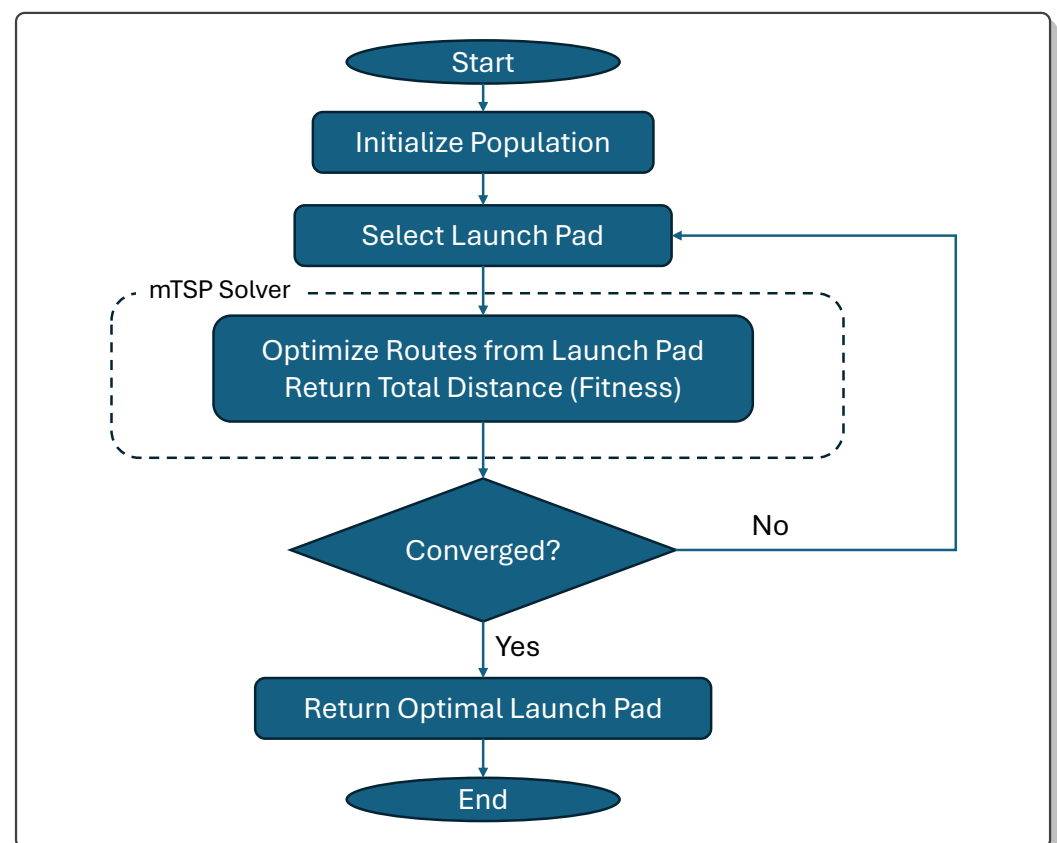
---

```

1: Initialize population using CustomPopulation.CreateInitialGeneration()
2: for all chromosome in population do
3:   Evaluate fitness using MtspFitness.Evaluate()
4: end for
5: while termination condition not met do
6:   Select parents using EliteSelection
7:   Perform crossover using UniformCrossover
8:   Apply mutation using UniformMutation
9:   for all offspring do
10:    Evaluate fitness using MtspFitness.Evaluate()
11:   end for
12:   Replace population with new generation
13:   Update best chromosome
14: end while
15: return Best starting point coordinate and best fitness (distance)

```

---



**Figure 3.** A high-level view of the complete operational framework.

**Table 1.** Genetic algorithm configuration for mTSP.

| Parameter          | Value/Description                                                                       |
|--------------------|-----------------------------------------------------------------------------------------|
| Fitness Function   | mTSP total distance                                                                     |
| Selection Process  | Elite selection                                                                         |
| Crossover Operator | Uniform crossover<br>Probability 0.75 per chromosome pair (default)                     |
| Mutation Operator  | Mix probability 0.5 per gene (default)<br>Uniform mutation<br>Probability 0.1 (default) |
| Initial Population | 50 (minimum) to 100 (maximum, adaptive)                                                 |
| Generations        | 100                                                                                     |

## 4. Experimental Evaluation

### 4.1. The Setup

As mentioned in the previous section, the mTSP modeling framework in [2] 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 (AMD, Santa Clara, CA, USA) processor with 16 cores and 32 threads operating at 4.0 GHz, and four NVIDIA GA102GL [RTX A5000] (NVIDIA, Santa Clara, CA, USA) graphics cards.

To facilitate a more straightforward comparison between brute-force and the 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 9 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  $80 \text{ scenarios} \times 10 \text{ iterations} = 800 \text{ tests}$ .

### 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 (Equation (4)) of the GA approach in relation to the ground truth provided by the brute-force method and the TimeRatio (Equation (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, and 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.

It is important to clarify what might initially appear counterintuitive in the equations. Since we are measuring computational time, superior performance corresponds to lower values rather than higher ones. When algorithm A requires less processing time than algorithm B, algorithm A demonstrates better efficiency. This relationship explains why TimeRatio values approaching the zero signal GA's advantage: the genetic algorithm completes its calculations faster than the exhaustive search method. Conversely, when TimeRatio exceeds one, the brute-force technique proves more time-efficient than the

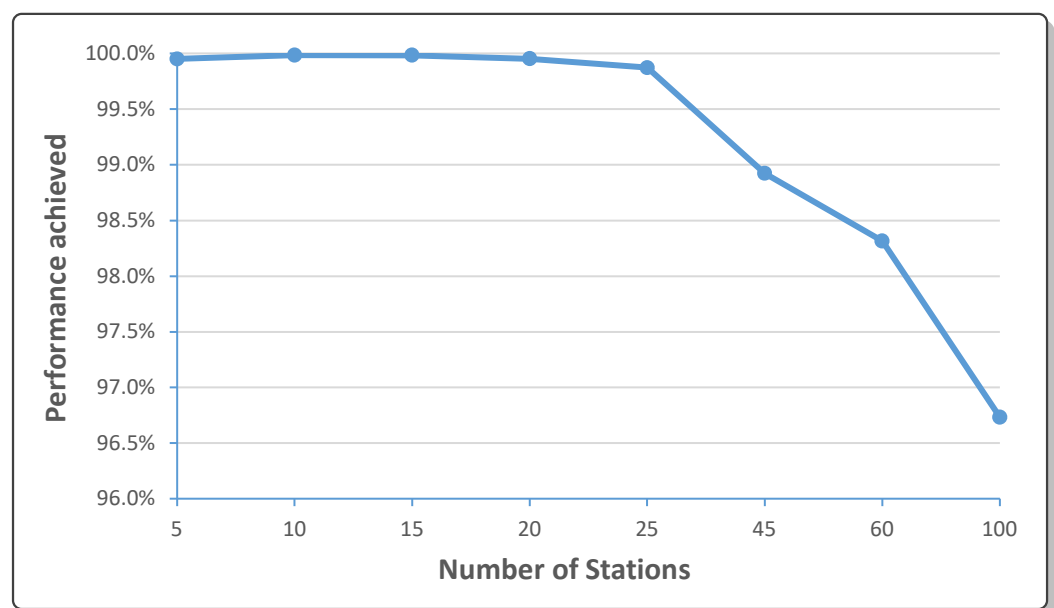
genetic approach. The same principle applies to performance metrics, where we measure distances and superior results correspond to smaller numerical values.

$$Performance = \frac{\text{Brute-Force Distance}}{\text{Genetic Algorithm Distance}} \quad (4)$$

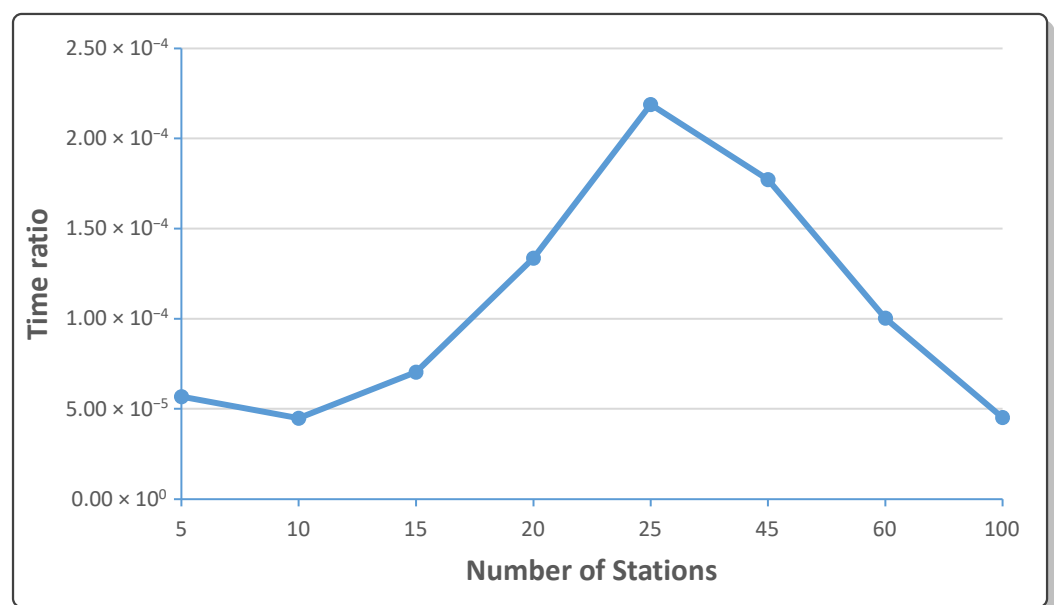
$$Time\ Ratio = \frac{\text{Brute-Force Time}}{\text{Genetic Algorithm Time}} \quad (5)$$

Based on the results obtained, as shown in Figures 4 and 5, 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 Figure 4 we see that the total traveled distance generated when the GA is used to predict a starting point is a near match of that of brute-force, ranging from 97% to 100%.



**Figure 4.** The extent to which the GA approximated the brute-force results (800 tests).



**Figure 5.** Time ratio indicating GA's speed advantage over brute-force (800 tests).

Additionally, as shown in Figure 5, the computational cost (consumed time) of the genetic algorithm is significantly lower than that of the brute-force method, with TimeRatio of the GA over brute-force methods being in the range of [0.00004490, 0.00021891] thus indicating the clear superiority of the GA approach, as previously discussed.

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 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 2.

**Table 2.** 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 |

Finally, the process of optimizing launch pad positioning and route assignment using the GA finds strong methodological support in the recent literature on truck–drone coordination. In their extensive review, Dang et al. [19] identify metaheuristic algorithms—including the GA, ACO [33], PSO, and SA [34]—as the predominant approaches for solving real-world variants of vehicle routing problems under operational constraints such as time windows, battery limits, and synchronization requirements. These methods are particularly well-suited for handling the mixed-integer nonlinear nature of location-routing problems in continuous space. The findings of our study, which demonstrate the GA’s ability to approach optimal solutions with substantial reductions in computational time, echo the broader consensus on the effectiveness of metaheuristics in complex, hybrid routing environments. This reinforces the relevance of our contribution to the UAV and logistics optimization communities.

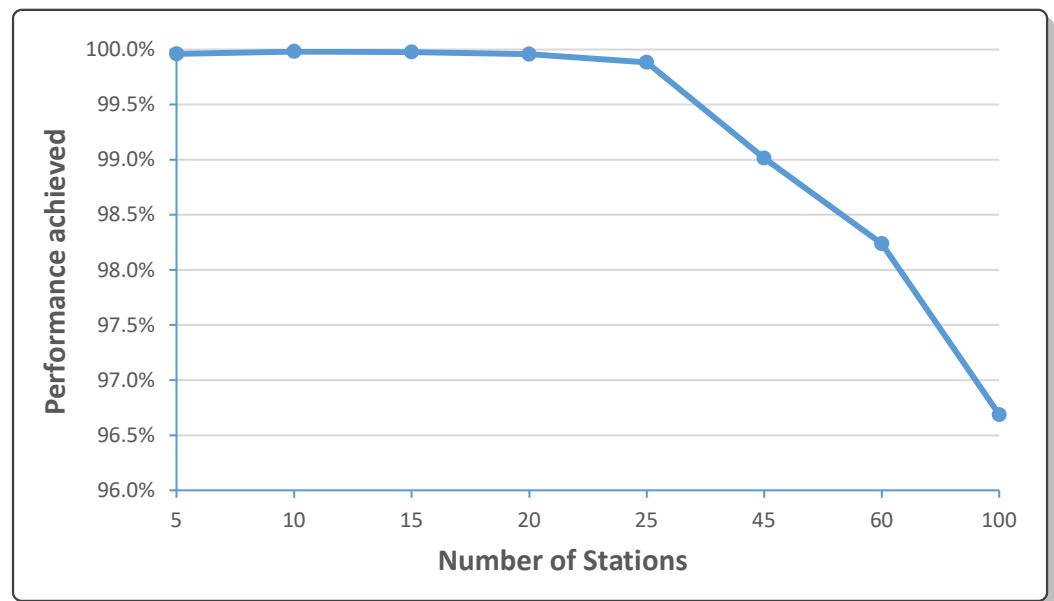
#### 4.3. Solidifying the Results

To further validate our findings and address potential concerns about the reliability of our initial experiment, we conducted an extended analysis that accounted for the inherent stochastic nature of genetic algorithms. Building on our work in [5], we expanded our testing methodology by executing each of the 800 tests an additional 10 times using the genetic algorithm approach, resulting in a comprehensive dataset of 8000 total executions.

By calculating the average performance across these multiple iterations, we were able to minimize the impact of random variations inherent in GA operations and obtain more statistically robust results. This approach enables us to present findings with increased confidence intervals and reliability metrics.

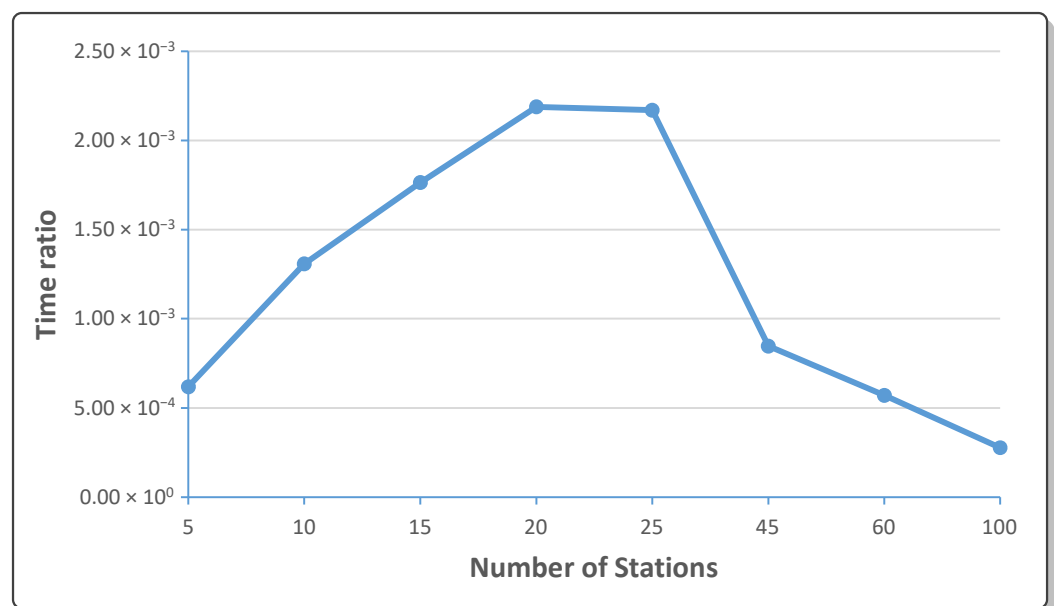
The extended results, consistent with our preliminary findings, confirm the effectiveness of the GA-based approach in identifying near-optimal starting points with significantly reduced computational requirements. As illustrated in Figure 6, the performance consistency of the GA approach demonstrates remarkable stability across multiple iterations, with average accuracy values consistently ranging between 96.7% and 100% when compared to the ground truth established by the brute-force method.





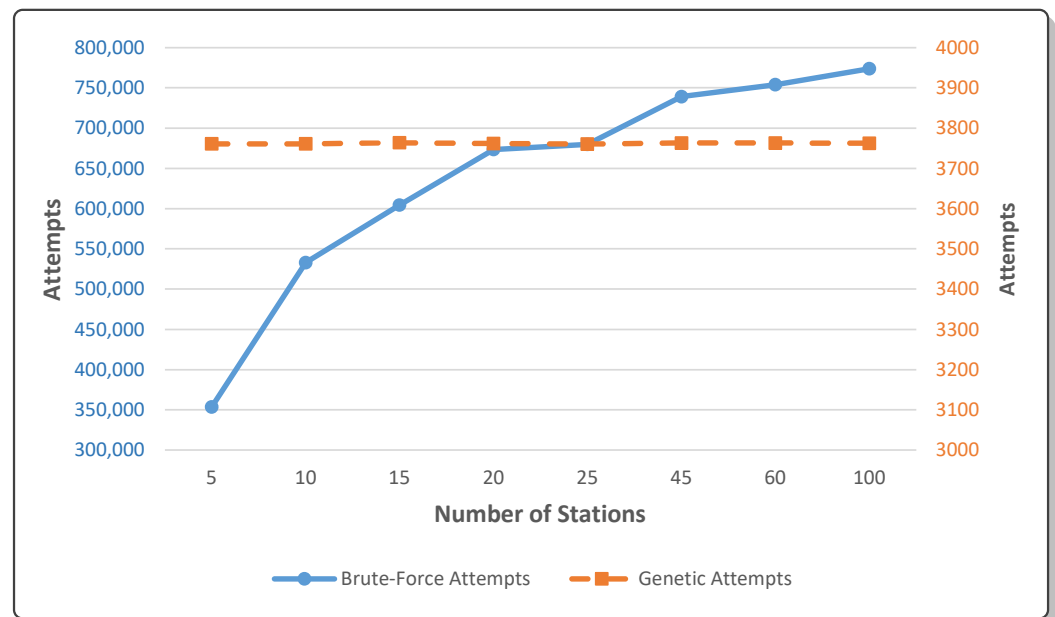
**Figure 6.** The extent to which the GA approximated the brute-force results (8000 tests).

Furthermore, Figure 7 presents the average computational time comparison between both approaches, highlighting the substantial efficiency advantage maintained by the GA method even when accounting for multiple executions. This temporal advantage remains consistently within the previously identified range, confirming the GA's superiority in terms of computational efficiency.



**Figure 7.** Time ratio indicating GA's speed advantage over brute-force (8000 tests).

A particularly illuminating metric is presented in Figure 8, which depicts the average number of attempts required by each method to converge on an optimal or near-optimal solution. While the brute-force approach necessitates a comprehensive evaluation of all potential starting points within the bounding box, the GA achieves comparable results with dramatically fewer attempts—requiring only a fraction of the computational effort expended by the exhaustive search method.



**Figure 8.** Average number of attempts for brute-force and GA (8000 tests).

Finally, to establish the statistical significance of our findings, we employed the Wilcoxon Signed-Rank Test [35] to compare the total distances produced by the brute-force method against those generated by the GA across all test cases. This non-parametric statistical hypothesis test was selected due to its robustness in comparing paired samples without assuming normal distribution. The extremely small  $p$ -value ( $3.86 \times 10^{-14} < 0.05$ ) indicates that the differences between the brute-force distance and GA distance are statistically significant. However, it is important to note that although these differences are statistically significant (i.e., not due to random chance), they are small in absolute magnitude, as evidenced by the GA achieving 97–100% accuracy relative to the brute-force method. This validates that the GA consistently produces solutions that, though systematically different from the optimal, remain of high practical quality and require only a fraction of the computational resources.

Across numerous test runs, we observed steady performance that confirms the GA-based approach's dependability. This consistency validates its practical application in operational scenarios where processing power and timing limitations must be carefully managed.

The expanded analysis not only strengthens our initial findings on the GA's effectiveness but also provides a more solid basis for applying this method in real-world drone deployment scenarios.

#### 4.4. Taking It Further

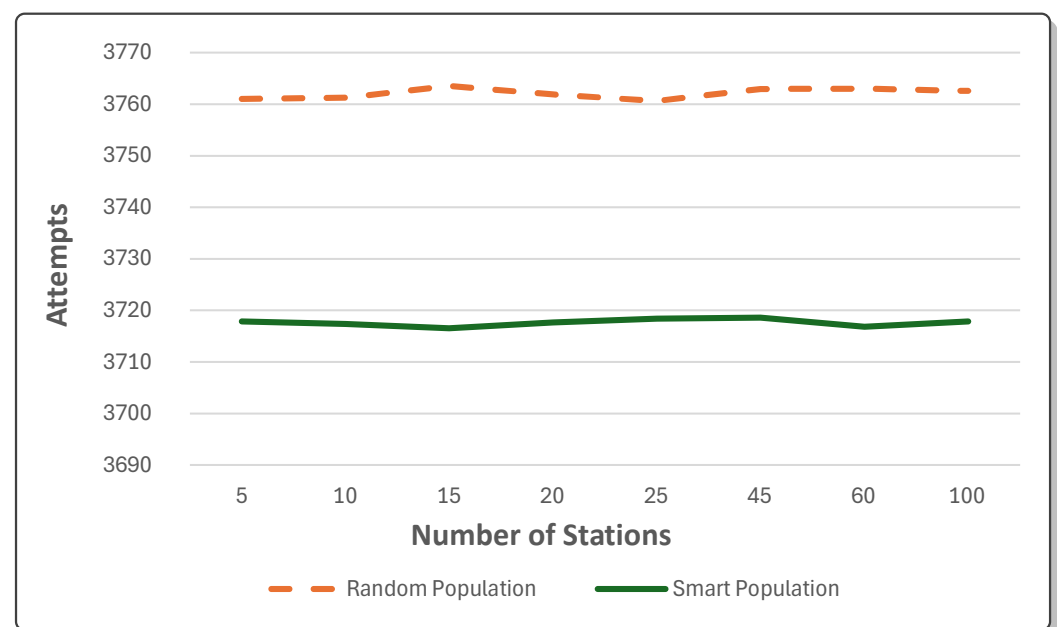
##### 4.4.1. Smart Initial Population

The initial population generation in genetic algorithms constitutes a pivotal stage that can significantly influence both convergence speed and solution quality. In our current implementation, the initial population is created by using randomly generated (X,Y) coordinates. Although this ensures diversity, relying solely on randomness in early generations can limit convergence efficiency. To address this, we incorporate problem-specific knowledge to seed the initial population more intelligently. For example, using geometrically significant positions—such as the Area Center, All Stations Centroid [36], All Stations Geometric Median [37], Bounding Box Center, Convex Hull [38] Centroid, and Convex Hull Geometric Median, we can offer a head start towards optimal spatial configurations. Furthermore, structured spatial sampling techniques could improve population diversity while concentrating on high-value regions. For example, a spiral pattern starting from the

center of the area and expanding outward toward the operational boundary could provide systematic coverage with a gradually decreasing density of points.

Similarly, the selection of the Adam Chromosome—the prototype individual used to seed the initial population—presents another opportunity for optimization. Rather than random assignment, utilizing a theoretically favorable position such as the Geometric Median of the Convex Hull could accelerate convergence by providing the evolutionary process with an advantageous starting point.

These refinements essentially constitute a “warm start” approach for the GA, concentrating computational resources on the most promising regions of the solution space from the outset (smart initial population). By incorporating domain knowledge into the initialization phase, our supplementary experiments demonstrate that both convergence speed and solution quality can be further enhanced, particularly for complex spatial distributions or when operating under stringent computational constraints. The data collected from these additional tests (8000) indicate acceleration in convergence (see Figure 9) when compared to purely random initialization strategies while maintaining comparable accuracy levels (see Figure 10).

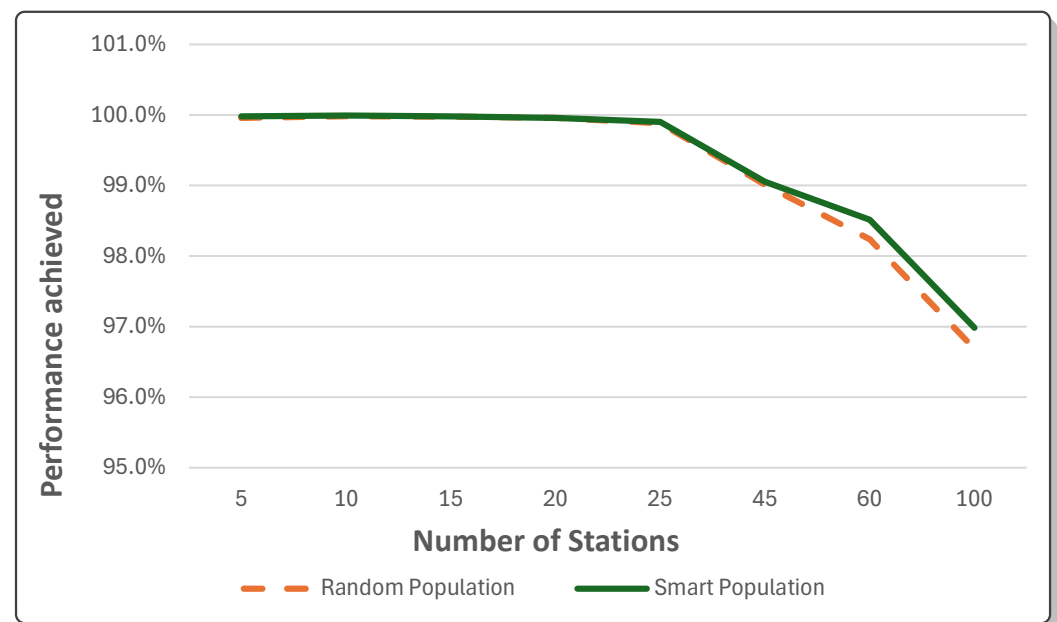


**Figure 9.** Comparing the number of attempts for convergence when using random initial population against smart initial population (8000 tests).

While our smart initialization strategy demonstrates modest improvements (approximately 2% reduction in convergence attempts), several limitations constrain its effectiveness. Our approach assumes relatively uniform station distributions across the operational area. In scenarios with heavily clustered station configurations, geometrically significant positions like centroids may fall within dense clusters rather than strategically advantageous locations, potentially biasing the genetic algorithm toward suboptimal regions. The current implementation uses fixed geometric positions (area center, convex hull centroid, etc.) without considering the underlying spatial structure of station distributions. This approach may not capture the optimal diversity needed for complex spatial configurations.

For heavily clustered station distributions, we propose a two-stage approach: first, apply clustering algorithms such as k-means [39] or DBSCAN [40] to identify station groups, then seed initial chromosomes at cluster centroids and boundaries to ensure coverage of both dense and sparse regions. Different initialization strategies may be optimal for varying problem sizes. Small-scale problems might benefit from geometric centroids,

while large-scale deployments could require hierarchical seeding approaches that consider multiple spatial scales. Combining deterministic geometric positions with strategic random sampling in high-potential regions could balance exploration with informed initialization.



**Figure 10.** Comparing the performance achieved by the GA when using random initial population against smart initial population (8000 tests).

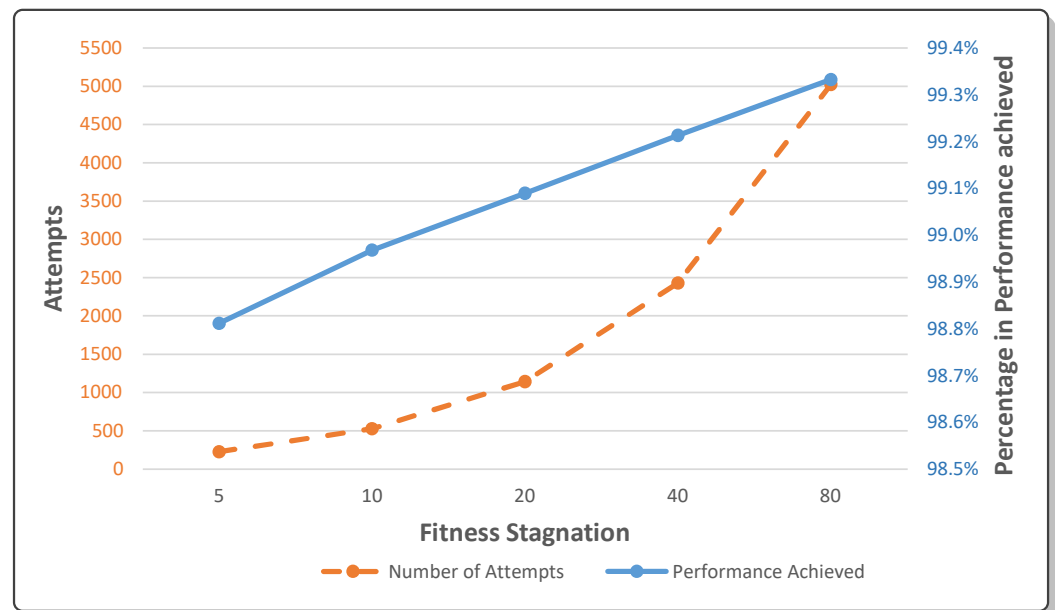
These enhancements represent natural extensions of the current methodology that could yield more substantial improvements in convergence characteristics while maintaining the theoretical foundation established in our current work. The modest gains observed with basic geometric seeding suggest considerable potential for more sophisticated initialization strategies tailored to specific problem characteristics and spatial distributions.

#### 4.4.2. Striking the Right Balance

As already mentioned, there exists a critical relationship between solution quality (performance gain) and computational expenditure (time required to complete the operation). Figure 2 illustrates that higher quality solutions can be achieved by increasing the number of generations, which consequently increases the number of attempts and completion time. However, this approach introduces inefficiencies: with excessively high generation counts (exceeding 300), the algorithm often continues running long after convergence, wasting computational resources; conversely, with insufficient generations (fewer than 50), the algorithm terminates prematurely, sacrificing potential quality improvements.

To address these limitations, we investigated an adaptive termination criterion (40,000 tests) that stops the algorithm when no fitness improvement occurs over a specified number of consecutive generations. This approach aims to dynamically balance quality and efficiency by responding to the actual convergence behavior rather than relying on predetermined thresholds. Our experimental results, presented in Figure 11, demonstrate the effectiveness of this adaptive approach across various stagnation thresholds.

The data reveals several significant patterns. First, even with a minimal stagnation threshold of 5 generations, the algorithm achieves 98.8% solution quality (performance achieved) while requiring only 227 attempts—dramatically fewer than the 3713 attempts needed when using a fixed threshold of 100 generations. This represents approximately a 94% reduction in computational effort while sacrificing less than 0.5% in solution quality.



**Figure 11.** Impact of fitness stagnation-based termination criteria on solution quality and computational effort (40,000 tests). Number of attempts (left axis, dashed line); performance achieved (right axis, solid line).

As the stagnation threshold increases, we observe incrementally higher solution quality accompanied by proportionally larger computational demands. A threshold of 80 generations yields marginally superior results (99.3% quality) but requires 22 ( $5026 \div 227$ ) times more attempts than a threshold of 5 generations. This result shows how additional computational work brings smaller and smaller improvements, a common trend in many optimization problems.

The relationship between the stagnation threshold and computational effort appears nearly linear, while quality improvements follow a logarithmic pattern—each doubling of computational effort yields progressively smaller quality gains. This characteristic makes the stagnation threshold an excellent tuning parameter for practitioners to adjust based on their specific requirements for solution quality versus computational constraints.

For time-sensitive applications where near-optimal solutions suffice, lower stagnation thresholds (5–10) offer exceptional efficiency. Conversely, applications demanding maximum precision might justify higher thresholds (40–80), particularly when computational resources permit extended processing time. These findings underscore the value of adaptive termination criteria in evolutionary algorithms, particularly for applications like drone deployment optimization where both solution quality and computational efficiency represent critical operational factors.

#### 4.4.3. Mitigating Local Minima

To address concerns regarding local minima and premature convergence, our implementation incorporates several mechanisms designed to enhance global search capabilities. We employ uniform crossover with a mixing probability of 0.5, which promotes thorough exploration of the solution space by ensuring balanced genetic material exchange between parent chromosomes. The adaptive termination criterion monitors convergence progress and terminates when no improvement occurs over five consecutive generations, allowing the algorithm to escape potential plateaus while preventing excessive computational expenditure on converged solutions.

Furthermore, our smart initial population initialization strategy draws chromosome seeds from geometrically significant positions, including area centroids, convex hull medi-

ans, and bounding box centers. This approach distributes candidate launch pad positions more uniformly across high-value regions rather than relying solely on random initialization, thereby improving the likelihood of discovering globally optimal solutions while reducing the risk of clustering around local optima.

#### 4.4.4. Comparative Analysis with Alternative Metaheuristics

Although genetic algorithms were selected based on solid theoretical foundations and their proven effectiveness in location-routing applications, a thorough evaluation requires examining alternative metaheuristic approaches. We therefore implemented and evaluated Particle Swarm Optimization and simulated annealing using identical experimental conditions to those applied in our genetic algorithm studies.

#### Parameter Configuration and Sensitivity Analysis

Before conducting comparative studies, we performed comprehensive sensitivity analyses to identify optimal parameter settings for each algorithm. For the PSO implementation, we systematically examined different parameter combinations according to the specifications outlined in Table 3.

**Table 3.** Parameter values in PSO sensitivity analysis.

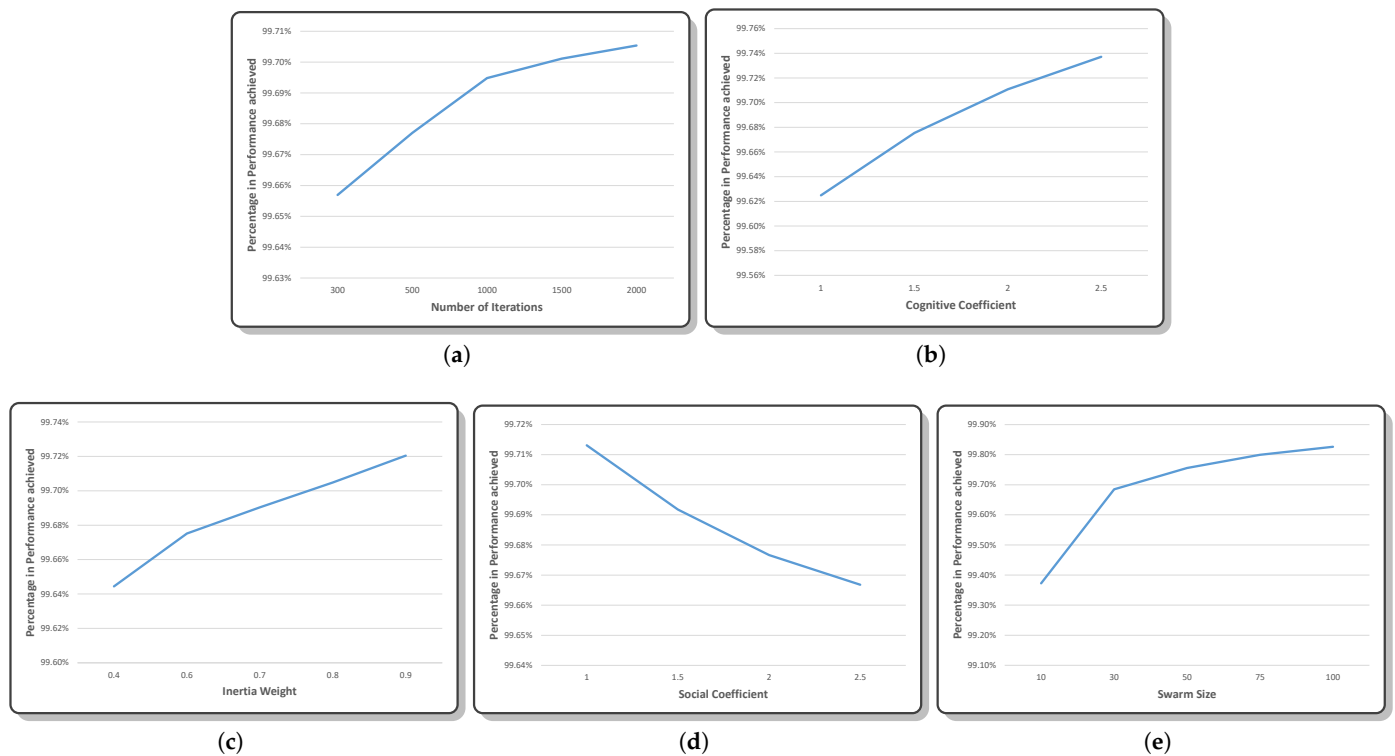
| Parameter             | Values                     | Description                                                                                                                                                                                                    |
|-----------------------|----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Social Coefficient    | 1, 1.5, 2, 2.5             | This parameter controls how much the particle is influenced by the best solution found by its neighbors in the swarm. Higher values make particles more likely to move toward the swarm's best known position. |
| Cognitive Coefficient | 1, 1.5, 2, 2.5             | Determines how much the particle is influenced by its own best known position. Values between 1 and 2.5 are commonly used in PSO implementations.                                                              |
| Inertia Weight        | 0.4, 0.6, 0.7, 0.8, 0.9    | Controls the momentum of the particle, with higher values allowing more exploration and lower values favoring exploitation. The range covers typical values used in the literature.                            |
| Total Iterations      | 300, 500, 1000, 1500, 2000 | The maximum number of iterations the algorithm will run. More iterations allow for better convergence but increase computation time.                                                                           |
| Swarm Size            | 10, 30, 50, 75, 100        | Number of particles in the swarm. Larger swarms can explore more space but require more computational resources. These values represent common swarm sizes used in practice.                                   |

Following exhaustive testing across all feasible parameter combinations (totaling 16,000,000 experimental runs), we assessed performance metrics for each configuration. The outcomes are illustrated in Figure 12.

Based on our analysis of these results, we determined the optimal parameter configuration shown in Table 4, which serves as the foundation for PSO algorithm execution.

Similarly, for simulated annealing, we conducted parameter tuning experiments (encompassing 1,000,000 tests) focusing on cooling rate schedules and initial temperature settings as shown in Table 5.





**Figure 12.** Results from the sensitivity analysis for the PSO parameters. (a) Number of iterations to converge. (b) Cognitive Coefficient. (c) Inertia Weight. (d) Social Coefficient. (e) Number of particles used or swarm size.

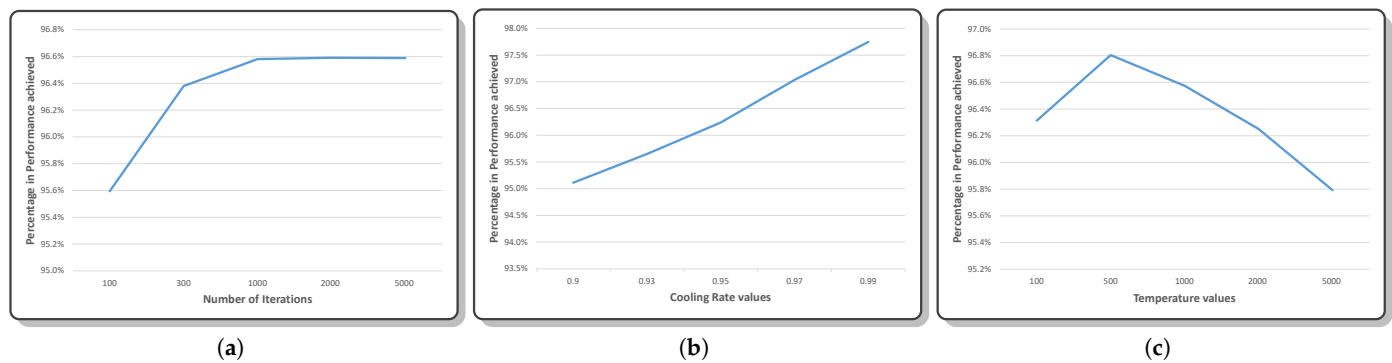
**Table 4.** PSO Parameter Configuration.

| Parameter             | Value |
|-----------------------|-------|
| Social Coefficient    | 1     |
| Cognitive Coefficient | 2.5   |
| Inertia Weight        | 0.9   |
| Total Iterations      | 1000  |
| Swarm Size            | 30    |

**Table 5.** Parameter values in SA sensitivity analysis.

| Parameter        | Values                      | Description                                                                                                                                             |
|------------------|-----------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| Cooling Rate     | 0.9, 0.93, 0.95, 0.97, 0.99 | Controls the rate at which the temperature decreases. A value of 0.99 means the temperature reduces by 1% each iteration, allowing gradual convergence. |
| Total Iterations | 100, 300, 1000, 2000, 5000  | The maximum number of iterations for the algorithm. Higher values improve solution quality but increase computation time.                               |
| Temperature      | 100, 500, 1000, 2000, 5000  | Initial temperature for the annealing process. Higher temperatures allow more exploration, while lower temperatures focus on exploitation.              |

Our sensitivity analysis for SA (depicted in Figure 13) revealed that a cooling rate of 0.99 combined with an initial temperature of 500 provided the most consistent convergence behavior across diverse problem sizes. The final parameter configuration for SA execution is presented in Table 6.



**Figure 13.** Results from the sensitivity analysis for the SA parameters. (a) Number of iterations to converge. (b) Cooling rate. (c) Temperature.

**Table 6.** SA parameter configuration.

| Parameter        | Value |
|------------------|-------|
| Cooling Rate     | 0.99  |
| Total Iterations | 1000  |
| Temperature      | 500   |

For the genetic algorithm comparative evaluation, we employed the parameter configuration outlined in Table 1, with the following modifications: Rather than utilizing the number of generations as the termination criterion, we implemented the fitness stagnation approach with a threshold value of 10, as detailed in Section 4.4.2. We also employed the smart populations as described in Section 4.4.1.

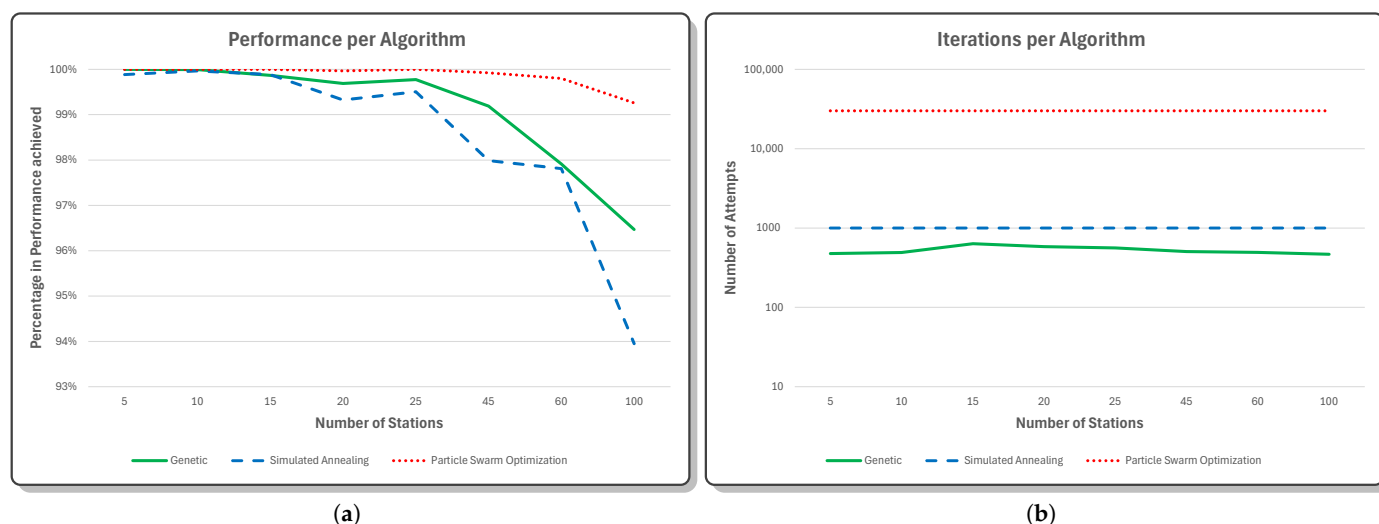
#### Comparative Methodology and Iteration Metrics

To ensure fair comparison across the three metaheuristic approaches, we employed optimal parameter configurations for each algorithm as determined through extensive sensitivity analyses. It is important to clarify the interpretation of “iterations” across different algorithms: for PSO, the total fitness evaluations equal the product of iterations (1000) and swarm size (30 particles), resulting in 30,000 total evaluations per problem instance. For SA, iterations directly correspond to fitness evaluations (1000). For the GA, we employed adaptive termination with a fitness stagnation threshold of 10 generations, allowing the algorithm to terminate when no improvement is observed over 10 consecutive generations, as detailed in Section 4.4.2.

This difference in termination criteria is not an experimental artifact but reflects each algorithm’s optimal operational characteristics. The fixed iteration counts for PSO and SA represent the configurations that yielded their best performance across our sensitivity analyses, while adaptive termination represents a fundamental algorithmic advantage of our GA implementation—its ability to recognize convergence and terminate efficiently without sacrificing solution quality.

#### Experimental Results and Performance Analysis

Figure 14 presents the comparative performance results across the three metaheuristic approaches, evaluated using the same 800 test scenarios employed in our primary genetic algorithm evaluation. The performance metrics represent the accuracy achieved relative to the brute-force ground truth, while iteration counts indicate the computational effort required for convergence.



**Figure 14.** Charts depicting the net effect of each algorithm. (a) Performance of each algorithm. (b) Number of iterations that each algorithm took in order to converge.

The experimental results reveal several critical insights regarding algorithmic performance and computational efficiency. PSO demonstrates the highest average solution quality at 99.87%, marginally exceeding both genetic algorithms (99.11%) and simulated annealing (98.54%). However, this superior performance comes at a substantial computational cost, requiring 30,000 iterations per problem instance compared to the adaptive convergence exhibited by genetic algorithms.

#### Computational Efficiency Considerations

The iteration requirements present a striking contrast between the three approaches. Genetic algorithms demonstrate remarkable efficiency with an average of 527 iterations required for convergence, representing approximately 57 times fewer iterations than PSO and nearly twice the efficiency of simulated annealing. This computational advantage becomes particularly pronounced in operational scenarios where time constraints and computational resources are limiting factors. The fixed iteration approach employed by both PSO and SA, while ensuring consistent computational bounds, reveals inherent inefficiencies in their convergence characteristics. PSO's requirement for 30,000 iterations to achieve optimal performance makes it computationally prohibitive for real-time applications, despite its marginally superior solution quality. The excessive iteration count suggests that PSO's convergence behavior is poorly suited to the continuous coordinate optimization aspects of our launch pad positioning problem. Simulated annealing occupies an intermediate position, requiring 1000 iterations consistently across all problem sizes. While more efficient than PSO, SA's fixed iteration schedule lacks the adaptive termination capabilities demonstrated by our genetic algorithm implementation. The declining performance of SA the increase in problem complexity, particularly evident in the 100-station scenarios where accuracy drops to 93.95%, indicates limitations in its exploration mechanisms for high-dimensional search spaces.

#### Algorithmic Behavior Analysis

The performance degradation patterns observed across increasing problem complexity provide additional insights into each algorithm's suitability for launch pad optimization. Genetic algorithms maintain relatively stable performance across varying problem sizes, with accuracy remaining above 96% even for large-scale scenarios. This consistency stems

from the population-based exploration mechanism that maintains diversity throughout the evolutionary process.

PSO's consistently high performance across all problem sizes demonstrates the effectiveness of swarm intelligence principles for continuous optimization problems. However, the computational cost associated with maintaining large swarm populations for extended iteration counts undermines its practical applicability.

Simulated annealing exhibits the most pronounced performance degradation with the increase in problem complexity. The temperature-based acceptance mechanism, while effective for escaping local optima in smaller search spaces, appears insufficient for maintaining solution quality in larger problem instances. The cooling schedule optimization becomes increasingly critical as the problem size grows, yet our sensitivity analysis revealed limited improvement potential within reasonable computational bounds.

#### Validation of Initial Algorithm Selection

These comparative results strongly validate our initial selection of genetic algorithms for launch pad positioning optimization. While PSO achieves marginally superior solution quality, the 57-fold increase in computational requirements renders it impractical for operational applications where timing constraints are paramount. The adaptive termination capabilities of genetic algorithms provide an optimal balance between solution quality and computational efficiency, characteristics essential for real-world drone deployment scenarios.

The modular integration between genetic algorithm chromosome encoding and mTSP fitness evaluation, as discussed in our initial rationale, proves advantageous compared to the more complex parameter tuning requirements observed in PSO and SA implementations. The natural representation of launch pad coordinates within genetic algorithm chromosomes facilitates direct integration with existing mTSP solvers, while PSO's particle position vectors and SA's solution state representations require additional transformation mechanisms.

Furthermore, the adaptive population size and termination criteria employed in our genetic algorithm implementation demonstrate superior resource utilization compared to the fixed parameter approaches necessitated by PSO and SA. This adaptability proves particularly valuable in operational environments where computational resources and time constraints vary based on mission requirements and available hardware configurations.

#### 4.5. Limitations

While our study demonstrates the computational effectiveness of genetic algorithms for optimizing launch pad positioning in drone operations, several important limitations must be acknowledged that affect the direct applicability of our findings to real-world UAV deployments.

The most significant limitation of our approach is the assumption that minimizing travel distance directly correlates with minimizing energy consumption. This fundamental simplification abstracts away the complex energy dynamics that characterize actual UAV operations. In reality, energy consumption in unmanned aerial vehicles depends on numerous interconnected factors that extend far beyond simple geometric distance calculations. Wind conditions represent one of the most critical factors affecting UAV energy consumption [41]. Headwinds can increase energy requirements compared to still-air conditions, while tailwinds can provide substantial energy savings. Crosswinds introduce additional complexity by requiring course corrections and increased control effort to maintain desired trajectories [42]. Our current model does not account for prevailing wind patterns or seasonal variations that would significantly influence optimal launch pad positioning in operational environments.

Terrain elevation and altitude changes introduce another layer of complexity that our 2D distance-based approach cannot capture. Climbing to higher altitudes requires substantial energy expenditure, while descending allows for energy recovery in some UAV configurations [43]. The presence of hills, valleys, or urban structures can create significant variations in energy consumption even for routes with identical horizontal distances. Furthermore, altitude changes affect air density and atmospheric conditions, which directly impact propulsion efficiency and battery performance.

Battery dynamics present additional challenges that our simplified model does not address. Lithium-ion batteries, commonly used in UAVs, exhibit non-linear discharge characteristics where available energy decreases with temperature, discharge rate, and battery age [44]. Cold weather conditions can further reduce available energy [45]. These factors would necessitate dynamic adjustments to launch pad positioning strategies based on environmental conditions and mission profiles.

Payload variations significantly influence energy consumption patterns in ways that our distance-minimization approach cannot capture. Heavier payloads require more energy for both propulsion and altitude maintenance, while payload distribution affects aerodynamic efficiency. Medical supply deliveries, surveillance equipment, and agricultural sensors each present unique weight and aerodynamic characteristics that would influence optimal launch pad positioning differently [46].

Flight maneuvers and speed profiles represent another area where our simplified model falls short of real-world complexity. Aggressive acceleration and deceleration patterns consume substantially more energy than smooth, consistent flight profiles [47]. Hovering operations, often required for data collection or precision delivery, can consume more energy per unit time compared to forward flight. Our current approach does not differentiate between mission segments that require hovering versus those involving continuous forward motion [48].

Our study also assumes perfect knowledge of station locations and static operational conditions, without accounting for positional uncertainty, measurement noise, or GPS inaccuracies. In practice, target locations may shift due to changing operational requirements, equipment failures, or access restrictions. Dynamic mission requirements would necessitate adaptive launch pad positioning strategies that can respond to changing conditions in real-time. Evaluating robustness under noisy environments represents an important direction for future research before operational deployment.

Despite these limitations, our work provides a valuable foundation for understanding the computational aspects of launch pad optimization and establishes a baseline methodology that can be extended to incorporate these real-world complexities. The genetic algorithm framework we present is sufficiently flexible to accommodate additional constraints and objectives as more sophisticated energy models are developed.

Future research should focus on integrating these real-world factors into comprehensive energy consumption models. This might include developing weather-aware optimization algorithms, incorporating terrain elevation data, and modeling battery performance under varying conditions. Such extensions would transform our current geometric optimization approach into a truly energy-aware deployment strategy suitable for practical UAV operations.

Our findings should therefore be interpreted as addressing a simplified baseline scenario that demonstrates the potential for computational optimization in UAV deployment planning, rather than a complete solution ready for immediate operational deployment. The substantial computational advantages demonstrated by our genetic algorithm approach suggest that even more complex, realistic optimization problems could be addressed efficiently using similar methodologies.

## 5. Conclusions

This study has demonstrated the significant impact of strategic launch pad positioning on drone mission efficiency in multi-drone operations. Our research establishes that optimizing the launch pad location can substantially improve operational performance through reduced travel distances and better route assignments. By comparing brute-force methods with genetic algorithm approaches, we have shown that near-optimal solutions can be achieved with dramatically reduced computational overhead.

The genetic algorithm implementation consistently achieved 97–100% accuracy compared to exhaustive search methods while requiring computational resources reduced by up to four orders of magnitude. This represents a crucial advancement for real-world applications where both solution quality and time/computational efficiency are essential operational factors.

Our exploration of refinements to the basic GA revealed additional opportunities for performance enhancement. The implementation of smart population initialization strategies demonstrated modest improvements in convergence speed while maintaining comparable accuracy levels. Similarly, adaptive termination criteria showed promise in further balancing solution quality with time/computational demands, with even minimal stagnation thresholds of five generations achieving 98.8% solution quality while reducing computational effort by approximately 94% compared to fixed generational approaches.

These findings have significant implications for drone deployment strategies in various application domains. Beyond the computational advantages, optimizing launch pad positioning signals a shift in how we approach the problem—moving from simply optimizing routes to optimizing the entire deployment strategy. This shift encourages system designers to think about the complete operational structure when creating energy-aware drone systems, rather than focusing solely on optimizing flight paths.

### 5.1. Future Work

The findings presented in this study establish a foundation for strategic launch pad positioning in drone operations, yet they simultaneously reveal numerous avenues for advancing this research toward more comprehensive and practically applicable solutions. The computational efficiency demonstrated by our genetic algorithm approach provides an encouraging basis for tackling increasingly complex optimization challenges in UAV deployment strategies.

#### 5.1.1. Integration of Realistic UAV Energy Dynamics

The most pressing direction for future research involves developing sophisticated energy consumption models that transcend the simplified distance-based approach employed in our current work. Rather than assuming direct proportionality between travel distance and energy expenditure, future iterations should incorporate comprehensive physical models that account for the multifaceted nature of UAV energy consumption. This enhancement would require integrating terrain elevation data and transitioning from 2D to 3D path planning algorithms. Digital elevation models could be incorporated into the fitness function to provide more accurate energy estimates that reflect real-world topographical challenges. Wind modeling represents another critical advancement opportunity. Meteorological data integration would allow the optimization algorithm to consider prevailing wind patterns, seasonal variations, and real-time weather conditions. Future work should explore dynamic wind field modeling and its integration with launch pad positioning strategies. Battery dynamics modeling constitutes an equally important research direction. Developing energy consumption models that account for these factors would enable more

accurate optimization of launch pad positions based on expected operational conditions and fleet battery health status.

#### 5.1.2. Dynamic and Adaptive Launch Pad Strategies

Our current approach assumes static launch pad locations throughout mission duration, yet many operational scenarios would benefit from dynamic repositioning capabilities. Future research should investigate scenarios where launch pads can be relocated during missions, introducing temporal optimization dimensions that could significantly enhance operational flexibility. Mobile launch pad platforms, such as ground vehicles or ships, present particularly interesting research opportunities. Dynamic positioning algorithms could continuously optimize launch pad locations based on changing mission requirements, weather conditions, or operational constraints. This would require the development of real-time optimization algorithms capable of handling moving reference points and evolving objective functions. Adaptive algorithms that respond to unexpected events or changing conditions represent another promising research direction. Mission requirements may change due to equipment failures, weather developments, or priority shifts that necessitate rapid reconfiguration of launch pad positions. Future work should explore machine learning approaches that can predict optimal repositioning strategies based on historical data and current operational contexts.

#### 5.1.3. Hybrid Optimization Algorithms

While genetic algorithms demonstrate excellent performance in our current implementation, future work should explore hybrid approaches that combine evolutionary computation with other optimization methodologies. Such hybrid methodologies could leverage the GA's global search capabilities while incorporating local optimization techniques to fine-tune solutions in promising regions. Variable neighborhood search or iterative local search could be integrated as post-processing steps to further minimize the risk of local minima while maintaining the structural advantages of the GA framework.

Machine learning integration presents another avenue for algorithmic enhancement. Reinforcement learning approaches could be employed to adaptively adjust genetic algorithm parameters based on problem characteristics and solution progress.

#### 5.1.4. Advanced Genetic Algorithm Refinements

Building upon the genetic algorithm improvements identified in our current work, future research should investigate weighted selection mechanisms that combine Elite Selection with diversity-promoting approaches such as Roulette Wheel Selection [49] or Stochastic Universal Sampling [50]. This could address potential premature convergence issues while maintaining the advantages of elite selection. Gaussian Mutation [51] operators represent another promising enhancement direction. Unlike Uniform Mutation, Gaussian mutation provides more controlled parameter adjustments that could improve convergence characteristics, particularly when dealing with continuous coordinate spaces.

The research directions outlined above represent a comprehensive roadmap for advancing launch pad positioning optimization from the current computational foundation toward practically applicable solutions that address the full complexity of real-world UAV operations. Each direction offers significant potential for improving operational efficiency while addressing the limitations identified in our current work.



**Author Contributions:** Conceptualization, G.G. and I.K.; methodology, G.G. and I.K.; software, G.G. and I.K.; validation, G.G. and I.K.; formal analysis, G.G. and I.K.; investigation, G.G. and I.K.; resources, G.G. and I.K.; data curation, G.G. and I.K.; writing—original draft preparation, G.G. and I.K.; writing—review and editing, G.G. and I.K.; visualization, G.G. and I.K.; supervision, I.K.; project administration, I.K. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The original data presented in the study are openly available in the “Path planning Starting Point Coordinates” GitHub repository at <https://github.com/ggasteratos/Path-planning-Starting-Point-Coordinates> (accessed on 12 October 2025).

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Vashisth, A.; Batth, R.S.; Ward, R. Existing Path Planning Techniques in Unmanned Aerial Vehicles (UAVs): A Systematic Review. In Proceedings of the 2021 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE), Dubai, United Arab Emirates, 17–18 March 2021; pp. 366–372. [\[CrossRef\]](#)
2. Gasteratos, G.; Karydis, I. Path Planning Optimisation for Multiple Drones: Repositioning the Starting Point. In Proceedings of the IFIP International Conference on Artificial Intelligence Applications and Innovations, Corfu, Greece, 27–30 June 2024; pp. 211–223. [\[CrossRef\]](#)
3. Quamar, M.M.; Al-Ramadan, B.; Khan, K.; Shafiullah, M.; El Ferik, S. Advancements and Applications of Drone-Integrated Geographic Information System Technology—A Review. *Remote Sens.* **2023**, *15*, 5039. [\[CrossRef\]](#)
4. Saeed, R.A.; Omri, M.; Abdel-Khalek, S.; Ali, E.S.; Alotaibi, M.F. Optimal path planning for drones based on swarm intelligence algorithm. *Neural Comput. Appl.* **2022**, *34*, 10133–10155. [\[CrossRef\]](#)
5. Gasteratos, G.; Karydis, I. Efficient Drone Path Planning through Strategic Launch Pad Positioning. In Proceedings of the 1st International Conference on Drones and Unmanned Systems (DAUS’ 2025), Granada, Spain, 19–21 February 2025; IFSA Publishing: Barcelona, Spain, 2025; pp. 254–259. [\[CrossRef\]](#)
6. Drezner, Z.; Wesolowsky, G.O. A Trajectory Approach to the Round-Trip Location Problem. *Transp. Sci.* **1982**, *16*, 56–66. [\[CrossRef\]](#)
7. Drezner, Z.; Wesolowsky, G.O. Facility Location on a Sphere. *J. Oper. Res. Soc.* **1978**, *29*, 997–1004. [\[CrossRef\]](#)
8. Drezner, Z.; Wesolowsky, G.O. Obnoxious Facility Location in the Interior of a Planar Network. *J. Reg. Sci.* **1995**, *35*, 675–688. [\[CrossRef\]](#)
9. Salhi, S.; Rand, G.K. The effect of ignoring routes when locating depots. *Eur. J. Oper. Res.* **1989**, *39*, 150–156. [\[CrossRef\]](#)
10. Nagy, G.; Salhi, S. Location-routing: Issues, models and methods. *Eur. J. Oper. Res.* **2007**, *177*, 649–672. [\[CrossRef\]](#)
11. Fan, X. A path-planning method for uav swarm under multiple environmental threats. *Drones* **2024**, *8*, 171. [\[CrossRef\]](#)
12. Xiong, T.; Li, H.; Ding, K.; Liu, H.; Li, Q. A Hybrid Improved Symbiotic Organisms Search and Sine–Cosine Particle Swarm Optimization Method for Drone 3D Path Planning. *Drones* **2023**, *7*, 633. [\[CrossRef\]](#)
13. Boccadoro, P.; Striccoli, D.; Grieco, L.A. An extensive survey on the Internet of Drones. *Ad Hoc Networks* **2021**, *122*, 102600. [\[CrossRef\]](#)
14. Shirabayashi, J.V.; Ruiz, L.B. Toward uav path planning problem optimization considering the internet of drones. *IEEE Access* **2023**, *11*, 136825–136854. [\[CrossRef\]](#)
15. Jones, M.; Djahel, S.; Welsh, K. Path-planning for unmanned aerial vehicles with environment complexity considerations: A survey. *ACM Comput. Surv.* **2023**, *55*, 234. [\[CrossRef\]](#)
16. Wu, Y.; Low, K.; Pang, B.; Tan, Q. Swarm-based 4d path planning for drone operations in urban environments. *IEEE Trans. Veh. Technol.* **2021**, *70*, 7464–7479. [\[CrossRef\]](#)
17. Lu, F.; Tian, S.; Feng, W.; Bi, H. Green drone multi-package delivery routing problem and its efficient algorithms. *Int. J. Gen. Syst.* **2025**, 1–32. [\[CrossRef\]](#)
18. Diller, J.; Han, Q. Energy-aware drone path finding with a fixed-trajectory ground vehicle. *J. Auton. Transp. Syst.* **2023**, *2*, 17. [\[CrossRef\]](#)
19. Dang, S.; Liu, Y.; Luo, Z.; Liu, Z.; Shi, J. A Survey of the Routing Problem for Cooperated Trucks and Drones. *Drones* **2024**, *8*, 550. [\[CrossRef\]](#)
20. Brimberg, J.; Hansen, P.; Mladenovic, N.; Salhi, S. A Survey of Solution Methods for the Continuous Location-Allocation Problem. *Int. J. Oper. Res.* **2008**, *5*, 1–12.

21. Mara, S.T.W.; Kuo, R.; Asih, A.M.S. Location-routing problem: A classification of recent research. *Int. Trans. Oper. Res.* **2021**, *28*, 2941–2983. [CrossRef]
22. Bektaş, T. The multiple traveling salesman problem: An overview of formulations and solution procedures. *Omega-Int. J. Manag. Sci.* **2006**, *34*, 209–219. [CrossRef]
23. Cheikhrouhou, O.; Khoufi, I. A comprehensive survey on the Multiple Traveling Salesman Problem: Applications, approaches and taxonomy. *Comput. Sci. Rev.* **2021**, *40*, 100369. [CrossRef]
24. Jain, M.; Saihijpal, V.; Singh, N.; Singh, S.B. An Overview of Variants and Advancements of PSO Algorithm. *Appl. Sci.* **2022**, *12*, 8392. [CrossRef]
25. Storn, R.; Price, K.V. Differential Evolution—A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [CrossRef]
26. Kang, M.; Lee, C. An Exact Algorithm for Heterogeneous Drone-Truck Routing Problem. *Transp. Sci.* **2021**, *55*, 1088–1112. [CrossRef]
27. Liang, Y.J.; Luo, Z.X. A Survey of Truck–Drone Routing Problem: Literature Review and Research Prospects. *J. Oper. Res. Soc. China* **2022**, *10*, 343–377. [CrossRef]
28. Fang, S.; Luo, Z. Multi-Traveling Salesman Algorithm for UAV Swarms Based on Improved Genetic Algorithm. In Proceedings of the 2024 IEEE 6th International Conference on Power, Intelligent Computing and Systems (ICPICS), Shenyang, China, 26–28 July 2024; pp. 1027–1032. [CrossRef]
29. C# Genetic Algorithm Library. Available online: <https://github.com/giacomelli/GeneticSharp> (accessed on 5 December 2024).
30. Whitley, D. A Genetic Algorithm Tutorial. *Stat. Comput.* **1994**, *4*, 65–85. [CrossRef]
31. Back, T. 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, 27–29 June 1994; Volume 1, pp. 57–62. [CrossRef]
32. Eiben, A.E.; Smith, J.E. *Introduction to Evolutionary Computing*; Springer: Berlin/Heidelberg, Germany, 2015. [CrossRef]
33. López-Ibáñez, M.; Stützle, T.; Dorigo, M., Ant Colony Optimization: A Component-Wise Overview. In *Handbook of Heuristics*; Springer International Publishing: Cham, Switzerland, 2016; pp. 1–37. [CrossRef]
34. van Laarhoven, P.J.M.; Aarts, E.H.L. Simulated Annealing: Theory and Applications. In *Mathematics and Its Applications*; Springer: Dordrecht, The Netherlands, 1987. [CrossRef]
35. Bernard, R.; Robert, J.G.; Mei-Ling, T.L. The Wilcoxon signed rank test for paired comparisons of clustered data. *Biometrics* **2006**, *62*, 185–192. [CrossRef]
36. Deakin, R.; Bird, S.; Grenfell, R. The Centroid? Where would you like it to be? *Cartography* **2002**, *31*, 153–167. [CrossRef]
37. Minsker, S.; Strawn, N. The Geometric Median and Applications to Robust Mean Estimation. *arXiv* **2023**, arXiv:2307.03111. [CrossRef]
38. Kwon, H.; Oh, S.; Baek, J.W. Algorithmic Efficiency in Convex Hull Computation: Insights from 2D and 3D Implementations. *Symmetry* **2024**, *16*, 1590. [CrossRef]
39. Likas, A.; Vlassis, N.; Verbeek, J.J. The global k-means clustering algorithm. *Pattern Recognit.* **2003**, *36*, 451–461. [CrossRef]
40. Deng, D. DBSCAN clustering algorithm based on density. In Proceedings of the 2020 7th international forum on electrical engineering and automation (IFEEA), Hefei, China, 25–27 September 2020; pp. 949–953.
41. Gu, R.; Zhao, Y.; Ren, X. Integrating wind field analysis in UAV path planning: Enhancing safety and energy efficiency for urban logistics. *Chin. J. Aeronaut.* **2025**, 103605. [CrossRef]
42. Thibbotuwawa, A.; Nielsen, P.; Zbigniew, B.; Bocewicz, G. Factors Affecting Energy Consumption of Unmanned Aerial Vehicles: An Analysis of How Energy Consumption Changes in Relation to UAV Routing. In Proceedings of the Information Systems Architecture and Technology: 39th International Conference on Information Systems Architecture and Technology—ISAT 2018, Nysa, Poland, 16–18 September 2018; Świątek, J., Borzowski, L., Wilimowska, Z., Eds.; Springer: Cham, Switzerland, 2019; pp. 228–238.
43. Nagy, I.; Laufer, E. Energy-Optimized 3D Path Planning for Unmanned Aerial Vehicles. *Appl. Sci.* **2024**, *14*, 6988. [CrossRef]
44. Lv, S.; Wang, X.; Lu, W.; Zhang, J.; Ni, H. The Influence of Temperature on the Capacity of Lithium Ion Batteries with Different Anodes. *Energies* **2022**, *15*, 60. [CrossRef]
45. Ercan, H.; Ayaz, F.A.; Ulucan, H. Effect of Temperature, Pressure and Humidity on Battery Consumption in Unmanned Aerial Vehicles. *J. Aviat.* **2025**, *9*, 5–12. [CrossRef]
46. Seth, A.; James, A.; Kuantama, E.; Mukhopadhyay, S.; Han, R. Aerodynamics and Sensing Analysis for Efficient Drone-Based Parcel Delivery. In Proceedings of the 2023 16th International Conference on Sensing Technology (ICST), Hyderabad, India, 17–20 December 2023; pp. 1–6. [CrossRef]
47. Qi, Y.; Jiang, H.; Huang, G.; Yang, L.; Wang, F.; Xu, Y. Multi-UAV path planning considering multiple energy consumptions via an improved bee foraging learning particle swarm optimization algorithm. *Sci. Rep.* **2025**, *15*, 14755. [CrossRef] [PubMed]

48. Su, S.; Shan, X.; Yu, P.; Wang, H. Energy Consumption Performance of a VTOL UAV In and Out of Ground Effect by Flight Test. *Drones* **2024**, *8*, 625. [[CrossRef](#)]
49. Pasala, S.; Kumar, B.N.; Satapathy, S.C. A Study of Roulette Wheel and Elite Selection on GA to Solve Job Shop Scheduling. In *Proceedings of the International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA)*; Satapathy, S.C., Udgata, S.K., Biswal, B.N., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 477–485. [[CrossRef](#)]
50. Pencheva, T.; Atanasov, K.; Shannon, A. Modelling of a stochastic universal sampling selection operator in genetic algorithms using generalized nets. In *Proceedings of the Tenth International Workshop on Generalized Nets*, Sofia, Bulgaria, 5 December 2009; pp. 1–7.
51. Bell, O. Applications of Gaussian Mutation for Self Adaptation in Evolutionary Genetic Algorithms. *arXiv* **2022**, arXiv:2201.00285. [[CrossRef](#)]

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.