


## Article

# Optimizing Autonomous Multi-UAV Path Planning for Inspection Missions: A Comparative Study of Genetic and Stochastic Hill Climbing Algorithms

Faten Aljalaud <sup>1,2,\*</sup> and Yousef Alohalı <sup>2</sup> 

<sup>1</sup> Computer Science Department, Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh 11564, Saudi Arabia

<sup>2</sup> Computer Science Department, College of Computer and Information Sciences, King Saud University, Riyadh 11451, Saudi Arabia; yousef@ksu.edu.sa

\* Correspondence: 438203858@student.ksu.edu.sa

**Abstract:** Efficient path planning is vital for multi-UAV inspection missions, yet the comparative effectiveness of different optimization strategies has not received much attention. This paper introduces the first application of the Genetic Algorithm (GA) and Hill Climbing (HC) to multi-UAV inspection of indoor pipelines, providing a unique comparative analysis. GA exemplifies the global search strategy, while HC illustrates an enhanced stochastic local search. This comparison is impactful as it highlights the trade-offs between exploration and exploitation—two key challenges in multi-UAV path optimization. It also addresses practical concerns such as workload balancing and energy efficiency, which are crucial for the successful implementation of UAV missions. To tackle common challenges in multi-UAV operations, we have developed a novel repair mechanism. This mechanism utilizes problem-specific repair heuristics to ensure feasible and valid solutions by resolving redundant or missed inspection points. Additionally, we have introduced a penalty-based approach in HC to balance UAV workloads. Using the Crazyswarm simulation platform, we evaluated GA and HC across key performance metrics: energy consumption, travel distance, running time, and maximum tour length. The results demonstrate that GA achieves a 22% reduction in travel distance and a 23% reduction in energy consumption compared to HC, which often converges to suboptimal solutions. Additionally, GA outperforms HC, Greedy, and Random strategies, delivering at least a 13% improvement in workload balancing and other metrics. These findings establish a novel and impactful benchmark for comparing global and local optimization strategies in multi-UAV tasks, offering researchers and practitioners critical insights for selecting efficient and sustainable approaches to UAV operations in complex inspection environments.

**Keywords:** autonomous; multi-UAV; path planning; inspection; genetic algorithm



Academic Editor: Chung-Neng Huang

Received: 2 October 2024

Revised: 17 December 2024

Accepted: 23 December 2024

Published: 27 December 2024

**Citation:** Aljalaud, F.; Alohalı, Y. Optimizing Autonomous Multi-UAV Path Planning for Inspection Missions: A Comparative Study of Genetic and Stochastic Hill Climbing Algorithms. *Energies* **2025**, *18*, 50. <https://doi.org/10.3390/en18010050>

**Copyright:** © 2024 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

Multi-UAV systems have attracted considerable interest in recent years because they can be used adaptably in different fields, such as environmental monitoring, disaster management, and multi-UAV inspection missions [1,2].

The inspection robot market is projected to experience a significant growth rate of 30.9% due to the growing emphasis on worker safety and the implementation of smart construction, which involves using autonomous systems and vehicles on construction sites [3]. UAV inspections, in particular, have been growing tremendously since 2010 [4], representing 45% of the total market value for UAVs [5].

Inspection tasks of UAVs can vary, such as inspection of bridges, powerlines, pipelines, and vessels [4,6,7]. One of the critical inspection applications is pipeline inspection missions [6]—particularly the inspection of pipelines and sprinkler systems in indoor infrastructures. Many countries around the world have regulations that require properties to install fire detection and sprinkler system pipelines, as indicated by the National Fire Protection Association (NFPA) in the US, the UK, and European standards like BS EN 12845 [8,9].

Efficient path planning is among the most challenging and core parts of multi-UAV missions, performing all kinds of tasks [10]. Optimal path planning is crucial to the success of multi-UAV inspection missions [3], which not only provides better coverage but also saves overall travel distance and, thus, energy. Drones may overlap their paths or miss critical inspection points, and they could run out of power before the end of a mission unless advanced navigation strategies are used. Moreover, optimal path planning and obstacle avoidance ensure no UAV collides with the others while operating together, adding safety to the mission.

Recent studies focus on finding the best methods to utilize UAVs' existing energy resources. For this reason, pipeline inspection missions involving multiple UAVs should prioritize reducing their energy consumption [11].

In our previous work, we introduced a dataset designed to support research in UAV-based pipeline inspection missions. The dataset contains 1300 samples of corrosion defects in pipelines categorized by two different maps [12]. Each instance corresponds to a different level of defect severity. The dataset provided the necessary data to design and evaluate algorithms that improve the performance of multi-UAV inspection missions.

In this study, we utilize this dataset to evaluate the performance of a Genetic Algorithm (GA) implemented within the CrazySwarm platform for multi-UAV path planning in pipeline inspection missions. To our knowledge, this is the first application of GA for optimizing path planning in multi-UAV pipeline inspection missions, providing novel insights into its performance compared to benchmarks.

The primary contributions of this study can be described as follows:

- Adapt Genetic Algorithm (GA) and Hill Climbing (HC) for a multi-UAV inspection mission:  
This includes developing a novel, problem-specific repair mechanism to ensure that crossover and mutation operations produce valid chromosomes and avoiding issues like duplicate or missing inspection locations, which are critical in real-world UAV inspections to ensure all areas are covered efficiently.
- Introduce a penalty computation to the HC algorithm:  
This aims to maintain balanced solutions regarding UAV tour lengths, which is important for distributing workloads evenly across UAVs, preventing overburdening of any UAV, and ensuring longer operational lifetimes.
- Compare the performance of GA and HC using different performance measures:  
This comparison is crucial for understanding the strengths and weaknesses of each algorithm in various scenarios, helping researchers and practitioners choose the best method based on specific requirements like efficiency or robustness.
- Analyze energy consumption through simulation results:  
Given the growing importance of energy efficiency in UAV operations, this objective contributes to a deeper understanding of how different algorithms impact energy usage, with implications for sustainable and efficient UAV deployments.

The rest of this paper is organized as follows: Section 2 presents the related work; Section 3 presents the problem formulation, assumptions, and implementation details for GA and HC; and the environment used to simulate the parameters is defined in the Experiment section. In the Results and Discussion section, we provide the results and

compare GA with HC algorithms against other benchmarks in terms of traveled distance, max tour length, running time, and energy consumption. Finally, the Conclusion section summarizes this study's findings and suggests further research.

## 2. Related Work

A review of the related literature reveals various applications of Genetic Algorithms (GA) in optimizing complex problems, including multi-UAV path planning. GA are evolutionary algorithms based on the process of natural selection, and they can be used to solve complex optimization problems. They work with a cycle of selection, crossover, and mutation to get closer and closer in an iterative manner toward the population whole of candidate solutions. GAs keep the population diverse, and by mixing parts of the strongest individuals, they can traverse a vast search space to avoid solutions falling into local optima. The traversing makes them well suited to jobs like multi-UAV path planning, where the best route optimized for minimal travel distance and energy usage is crucial. GA generates high-quality solutions for search and optimization problems [13]. GA path planning can greatly aid UAVs. First, GA efficiently explores large solution spaces, making them ideal for complex problems with many answers. They also have the flexibility to incorporate various mission objectives and unanticipated conditions. GA's constant solution improvement provides robustness under challenging situations [14].

Several studies have applied GA to different inspection and maintenance tasks. For instance, Zheng et al. [15] used GA to automatically inspect metallic surfaces using images. Their work presents an experimental system for inspecting metallic surfaces with machine vision and proposes an intelligent approach using morphology and GA to detect structural defects on bumpy surfaces.

Haladuick et al. [16] applied GA to the inspection and maintenance planning field. The GA effectively determined the best inspection and maintenance plan for a pressure vessel system by evaluating the plans using the failure-to-repair cost ratio. Similarly, Moura et al. [17] applied GA to find the best inspection plan considering the inspection's cost and the probability of failure. They tested their approach on oil and gas separator vessels.

Dedeurwaerder et al. [18] employed a GA to address the issue of navigating independent climbing robots for inspecting steel truss bridges. The robots, dispatched from four depots at the four corners of the bridge, must traverse each component of the bridge truss while limiting the distance traveled and ensuring an equitable distribution of tasks among the robots.

In 2019, Cao et al. [19] introduced a GA-based approach for multi-UAV path planning in cooperative reconnaissance missions. The topic was analyzed in the scenario where each UAV has a distinct base or starting point. Graph theory was employed to convert this problem into a combinatorial optimization problem focused on finding the shortest path. The goal was to shorten the duration of occupancy in order to detect adversary radars effectively. However, they neglected to assess their approach compared to the proposed algorithm.

Hill Climbing (HC) is chosen as a comparison algorithm to GA because it represents a contrasting approach to optimization. HC's simplicity and efficiency in finding local optima through incremental improvements make it a suitable baseline against the more complex global search strategy of GA [20]. This comparison highlights the trade-offs between an enhanced stochastic local search (HC) and a stochastic global search (GA), providing insights into their performance in multi-UAV path planning. Evaluating both methods reveals the strengths and weaknesses of locally focused versus globally oriented optimization strategies, illustrating their practical relevance in real-world scenarios. Recent research also highlights the potential of hybrid optimization approaches that combine the

strengths of multiple algorithms. For example, a hybrid optimization approach combining Particle Swarm Optimization (PSO) and GA for multi-UAV path planning was developed by Haghighi et al. [21]. Their method, HPSOGA, capitalizes on the strengths of PSO for escaping local minima and GA for solution refinement, improving both computational efficiency and solution quality. HPSOGA demonstrates superior performance in reducing computational time and enhancing path quality, especially in complex multi-agent scenarios. Similarly, Ali et al. [22] developed a hybrid metaheuristic algorithm combining Maximum–Minimum Ant Colony Optimization (MMACO) and Differential Evolution (DE) to optimize the selection of the most suitable colony for targeting designated areas in dynamic environments. By dividing the ant colony into sub-colonies for localized optimization and employing DE for global refinement, the MMACO-DE algorithm successfully addressed collision avoidance and coordination challenges in complex terrains. Wang et al. [23] further proposed an improved Genetic Algorithm integrated with Simulated Annealing (GAISA) to solve multi-UAV task allocation problems modeled as MTSPs. GAISA combines the fast convergence of GA with SA’s ability to avoid local optima, achieving superior task allocation results in simulations.

While hybrid methods like HPSOGA exhibit great potential, the focus of this study is on exploring the standalone capabilities of GA to establish a baseline for its application in multi-UAV path planning. Unlike hybrid approaches such as HPSOGA, MMACO-DE, and GAISA, our study uniquely focuses on the original GA approach to establish a robust baseline for optimizing multi-UAV inspection tasks, particularly in pipeline networks, an area yet to be explored. A detailed comparison of the proposed work with existing studies is presented in Table 1, which highlights the methodologies, objectives, problem domains, and key results of each study for better understanding.

**Table 1.** Comparison of existing studies in the literature, highlighting the methodologies, objectives, problem domains, and key results.

Study	Methodology	Objective	Problem Domain	Key Results
Zheng et al. [15]	Genetic Algorithm (GA)	Inspect metallic surfaces using machine vision	Surface inspection tasks	Proposed an intelligent approach using GA for detecting structural defects on bumpy surfaces.
Haladuick et al. [16]	Genetic Algorithm (GA)	Plan inspections and maintenance	Pressure vessel inspection and maintenance	Determined optimal inspection and maintenance plans based on failure-to-repair cost ratio.
Moura et al. [17]	Genetic Algorithm (GA)	Optimize inspection plans	Oil and gas separator vessels	Developed cost-effective inspection plans considering probability of failure.
Dedeurwaerder et al. [18]	Genetic Algorithm (GA)	Navigate robots for bridge inspection	Steel truss bridge inspection	Optimized task allocation to minimize travel distance and ensure equitable task distribution.
Cao et al. [19]	Genetic Algorithm (GA)	Plan paths for cooperative UAV reconnaissance	Multi-UAV reconnaissance missions	Employed graph theory to optimize routes, reducing duration of occupancy for radar detection.
Haghighi et al. [21]	HPSOGA (Hybrid PSO-GA)	Enhance path quality and reduce computation time	Multi-UAV path planning	Demonstrated superior computational efficiency and solution quality in multi-agent scenarios.
Ali et al. [22]	MMACO-DE (Hybrid Metaheuristic)	Optimize colony path selection	Multi-UAV path planning in dynamic environments	Achieved robust collision avoidance and reduced travel distances in complex terrains.
Wang et al. [23]	GAISA (Genetic Algorithm + Simulated Annealing)	Task allocation for multi-UAVs	Multi-UAV MTSP problem	Outperformed standalone GA and SA, achieving better task allocation and avoidance of local optima.

According to recent reviews on UAVs and path planning, such as the one by Saadi et al. [24] and Israr et al. [25], GA has not been applied to multi-UAV path planning without hybridization with other algorithms, and HC has never been applied to a problem like ours. Unlike previous works that primarily focus on hybrid GA models or different applications

of GA, our work uniquely applies the original GA to optimize path planning in complex multi-UAV inspection tasks, particularly in pipeline networks, a domain that has not yet been explored.

### 3. Method

This study addresses the same multi-UAV path planning inspection mission problem described in our previous work [26]. The objective is to optimize the inspection paths for multiple UAVs to minimize the total travel distance while ensuring complete coverage of the inspection area.

#### 3.1. Path Planning Model

Before tackling the problem of multi-UAV path planning for an inspection mission, it is necessary to establish some fundamental assumptions:

1. UAVs travel in straight lines.
2. Altitude layering [27]: Pipes and other unmanned aerial vehicles (UAVs) can collide. Therefore, UAVs must operate at altitudes either above or below the level of pipes. UAVs employ altitude layering upon takeoff. Subsequently, the z dimension remains unchanged following the takeoff of the UAV. It is important to note that takeoff and landing times are negligible.
3. The dimensions of the cell are either smaller or equal to the detection range of the UAV.
4. UAVs can identify defects within a specific range of the pipe.
5. The battery swapping time is disregarded [5].
6. To prevent collisions, the number of operational UAVs must be limited to one per location [28].
7. Due to the nature of this indoor inspection mission, UAVs are not affected by weather conditions.

The variables of the proposed model are as follows:

$U$ —total number of UAVs;

$u$ —index of a UAV;

$D_{i,j}$ —distance traveled by UAV between the  $i$ th waypoint and  $j$ th waypoint;

$N_{u,w}$ —number of waypoints in a viable route for UAV  $u$ ;

$Path_u$ —viable path for a UAV  $u$ ;

$Cost_{Path_u}$ —total distance traveled associated with  $Path_u$ ;

$\delta_m$ —starting energy of every UAV;

$\delta_u$ —UAV's energy at a given time;

$r_{i,j}$ —energy consumption between the  $i$ th waypoint and  $j$ th waypoint;

$\sum_j r_{u,j}$ —total energy consumption of the  $u$ th UAV;

$\mu$ —coefficient to denote energy consumption.

In this model, the solution  $Path_u$  is represented as a sequence of three-dimensional waypoints [29].

Equation (1) defines  $Path_u$ :

$$Path_u = (w_{u,1}, w_{u,2}, \dots, w_{u,N_{u,w}}) \quad (1)$$

Each waypoint ( $w_{u,i}$ ) of  $Path_u$  represent the  $i$ -th waypoint of UAV  $u$  path.

We state our problem as a variant of the singly constrained traveling salesman problem (TSP) [30], where there might be more than one salesmen in which case it becomes NP-hard. Moreover, the energy constraint was added to the path selection decision. The UAV  $u$  cannot choose a path that exceeds its energy limit,  $\delta_u$ .

Our objective ( $F_{Objective}$ ) focuses on improving the quality of trajectory planning by reducing total UAV flight distance.

Equation (2) describes the above model.

$$F_{Objective} = \begin{cases} \min \left( \left( \sum_{i=1}^U Cost_{Path_i} \right) \right) \\ s.t. \begin{cases} \sum_j r_{1,j} < \delta_1 \\ \dots \\ \sum_j r_{U,j} < \delta_U \end{cases} \end{cases} \quad (2)$$

Assuming the velocity is 1 m/s, we computed the energy consumption  $r$  as follows [30–32]:

$$r_{i,j} = \mu * D_{i,j} \quad (3)$$

The following Equation (4) explains this relationship mathematically as the summation of distances covered by trajectory to  $Path_u$ . As a result, the equation finds the cost associated with each trajectory.

$$Cost_{Path_u} = \sum_{j=1}^{N_{u,w}-1} D_{W_{u,j}, W_{u,j+1}} \quad (4)$$

To solve this multi-UAV path planning optimization problem efficiently, it is essential to select algorithms capable of addressing its multi-objective and multi-agent nature, which involves minimizing energy consumption and balancing UAV workloads across dynamic environments. While A\* is widely used for single-agent shortest-path problems, its reliance on heuristics for a single goal state makes it unsuitable for multi-objective optimization tasks, such as those in multi-UAV path planning or TSP problems. A review of the literature and initial implementation attempts revealed no admissible heuristic capable of addressing multiple goals simultaneously. This limitation has likely contributed to A\* not being widely used as a benchmark for TSP or multi-UAV path planning problems.

Similarly, Particle Swarm Optimization (PSO) is less effective for discrete problems like TSP due to its reliance on a velocity update mechanism designed for continuous solution spaces [33,34]. While adaptations like discrete PSO have been proposed, studies, including Clerc's influential work [34], have shown it to be less efficient compared to other algorithms. Recent benchmarks confirm that PSO performs poorly on TSP tasks, delivering less effective results than other metaheuristics [35]. These limitations further justify our focus on GA and HC, which are better suited for the discrete and multi-objective nature of the multi-UAV path planning problem.

In contrast, GA and HC offer robust alternatives for addressing the objectives of this study. GA excels in global exploration and maintaining solution diversity, avoiding local optima, and handling complex, non-linear search spaces. HC complements GA with its computational efficiency and ability to refine local solutions. Together, these algorithms are well-suited for the dynamic and multi-objective nature of multi-UAV inspection missions, where balancing workloads and minimizing energy consumption are critical.

### 3.2. Genetic Algorithm

We adopted the grouping-based GA chromosome representation [36], reviewed in [37], and detailed in Table 2, which outlines the algorithmic process:

- Initial population
  - Generate an initial population of random chromosomes.
  - Calculate the fitness of each chromosome in the population.



**Table 2.** Steps of Genetic Algorithm (GA).

Step	Description
1	<ul style="list-style-type: none"> <li>- Initialize Population</li> <li>- Generate an initial population of random chromosomes.</li> <li>- Calculate the fitness of each chromosome in the population.</li> </ul>
2	Repeat Until Termination Criteria Are Met <ol style="list-style-type: none"> <li>a. Selection: Select two parent chromosomes.</li> <li>b. Crossover:               <ul style="list-style-type: none"> <li>- Choose two random crossover points.</li> <li>- Perform crossover to generate offspring chromosomes.</li> <li>- Apply problem-specific repair heuristics to ensure validity.</li> <li>- Add repaired offspring to the population.</li> </ul> </li> <li>c. Mutation:               <ul style="list-style-type: none"> <li>- Randomly select one UAV tour from a chromosome.</li> <li>- Swap two distinct locations within the selected tour.</li> </ul> </li> <li>d. Replacement:               <ul style="list-style-type: none"> <li>- Combine parents and offspring into an expanded population.</li> <li>- Sort the population by fitness values.</li> <li>- Keep the best chromosomes to return the population to its original size.</li> </ul> </li> </ol>
3	<ul style="list-style-type: none"> <li>- Stop the algorithm if the fitness stagnates for 30 generations or a maximum of 1000 generations is reached.</li> </ul>
4	<ul style="list-style-type: none"> <li>- Return the best chromosome as the solution.</li> </ul>

- Selection

The GA employs a rank-based roulette-wheel selection strategy. This strategy gives a higher probability to chromosomes with lower fitness, making them more likely to be chosen as parents for the crossover step.

- Crossover

Two crossover points are chosen at random after selecting two different chromosomes as parents. This operation generates offspring chromosomes, which may not always be valid for the multi-UAV inspection mission. Specifically, the crossover operation can produce infeasible chromosomes, which violate the problem's essential constraints.

To address this, we developed a problem-specific repair heuristic to ensure the validity and feasibility of the resulting chromosomes after the crossover operation. This repair process is critical to maintaining the integrity of the solution, as it corrects common violations that arise in the multi-agent UAV inspection context. A chromosome is considered infeasible if it exhibits one or both of the following violations:

- Location Duplication: A location is inspected by more than one UAV, leading to redundancy.
- Location Omission: A location is not assigned to any UAV, resulting in incomplete coverage.

Our repair mechanism is designed to handle these cases efficiently:

- Location Duplication Handling: The repair mechanism scans all UAV tours, starting with the first UAV, and maintains a record of all inspected locations. If a location appears more than once across different tours, it is deleted from the duplicate tour(s), ensuring that only one UAV inspects each location.
- Location Omission Handling: Once duplicates are removed, the mechanism checks for missing locations that have not been assigned to any UAV. The nearest UAV (based on the most recent inspected location) is identified for each missing location, and the missing location is added to its tour, ensuring complete coverage of the inspection area.

This repair process ensures that the crossover operation results in manageable solutions, which is crucial in adapting the Genetic Algorithm to multi-UAV path planning. After applying the repair heuristics, we may find that some tours must be in order. To address this, we shuffle the affected tours randomly to restore a valid order, ensuring that the algorithm continues to explore diverse solutions.

By employing this novel repair mechanism, our approach effectively prevents the generation of infeasible solutions, maintaining both the completeness and correctness of the UAV tours. This contribution is particularly significant in the multi-UAV context, where each location must be inspected precisely once, and violations in the chromosome structure could otherwise compromise the quality of the solution.

Finally, the population includes both parents and offspring produced by the crossover. We implemented an efficient hashing-based data structure to sort the population using fitness values as keys, facilitating efficient selection and replacement processes.

- Mutation

In the mutation of the chromosome, one of the chromosome's UAVs is chosen at random and selected to swap two distinct locations in the UAV's tour.

- Replacement

Our approach will expand the population after the crossover and mutation because we will keep both the parents and offspring. However, the population must be minimized to the original population size before evaluating the termination criteria. In other words, the chromosomes with the lowest fitness values in the original population will be replaced with those with higher fitness values.

- Termination criterion

Since the original paper [36] used a time constraint as the stopping criteria, we used either one of two stopping criteria as the paper [38]. The number of generations is 1000 as in [39–41], and we terminate the execution once the fitness does not improve for more than 30 generations.

The parameters for the GA were carefully chosen based on insights from the existing literature (see Table 3). The number of generations was set to 1000, following recommendations in [36], which demonstrated its effectiveness in achieving high-quality solutions for similar optimization problems. The population size was also guided by [36], ensuring sufficient diversity to explore the solution space effectively while maintaining computational efficiency. The crossover and mutation rates were set to 0.9 and 0.03, respectively, based on the findings of [42], which emphasize their importance in preserving genetic diversity and avoiding local optima.

**Table 3.** Genetic Algorithm parameters.

Population size	100
Crossover rate	90%
Mutation rate	3%
Generations number	1000

### 3.3. Hill Climbing Algorithm

The Hill Climbing (HC) algorithm begins with an initial solution and iteratively moves to a neighboring solution whenever an improvement in fitness is observed [20]. This step is crucial in the algorithm's progression.

The solution representation was based on the single-chromosome GA representation described in [36], using a permutation-based approach. Our algorithm is shown in Table 4.



We generate new solutions from neighbors by permuting two places in a UAV tour, interchange locations between two different UAV tours, and directing a place from one tour to another.

**Table 4.** Steps of Hill Climbing algorithm.

Step	Description
1	Set parameters(Maximum iterations, initial penalty factor, and penalty reduction rate)
2	Generate an initial solution (balanced tour lengths between UAVs)
3	Set the initial solution as the current
4	Loop until Maximum iteration is reached
5	- Calculate the fitness of the solution
6	- Generate new solutions from the neighbors of the current solution
7	- Set the current solution as the new one if it has a better fitness value
8	- Reduce the penalty factor over time
9	Return the solution

We introduce stochastic elements into the search process using different methods to generate neighbors. This randomness, which includes the random selection of locations to swap or move, enables HC to investigate a broader range of solutions. Additionally, it generates neighbors through multiple methods, increasing variability and the potential to discover optimal solutions by avoiding deterministic pitfalls.

However, moving locations between tours can alter the balance between different UAV tours, particularly in terms of tour lengths. To address this, the fitness function incorporates a penalty mechanism to account for the difference between the maximum and minimum tour lengths, promoting more balanced solutions.

The steps of penalty calculations are shown in Figure 1. The overall fitness score combines travel distance and workload imbalance, where the following apply:

- Higher scores represent more efficient travel routes and better workload distribution.
- Lower scores indicate less efficient or poorly balanced solutions.

The penalty mechanism consists of the following components:

- Initial Penalty Factor: Starts at a high value to penalize imbalances in early iterations strongly.
- Minimum Penalty Factor: Defines the minimum value of the penalty factor so that it does not decay away.
- Penalty Reduction Rate: Defines the minimum value of the penalty factor so that it does not decay away.

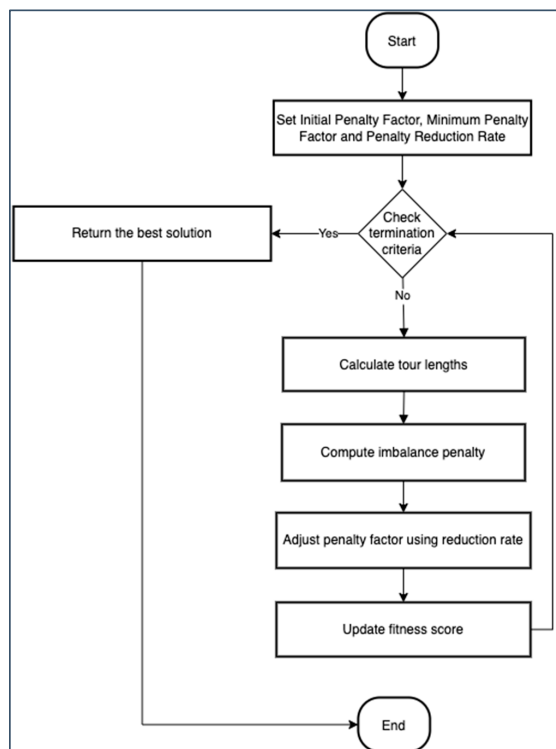
The imbalance penalty is calculated as follows:

$$\text{Imbalance Penalty} = [\max(\text{tour length}) - \min(\text{tour length})] * \text{penalty factor} \quad (5)$$

In order to redirect the algorithm's focus from workload balancing to optimizing total travel distance in subsequent iterations, the penalty factor is progressively reduced over time using the following equation:

$$\text{Penalty factor}_{t+1} = \max(\text{Minimum Penalty factor}, \text{Penalty factor}_t \times \text{Penalty Reduction Rate}) \quad (6)$$

The overall fitness function used in the fitness method combines both the total distance and the imbalance penalty.



**Figure 1.** Steps of the penalty calculations in the Hill Climbing algorithm, showing how imbalance penalties are computed and adjusted to ensure balanced UAV workload distribution.

This approach ensures that the penalty remains significant during the early iterations, gradually diminishing as the algorithm progresses. By doing so, it balances the trade-off between exploration and exploitation—in other words, between emphasizing workload balancing (exploration) and refining efficient routes (exploitation).

Table 5 illustrates how the penalty factor dynamically reduces workload imbalance over iterations, improving the distribution of locations (tour lengths) among UAVs. Initially, with a high penalty factor of 20.00, the workload imbalance is significant at 40.00. As iterations progress, the penalty factor gradually decreases, leading to a more balanced distribution of locations. By iteration 5, the workload imbalance drops to 32.58, and by iteration 15, the imbalance further improves to 20.53 as the penalty factor reduces to 9.75. This gradual reduction in penalty ensures that the HC algorithm encourages workload balance while maintaining exploration across the search space.

**Table 5.** Penalty factor reduction over iterations and its effect on workload imbalance, showing improved balance as iterations progress.

Iteration	Penalty Factor	Workload Imbalance
0	20.00	40.00
1	19.00	40.00
5	15.48	32.58
7	14.70	32.58
10	11.97	26.54
12	11.38	26.54
14	10.27	26.54
15	9.75	20.53

The parameters of the HC algorithm, such as the Initial Penalty Factor, Minimum Penalty Factor, and Penalty Reduction Rate, were chosen empirically and are listed in Table 6.

**Table 6.** Parameters of HC.

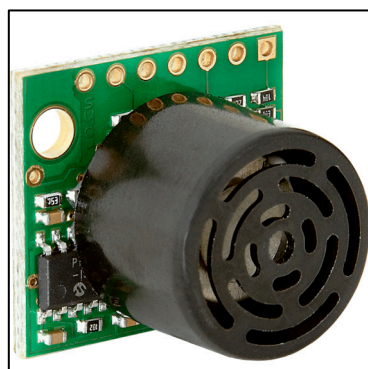
Initial Penalty Factor	20
Minimum Penalty Factor	5
Penalty Reduction Rate	0.95
Iterations	2000

#### 4. Experiment

Our experiments employed CrazySwarm, a quadrotor research platform framework that integrated well-tested hardware and software components for an autonomous aerial robotics study [43]. It seamlessly transitions from simulation to real-world implementation [44]. In our work, we implemented all algorithms in Python 3.9 and performed simulations on Google Colab Pro+. We modeled the UAVs using the Crazyflie quadrotor and used CrazySwarm to simulate their behavior. From this, we selected a sensor for the mission and UAV, meticulously ensuring that the parameters reflected real-world conditions. For our experiments, we used the Crazyflie drone equipped with an ultrasonic sensor (LV-MaxSonar-EZ2), capable of detecting objects within a range of 0–254 inches (~6.75 m), as used in previous work [45]. The Crazyflie UAV is shown in Figure 2 and the ultrasonic sensor is displayed in Figure 3.



**Figure 2.** Crazyflie UAV.



**Figure 3.** Ultrasonic sensor (LV-MaxSonar-EZ2).

We also considered Crazyflie's properties [46,47] to evaluate the UAVs' energy consumption. Table 7 gives a summary of what UAV energy parameters involve.

**Table 7.** Energy-related parameters.

Parameter	Value
Speed	1 m/s
$\delta_m$	2430 J
$\mu$	5.8 J/s

The corrosion defects dataset, originally introduced in [12], was utilized to evaluate the performance of the proposed GA for the inspection mission. This dataset was specifically designed to address the lack of publicly available datasets for indoor steel pipe inspection, as highlighted in [12]. It consists of 1300 samples of corrosion defects derived from a realistic fire sprinkler system template. This template accurately simulates the typical layout and complexity of real-world sprinkler systems, ensuring the relevance of the dataset to practical inspection scenarios.

To create the dataset, the sprinkler system template was preprocessed into an occupancy matrix, a grid-based representation commonly used in robotics and path planning [48]. Corrosion defects were then distributed across the map using a polar coordinate method to simulate hotspots, mimicking the clustered and unpredictable nature of corrosion in real-world pipelines. The severity of corrosion varied from 1% to 30% of the pipe's surface area. These severity levels represent the percentage of the pipe's surface covered by corrosion, ranging from minor surface rust (1%) to significant structural damage (30%). This range reflects the diverse levels of damage typically encountered in real-world pipeline inspections. For this study, we used map#1 and map#2, representing two different configurations of the sprinkler system. We used instance#1 for each map and each severity level (simple, average, and advanced). This dataset, with its realistic representation of corrosion defects and pipeline layouts, provides a robust platform for assessing the effectiveness of the proposed GA in optimizing multi-UAV inspection missions.

The dataset consists of 1300 samples of corrosion defects and was employed in various inspection scenarios to assess the optimization of drone path planning within the CrazySwarm platform. We used map#1 and map#2, which were based on real-life pipeline systems. We used instance#1 for each map for each severity level (simple, average, and advanced).

For this study, we use performance metrics obtained from recent work to evaluate the effectiveness of our method. Example metrics are the total cost/fitness value (which corresponds to distance traveled), the maximum tour length, the running time, and average energy consumption. All scenario executions were performed 30 times to lessen the variability in performance measurements [49].

In this study, we specifically focus on evaluating the performance of the newly introduced GA and HC algorithms. We plot them against random and greedy algorithms.

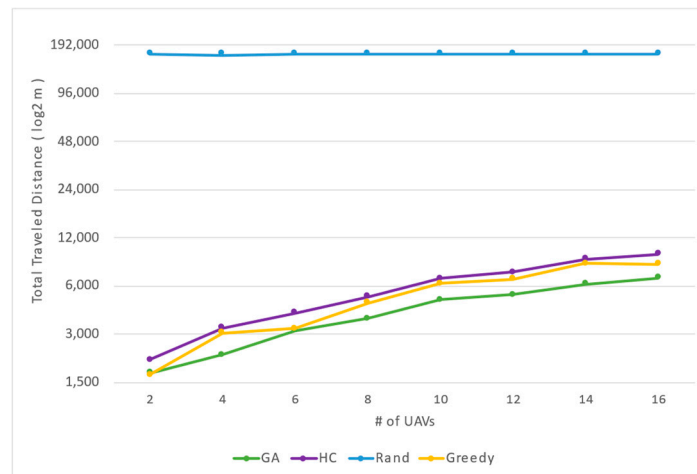
The Greedy Algorithm for UAV path planning operates by having each UAV select the nearest unvisited location from its current position. Starting from an initial point, the UAV continuously chooses the closest unvisited location, typically using Euclidean distance, until all locations are covered. On the other hand, the Random Strategy for UAV path planning involves each UAV selecting its following location randomly from the set of unvisited locations. Starting from an initial point, the UAVs make decisions without considering distance or other optimization criteria, resulting in paths that vary significantly in length and efficiency. In all our simulations, we control the number of UAVs, which varies between 2, 4, 6, 8, 10, 12, 14, and 16.

## 5. Results and Discussion

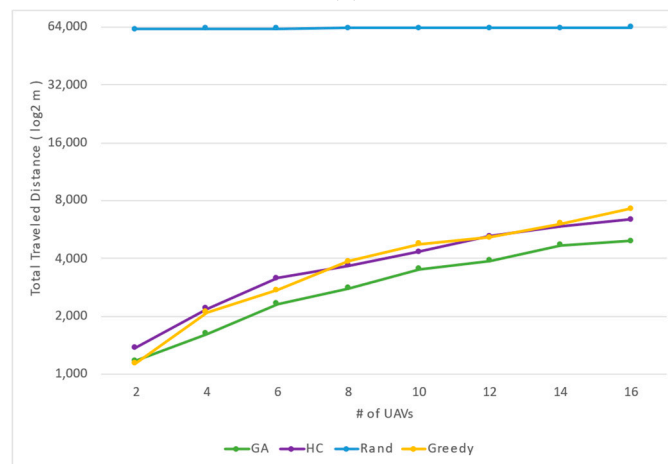
We conducted at least 30 trials for each experiment and plotted the results with a linear  $x$ -axis for the number of UAVs and a logarithmic base-2  $y$ -axis to represent performance metrics. Performance Indicators were the total traveled distance, the maximum tour length, execution time per mission, and average energy consumption, shown in Figure 4, Figure 5, Figure 6, and Figure 7, respectively. For Map#1, with 1785 locations, and Map #2 (996 locations), the experimental results notice common characteristics among different performance measures that are discussed below.

### 5.1. Total Traveled Distance

The total traveled distance increases with the number of UAVs for all algorithms. Nonetheless, GA consistently achieves lower total traveled distances than HC, the Greedy Algorithm, and the Random Strategy. This result fulfills the objective of optimizing UAV paths, as GA demonstrates a superior ability to minimize traveled distance. The Greedy Algorithm, while better than the Random Strategy, performs worse than GA because it makes locally optimal choices that do not always lead to globally optimal solutions. As expected, the Random strategy has the highest total traveled distances due to its lack of optimization criteria [30].



(a)



(b)

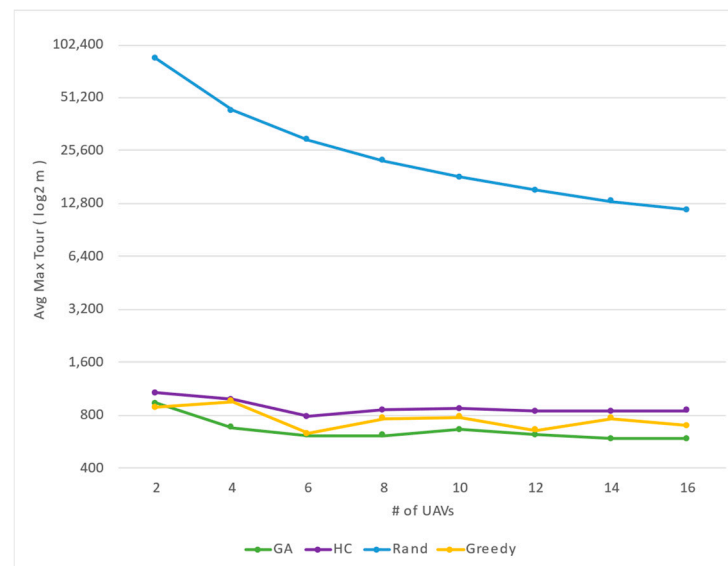
**Figure 4.** Total traveled distance across algorithms: (a) Map#1; (b) Map#2, showing mean values calculated over 30 runs.

### 5.2. Max Tour Length

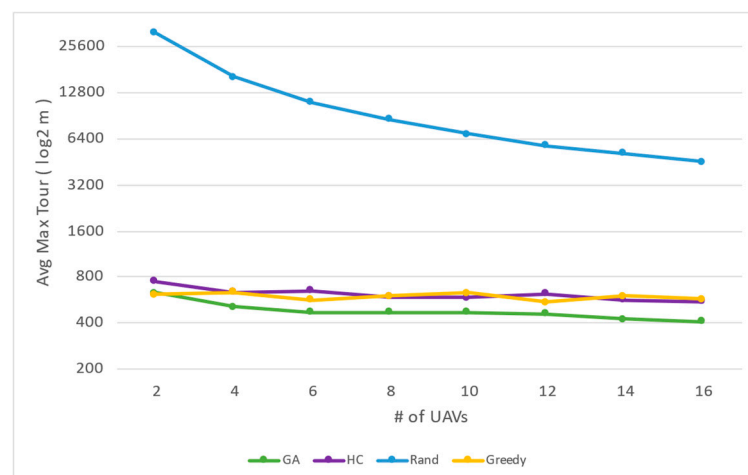
Only GA yields modestly maximum tour durations (the furthest any single UAV traverses) better than all the other benchmarks. It limits the work for each UAV so that no single UAV is overloaded. The Greedy strategy, which performed better than HC and the Random start, also produced a higher maximum tour than GA.

The GA application gives a more balanced path distribution. This may be because GA performs better than HC in finding the optimal path because it can explore a broad range of solutions and, therefore, avoid the local optimal [50]. A weakness of the HC implementation is that it often became caught in local optima and led to longer individual tours.

A notable observation is that HC's implementation, despite using a method to manipulate tour lengths by reallocating locations between tours, the max tour length of the HC performance remains steady. This shows the importance of the penalty mechanism introduced to HC, as described in our objectives, in maintaining more balanced solutions. The random strategy, as expected, performs the worst in balancing the tour lengths.



(a)



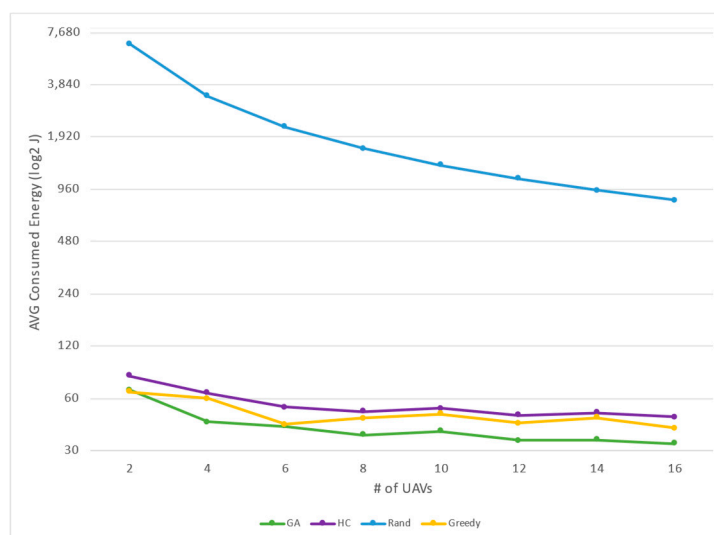
(b)

**Figure 5.** Maximum tour length across algorithms: (a) Map#1; (b) Map#2, showing mean values calculated over 30 runs.

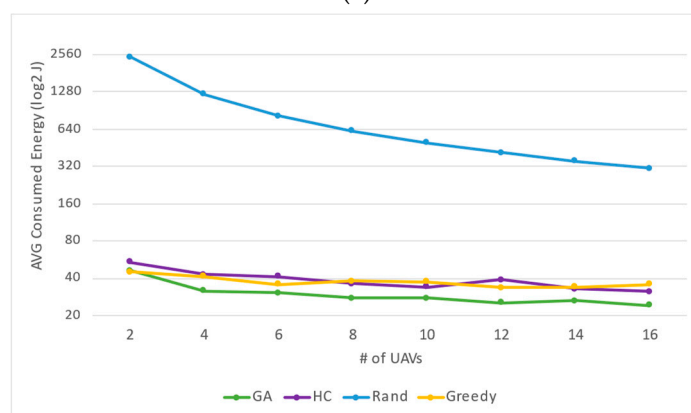


### 5.3. Average Energy Consumption per UAV

GA minimizes average energy consumption per UAV, achieving higher path optimality by avoiding unnecessary flights [50]. The Greedy Algorithm performs moderately better than HC and the Random Strategy but still falls short of GA. HC's average energy consumption is higher and more variable due to its inefficiency in avoiding local optima. With its random path selections, the Random Strategy results in the highest and most inconsistent energy consumption. These findings align with our study's objective, showing GA's superior performance across multiple performance metrics, including energy efficiency.



(a)



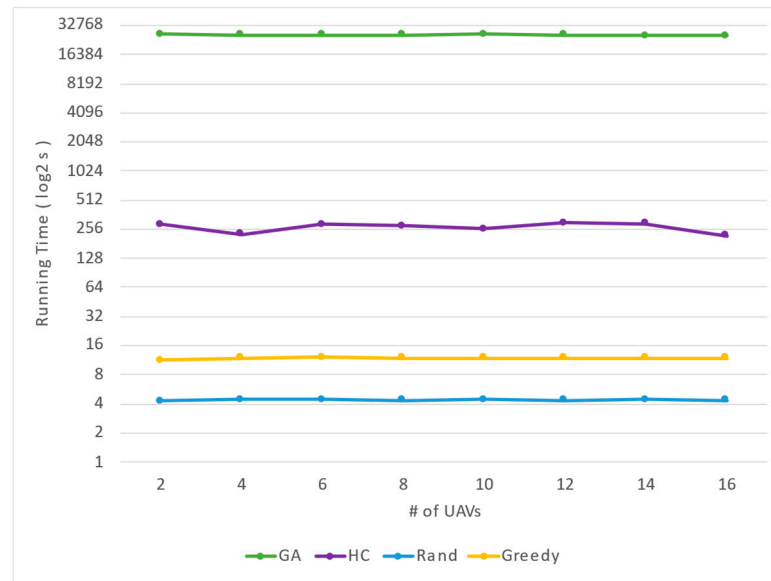
(b)

**Figure 6.** Average consumed energy across algorithms: (a) Map#1; (b) Map#2, showing mean values calculated over 30 runs.

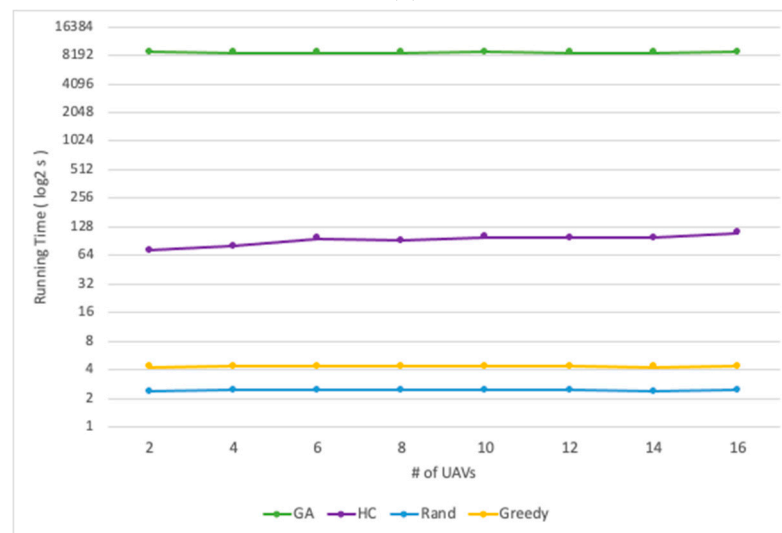
### 5.4. Running Time

The computational complexity of HC remains relatively stable as the number of UAVs increases, showing lower computation costs compared to GA. Although GA is a robust heuristic algorithm that excels in path planning for multi-UAV systems, it requires significantly more computational time [14,24,30]. This is particularly noticeable in larger maps (Map#1), where path optimization becomes more computationally intensive. Although HC terminates faster, its early convergence due to getting trapped in local optima often leads to inconsistent optimization results [51]. In contrast, GA's use of crossover and mutation allows for a more thorough exploration of the solution space, maintaining diversity in solu-

tions and delivering more consistent and optimized results over time. The GA technique helps progress steadily and gives more consistent optimization results than HC [52].



(a)



(b)

**Figure 7.** Running time across algorithms: (a) Map#1; (b) Map#2, showing mean values calculated over 30 runs.

This analysis shows that GA excels in energy efficiency and path optimization, especially in scenarios where computational overhead is not the primary concern. The use of GA demonstrates superior performance in optimizing UAV paths, balancing workloads, and minimizing energy consumption. These strengths validate its suitability for complex multi-UAV path planning tasks, making it a candidate when high-quality optimization is essential.

GA demonstrates its scalability and robustness, as observed in various map sizes or the number of UAVs. In addition, the crossover and mutation by GA help maintain solution diversity, preventing premature solution convergence. Thus, it ensures a more thorough exploration of the search space. Maintaining solution diversity is especially useful in dynamic environments, wherein the problem landscape might change with time. These properties render GA a robust tool in current path-planning tasks and make it a proven

choice for more complex multi-UAV operations. As the computational cost is negligible compared to its gains in optimization quality, this trade-off fully justifies GA as a candidate of choice over other techniques for application in realistic conditions where both efficiency and effectiveness are essential.

While GA consistently outperforms HC regarding path optimization, energy consumption, and solution robustness, HC still has its merits in specific contexts. For instance, HC demonstrates a faster runtime and lower computational cost, particularly in smaller map sizes or scenarios where computational efficiency is prioritized over solution quality. These characteristics make HC a viable option in simple environments where the problem size and complexity are limited. However, this efficiency diminishes as the problem scale increases, revealing HC's inherent limitations.

Scalability remains a critical consideration for both algorithms in more complex environments. While HC is efficient for small-scale problems, its localized search mechanism becomes computationally expensive in larger scenarios, and its dependence on the initial solution increases the likelihood of less effective results in high-dimensional solution spaces. Similarly, GA, despite its robustness and scalability, requires substantial computational resources to maintain population diversity and handle larger generations, which can pose challenges for its application in large-scale real-world scenarios.

Additionally, transitioning these algorithms to real-world applications presents several challenges, such as sensor inaccuracies due to environmental factors like temperature and humidity variations, which affect the speed of sound, and surface properties of objects, like irregular or absorptive materials, which can weaken or misdirect the reflected signal. These observations underscore the need for scalable algorithmic enhancements to ensure their adaptability to increasingly complex multi-UAV inspection missions and real-world environments.

Moreover, coordinating multiple UAVs in dynamic environments requires robust communication systems. Issues such as latency, signal interference, and network congestion could impair coordination, especially in large-scale missions. Implementing decentralized communication protocols or fault-tolerant strategies could improve system resilience. Furthermore, real-world environments are subject to unpredictable changes, such as weather conditions or moving obstacles, necessitating the development of adaptive path-planning algorithms that can dynamically respond to these changes and ensure mission continuity. These considerations highlight the importance of future work to validate our algorithms in real-world conditions and explore enhancements to address these practical challenges.

To further substantiate the robustness of our results, we calculated the Confidence Intervals (CIs) for each performance metric across all experiments. The CIs are represented in the format "Mean  $\pm$  MOE," where MOE (Margin of Error) is derived based on the variability in the results at a 95% confidence level. This representation provides a clear indication of the uncertainty associated with the reported mean values, offering valuable insights into the consistency and reliability of the algorithms under different scenarios. The CI results for the total cost, maximum tour length, and average energy consumption are summarized in Tables 8 and 9 for map#1 and map#2, respectively. These tables reinforce the analysis of the performance metrics presented earlier.

The inclusion of MOE highlights the stability of GA and HC, with GA exhibiting tighter CIs, indicating greater reliability in its performance metrics. For Greedy, zero variability (MOE = 0) aligns with its deterministic behavior, whereas for Random, a higher MOE is expected due to its stochastic nature. This detailed CI analysis underscores the robustness of GA and HC, as well as the predictable and variable characteristics of Greedy and Random algorithms, respectively.

**Table 8.** Confidence Intervals (CI) for Map#1 across performance measures (Total Cost, Max Tour Length, and Avg Consumed Energy) for algorithms (GA, HC, Greedy, and Random) with varying UAVs.

Performance Measure	#UAVs	Random Algorithm	Greedy Algorithm	Genetic Algorithm	Hill Climbing Algorithm
Running Time	2	4.2368 ± 0.0009	11.2477 ± 0.0006	26,525.7415 ± 2.3460	285.5000 ± 0.2892
	4	4.3919 ± 0.0003	11.8995 ± 0.0005	26,392.6020 ± 6.8387	225.9625 ± 0.0278
	6	4.4072 ± 0.0003	12.0419 ± 0.0010	26,226.7910 ± 2.7771	286.1125 ± 0.3214
	8	4.3631 ± 0.0010	11.9587 ± 0.0010	26,328.6515 ± 3.5733	274.7678 ± 0.2702
	10	4.4035 ± 0.0005	12.0037 ± 0.0012	26,439.6750 ± 3.3961	258.2400 ± 0.1844
	12	4.3625 ± 0.0011	11.9776 ± 0.0023	26,381.7405 ± 1.9944	297.5850 ± 0.2543
	14	4.3945 ± 0.0004	11.9866 ± 0.0016	25,809.1295 ± 2.2412	293.0946 ± 0.3242
	16	4.3595 ± 0.0010	12.0160 ± 0.0011	25,672.7115 ± 6.8501	218.7625 ± 0.0093
Total Cost	2	168,879.84 ± 739.03	1680.03 ± 0.00	1720.86 ± 84.39	2076.89 ± 293.92
	4	168,611.57 ± 773.37	3070.13 ± 0.00	2236.27 ± 295.71	3300.75 ± 60.49
	6	168,927.01 ± 696.49	3263.52 ± 0.00	3170.09 ± 222.96	4105.14 ± 161.66
	8	169,032.47 ± 795.13	4718.36 ± 0.00	3771.74 ± 186.21	5163.81 ± 264.72
	10	169,005.85 ± 708.78	6229.98 ± 0.00	4958.38 ± 343.22	6727.07 ± 539.13
	12	169,752.68 ± 646.68	6629.35 ± 0.00	5294.08 ± 322.36	7353.05 ± 539.18
	14	169,315.10 ± 698.07	8321.55 ± 0.00	6185.52 ± 540.74	8859.71 ± 499.18
	16	169,850.51 ± 695.18	8273.02 ± 0.00	6751.72 ± 430.67	9560.68 ± 245.21
Max Tour Length	2	85,142.34 ± 425.38	886.00 ± 0.00	932.90 ± 49.74	1073.54 ± 160.03
	4	43,504.05 ± 301.62	955.69 ± 0.00	682.36 ± 98.84	984.98 ± 64.32
	6	29,351.65 ± 190.57	628.20 ± 0.00	613.10 ± 52.34	783.67 ± 176.03
	8	22,322.10 ± 173.71	765.59 ± 0.00	612.51 ± 17.23	855.75 ± 141.95
	10	18,096.78 ± 144.78	782.40 ± 0.00	660.27 ± 83.77	873.80 ± 99.37
	12	15,222.27 ± 145.60	659.20 ± 0.00	619.34 ± 36.60	840.69 ± 90.98
	14	13,187.58 ± 99.60	766.85 ± 0.00	587.59 ± 48.18	843.94 ± 70.24
	16	11,753.79 ± 139.32	696.28 ± 0.00	585.66 ± 40.87	846.88 ± 74.80
Avg consumed Energy	2	6611.65 ± 28.93	65.77 ± 0.00	67.37 ± 3.30	81.31 ± 11.51
	4	3300.57 ± 15.14	60.10 ± 0.00	43.78 ± 5.79	64.61 ± 1.18
	6	2204.50 ± 9.09	42.59 ± 0.00	41.37 ± 2.91	59.13 ± 4.16
	8	1654.41 ± 7.78	46.18 ± 0.00	36.92 ± 1.82	50.54 ± 8.22
	10	1323.32 ± 5.55	48.78 ± 0.00	38.82 ± 4.69	53.25 ± 6.61
	12	1107.64 ± 4.22	43.26 ± 0.00	34.54 ± 2.10	45.15 ± 2.29
	14	946.96 ± 3.90	46.54 ± 0.00	34.59 ± 3.67	48.53 ± 1.17
	16	831.21 ± 3.40	40.49 ± 0.00	33.04 ± 4.52	49.01 ± 4.07

**Table 9.** Confidence Intervals (CI) for Map#2 across performance measures (Total Cost, Max Tour Length, and Avg Consumed Energy) for algorithms (GA, HC, Greedy, and Random) with varying UAVs.

Performance Measure	#UAVs	Random Algorithm	Greedy Algorithm	Genetic Algorithm	Hill Climbing Algorithm
Running Time	2	2.3897 ± 0.0009	4.2910 ± 0.0006	8799.4350 ± 2.3460	72.4250 ± 0.2892
	4	2.4477 ± 0.0003	4.3827 ± 0.0005	8777.0790 ± 6.8387	80.2750 ± 0.0278
	6	2.4513 ± 0.0003	4.3817 ± 0.0010	8738.5140 ± 2.7771	96.6589 ± 0.3214
	8	2.4421 ± 0.0010	4.3850 ± 0.0010	8667.7805 ± 3.5733	92.4875 ± 0.2702
	10	2.4413 ± 0.0005	4.3825 ± 0.0012	8833.7792 ± 3.3961	100.5500 ± 0.1844
	12	2.4459 ± 0.0011	4.3737 ± 0.0023	8707.0840 ± 1.9944	97.6500 ± 0.1665
	14	2.3747 ± 0.0004	4.2675 ± 0.0016	8740.5770 ± 2.2412	98.5500 ± 0.3242
	16	2.4446 ± 0.0010	4.3898 ± 0.0011	8786.2005 ± 6.8501	109.6219 ± 0.0093
Total Cost	2	62,320.87 ± 739.03	1142.63 ± 0.00	1178.01 ± 46.64	1378.11 ± 66.40
	4	62,572.60 ± 773.37	2102.51 ± 0.00	1618.68 ± 166.61	2197.20 ± 272.45
	6	62,806.81 ± 696.49	2734.81 ± 0.00	2330.71 ± 186.50	3154.82 ± 442.40
	8	63,021.98 ± 795.13	3861.83 ± 0.00	2808.42 ± 182.34	3677.63 ± 377.96
	10	63,056.04 ± 708.78	4770.64 ± 0.00	3529.25 ± 163.20	4346.53 ± 677.32
	12	63,045.09 ± 646.68	5145.81 ± 0.00	3494.82 ± 236.37	5201.45 ± 570.51
	14	63,191.68 ± 698.07	6055.33 ± 0.00	4936.81 ± 200.75	5879.40 ± 663.79
	16	63,394.84 ± 695.18	7241.23 ± 0.00	4473.31 ± 239.03	6390.34 ± 580.70
Max Tour Length	2	31,679.04 ± 425.38	614.81 ± 0.00	629.74 ± 45.73	748.06 ± 23.45
	4	16,245.85 ± 301.62	636.65 ± 0.00	512.53 ± 60.19	637.11 ± 46.48
	6	11,154.48 ± 190.57	564.26 ± 0.00	470.27 ± 34.73	649.31 ± 81.51
	8	8576.03 ± 173.71	599.32 ± 0.00	470.17 ± 30.30	595.96 ± 64.31
	10	6896.86 ± 144.78	628.52 ± 0.00	472.95 ± 5.10	587.33 ± 44.48

Table 9. Cont.

Performance Measure	#UAVs	Random Algorithm	Greedy Algorithm	Genetic Algorithm	Hill Climbing Algorithm
Avg consumed Energy	12	5786.41 ± 145.60	544.65 ± 0.00	461.49 ± 43.00	618.58 ± 68.10
	14	5128.19 ± 99.60	598.99 ± 0.00	425.23 ± 25.45	564.08 ± 85.53
	16	4519.97 ± 139.32	576.18 ± 0.00	407.42 ± 51.54	552.16 ± 99.44
	2	2439.86 ± 57.87	44.73 ± 0.00	46.12 ± 1.83	53.95 ± 2.60
	4	1224.86 ± 60.55	41.16 ± 0.00	31.69 ± 3.26	43.01 ± 3.62
	6	819.63 ± 54.54	35.69 ± 0.00	30.42 ± 2.43	41.17 ± 5.77
	8	616.83 ± 62.26	37.80 ± 0.00	27.49 ± 1.78	35.99 ± 8.03
	10	493.73 ± 55.50	37.35 ± 0.00	27.63 ± 1.28	34.03 ± 2.41
	12	411.37 ± 50.64	33.58 ± 0.00	25.41 ± 4.10	38.81 ± 4.50
	14	353.42 ± 54.66	33.87 ± 0.00	26.21 ± 2.46	32.88 ± 3.71
	16	310.24 ± 54.43	35.44 ± 0.00	24.09 ± 3.19	31.27 ± 5.09

## 6. Conclusions

This study presented the first application of a Genetic Algorithm (GA) for path planning in multi-UAV aerial inspection missions, comparing its performance to benchmarks. GA demonstrated improved performance compared to Hill Climbing (HC) for optimizing path planning in multi-UAV inspection missions.

In contrast, HC exhibited inconsistent running times due to frequent entrapment in local optima, leading to suboptimal performance. However, HC showed potential in scenarios where computational efficiency is more important than solution quality, particularly in smaller problem spaces or when quick, feasible solutions are sufficient.

This study highlights GA's key advantages in multi-UAV operations, particularly when optimizing path-planning tasks in complex and dynamic environments. GA excels in maintaining solution quality and adapting to changing conditions, making it a highly effective tool for multi-UAV path planning.

Future work should include a systematic sensitivity analysis of GA parameters to understand their impact on performance and robustness. This would provide insights into optimizing parameter settings and guide the development of adaptive mechanisms that dynamically adjust parameters. Exploring hybrid approaches that combine the strengths of GA and HC could lead to quicker solutions with improved overall quality. For example, HC could be used to refine solutions generated by GA, leveraging its efficiency in local search while addressing GA's higher computational overhead. This approach aligns with previous studies that have demonstrated the effectiveness of hybrid GA-HC techniques in improving convergence speed and solution quality. For instance, hybrid GA-HC approaches have been shown to outperform standalone GA or HC in timetabling problems by leveraging HC's capability to refine local solutions, thus avoiding local optima and achieving fast convergence. The hybrid algorithm GA\_HC in the literature [53,54] highlights that the hybrid GA-HC algorithm supports the idea that combining GA's global search capabilities with HC's local search efficiency results in better solutions than using either algorithm alone.

To further validate these strategies, it would be beneficial to test hybrid GA-HC algorithms in real-world field operations under diverse conditions. The proposed hybridization strategies aim to improve computational efficiency and robustness in dynamic multi-UAV inspection missions, where adaptability to changing environmental conditions and task requirements is crucial. However, real-world scenarios may present additional challenges, such as UAVs depleting their battery power during missions or encountering unexpected obstacles. To address these edge cases, future work could incorporate dynamic energy constraints, enabling UAVs to autonomously return to charging stations when battery levels are critically low. Furthermore, integrating real-time obstacle detection and replanning mechanisms would enhance the system's robustness by allowing UAVs to dynamically adjust their paths in response to unforeseen impediments. Additionally, flying intruders,

such as birds entering the facility, and maintenance equipment, such as scaffolding, could disrupt operations, requiring UAVs to dynamically navigate around these unexpected obstacles to ensure mission continuity and safety. Emerging advanced methods in machine learning could further enhance UAV mission performance and adaptability. Incorporating reinforcement learning and neural networks presents exciting opportunities for innovation. These methods could enable UAVs to adapt to unforeseen changes and optimize mission performance in complex, dynamic environments. RL's ability to learn optimal strategies through interaction with dynamic environments could enable UAVs to adapt to unforeseen changes, such as moving obstacles or variable weather conditions. Meanwhile, neural networks excel in processing complex, large-scale data, such as sensor inputs or high-dimensional state spaces, to make real-time decisions and predictions. Integrating these advanced methods could further optimize mission performance, ensuring scalability, adaptability, and enhanced energy efficiency in real-world UAV operations.

Finally, to further enhance the dataset and minimize potential biases, future work will focus on incorporating a wider range of environmental conditions and corrosion types. This includes simulating factors like temperature, humidity, and soil acidity, as well as introducing various forms of corrosion, such as pitting, crevice, and stress corrosion cracking.

**Author Contributions:** Conceptualization, F.A. and Y.A.; Formal analysis, F.A.; Methodology, F.A., and Y.A.; Supervision, Y.A.; Writing—original draft, F.A.; Writing—review and editing, Y.A. All authors have read and agreed to the published version of the manuscript.

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

**Data Availability Statement:** The data presented in this study are openly available <https://doi.org/10.5281/zenodo.10809418>.

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

## References

1. Skorobogatov, G.; Barrado, C.; Salamí, E. Multiple UAV Systems: A Survey. *Unmanned Syst.* **2020**, *8*, 149–169. [CrossRef]
2. Elmeseiry, N.; Alshaer, N.; Ismail, T. A Detailed Survey and Future Directions of Unmanned Aerial Vehicles (UAVs) with Potential Applications. *Aerospace* **2021**, *8*, 363. [CrossRef]
3. Lee, A.J.; Song, W.; Yu, B.; Choi, D.; Tirtawardhana, C.; Myung, H. Survey of robotics technologies for civil infrastructure inspection. *J. Infrastruct. Intell. Resil.* **2023**, *2*, 100018. [CrossRef]
4. Ahmed, F.; Mohanta, J.C.; Keshari, A.; Yadav, P.S. Recent Advances in Unmanned Aerial Vehicles: A Review. *Arab. J. Sci. Eng.* **2022**, *47*, 7963–7984. [CrossRef]
5. Shakhathreh, H.; Sawalmeh, A.H.; Al-Fuqaha, A.; Dou, Z.; Almaita, E.; Khalil, I.; Othman, N.S.; Khreishah, A.; Guizani, M. Unmanned Aerial Vehicles (UAVs): A Survey on Civil Applications and Key Research Challenges. *IEEE Access* **2019**, *7*, 48572–48634. [CrossRef]
6. Macaulay, M.O.; Shafiee, M. Machine learning techniques for robotic and autonomous inspection of mechanical systems and civil infrastructure. *Auton. Intell. Syst.* **2022**, *2*, 8. [CrossRef]
7. Mohsan, S.A.H.; Khan, M.A.; Noor, F.; Ullah, I.; Alsharif, M.H. Towards the Unmanned Aerial Vehicles (UAVs): A Comprehensive Review. *Drones* **2022**, *6*, 147. [CrossRef]
8. Sprinkler Standards. The European Fire Sprinkler Network (EFSN). Available online: <https://www.eurosprinkler.org/sprinkler-standards-2/> (accessed on 16 July 2024).
9. NFPA 13: Sprinkler System Design & Installation Fundamentals. Digitize. Available online: <https://www.digitize-inc.com/blog/nfpa-13-sprinkler-system-installations.php> (accessed on 16 July 2024).
10. Zhang, H.; Xin, B.; Dou, L.; Chen, J.; Hirota, K. A review of cooperative path planning of an unmanned aerial vehicle group. *Front. Inform. Technol. Elect. Eng.* **2020**, *21*, 1671–1694. [CrossRef]
11. Sharma, A.; Vanjani, P.; Paliwal, N.; Basnayaka, C.M.; Jayakody, D.N.K.; Wang, H.-C.; Muthuchidambaranathan, P. Communication and networking technologies for UAVs: A survey. *J. Netw. Comput. Appl.* **2020**, *168*, 102739. [CrossRef]
12. Aljalaud, F.; Alohal, Y. Enhancing Inspection Tasks: A Dataset for Corrosion Defects in Pipelines. *Indones. J. Comput. Sci.* **2024**, *13*, 12. [CrossRef]



13. Albadr, M.A.; Tiun, S.; Ayob, M.; AL-Dhief, F. Genetic Algorithm Based on Natural Selection Theory for Optimization Problems. *Symmetry* **2020**, *12*, 1758. [[CrossRef](#)]
14. Agrawal, S.; Patle, B.K.; Sanap, S. A systematic review on metaheuristic approaches for autonomous path planning of unmanned aerial vehicles. *Drone Syst. Appl.* **2024**, *12*, 1–28. [[CrossRef](#)]
15. Zheng, H.; Kong, L.X.; Nahavandi, S. Automatic inspection of metallic surface defects using genetic algorithms. *J. Mater. Process. Technol.* **2002**, *125*, 427–433. [[CrossRef](#)]
16. Haladuick, S.; Dann, M.R. Genetic Algorithm for Inspection and Maintenance Planning of Deteriorating Structural Systems: Application to Pressure Vessels. *Infrastructures* **2018**, *3*, 32. [[CrossRef](#)]
17. Moura, M.D.C.; Lins, I.D.; Droguett, E.L.; Soares, R.F.; Pascual, R. A Multi-Objective Genetic Algorithm for determining efficient Risk-Based Inspection programs. *Reliab. Eng. Syst. Saf.* **2015**, *133*, 253–265. [[CrossRef](#)]
18. Dedeurwaerder, B.; Louis, S.J. A Meta Heuristic Genetic Algorithm for Multi-Depot Routing in Autonomous Bridge Inspection. In Proceedings of the 2022 IEEE Symposium Series on Computational Intelligence (SSCI), Singapore, 4–7 December 2022; IEEE: New York, NY, USA, 2022; pp. 1194–1201. [[CrossRef](#)]
19. Cao, Y.; Wei, W.; Bai, Y.; Qiao, H. Multi-base multi-UAV cooperative reconnaissance path planning with genetic algorithm. *Clust. Comput.* **2019**, *22*, S5175–S5184. [[CrossRef](#)]
20. Russell, S.J.; Norvig, P. *Artificial Intelligence: A Modern Approach*, 3rd ed.; Prentice Hall Pearson: Upper Saddle River, NJ, USA, 2016.
21. Haghighi, H.; Sadati, S.H.; Dehghan, S.M.M.; Karimi, J. Hybrid Form of Particle Swarm Optimization and Genetic Algorithm For Optimal Path Planning in Coverage Mission by Cooperated Unmanned Aerial Vehicles. *J. Aerosp. Technol. Manag.* **2020**, *12*, e4320. [[CrossRef](#)]
22. Ali, Z.A.; Zhangang, H.; Zhengru, D. Path planning of multiple UAVs using MMACO and DE algorithm in dynamic environment. *Meas. Control* **2023**, *56*, 459–469. [[CrossRef](#)]
23. Wang, Y.; Shi, Y.; Liu, Y. Research on improved genetic simulated annealing algorithm for multi-UAV cooperative task allocation. *J. Phys. Conf. Ser.* **2022**, *2246*, 012081. [[CrossRef](#)]
24. Saadi, A.A.; Soukane, A.; Meraihi, Y.; Gabis, A.B.; Mirjalili, S.; Ramdane-Cherif, A. UAV Path Planning Using Optimization Approaches: A Survey. *Arch. Computat Methods Eng.* **2022**, *29*, 4233–4284. [[CrossRef](#)]
25. Israr, A.; Ali, Z.A.; Alkhamash, E.H.; Jussila, J.J. Optimization Methods Applied to Motion Planning of Unmanned Aerial Vehicles: A Review. *Drones* **2022**, *6*, 126. [[CrossRef](#)]
26. Aljalaud, F.; Kurdi, H.; Youcef-Toumi, K. Autonomous Multi-UAV Path Planning in Pipe Inspection Missions Based on Booby Behavior. *Mathematics* **2023**, *11*, 2092. [[CrossRef](#)]
27. Yan, F.; Zhu, X.; Zhou, Z.; Chu, J. A Hierarchical Mission Planning Method for Simultaneous Arrival of Multi-UAV Coalition. *Appl. Sci.* **2019**, *9*, 1986. [[CrossRef](#)]
28. Dewangan, R.K.; Shukla, A.; Godfrey, W.W. Three dimensional path planning using Grey wolf optimizer for UAVs. *Appl. Intell.* **2019**, *49*, 2201–2217. [[CrossRef](#)]
29. Yang, L.; Guo, J.; Liu, Y. Three-Dimensional Uav Cooperative Path Planning Based on the Mp-Cgwo Algorithm. *Int. J. Innov. Comp. Inf. Control* **2020**, *16*, 991–1006. [[CrossRef](#)]
30. Pan, Y.; Yang, Y.; Li, W. A Deep Learning Trained by Genetic Algorithm to Improve the Efficiency of Path Planning for Data Collection With Multi-UAV. *IEEE Access* **2021**, *9*, 7994–8005. [[CrossRef](#)]
31. Ahmed, N.; Pawase, C.J.; Chang, K. Distributed 3-D Path Planning for Multi-UAVs with Full Area Surveillance Based on Particle Swarm Optimization. *Appl. Sci.* **2021**, *11*, 3417. [[CrossRef](#)]
32. Teng, H.; Ahmad, I.; Alamgir, M.S.M.; Chang, K. 3D Optimal Surveillance Trajectory Planning for Multiple UAVs by Using Particle Swarm Optimization With Surveillance Area Priority. *IEEE Access* **2020**, *8*, 86316–86327. [[CrossRef](#)]
33. Strasser, S.; Goodman, R.; Sheppard, J.; Butcher, S. A New Discrete Particle Swarm Optimization Algorithm. In Proceedings of the Genetic and Evolutionary Computation Conference 2016, Denver, CO, USA, 20–24 July 2016; ACM: New York, NY, USA, 2016; pp. 53–60. [[CrossRef](#)]
34. Moghtadernejad, S.; Adey, B.T.; Hackl, J. Prioritizing Road Network Restorative Interventions Using a Discrete Particle Swarm Optimization. *J. Infrastruct. Syst.* **2022**, *28*, 04022039. [[CrossRef](#)]
35. Tosoni, D.; Galli, C.; Hanne, T.; Dornberger, R. Benchmarking Metaheuristic Optimization Algorithms on Travelling Salesman Problems. In Proceedings of the 2022 8th International Conference on e-Society, e-Learning and e-Technologies (ICSLT), Rome Italy, 10–12 June 2022; ACM: New York, NY, USA, 2022; pp. 20–25. [[CrossRef](#)]
36. Brown, E.C.; Ragsdale, C.T.; Carter, A.E. A grouping genetic algorithm for the multiple traveling salesperson problem. *Int. J. Info. Tech. Dec. Mak.* **2007**, *6*, 333–347. [[CrossRef](#)]
37. Ramos-Figueroa, O.; Quiroz-Castellanos, M.; Mezura-Montes, E.; Kharel, R. Variation Operators for Grouping Genetic Algorithms: A Review. *Swarm Evol. Comput.* **2021**, *60*, 100796. [[CrossRef](#)]
38. Majumdar, J.; Bhunia, A.K. Genetic algorithm for asymmetric traveling salesman problem with imprecise travel times. *J. Comput. Appl. Math.* **2011**, *235*, 3063–3078. [[CrossRef](#)]

39. Varadarajan, S.; Whitley, D. The massively parallel mixing genetic algorithm for the traveling salesman problem. In Proceedings of the Genetic and Evolutionary Computation Conference, Prague Czech Republic, 13–17 July 2019; ACM: New York, NY, USA, 2019; pp. 872–879. [[CrossRef](#)]
40. Villegas, L.M.B.; Morales, S.O.C. Studying the effect of Eliminating Repeated Individuals from the Population in a Genetic Algorithm: Solution Perspectives for the Travelling Salesman Problem. *J. Eng. Res. Rep.* **2021**, *20*, 97–102. [[CrossRef](#)]
41. Ahmed, Z.H. A multi-parent genetic algorithm for the quadratic assignment problem. *Opsearch* **2015**, *52*, 714–732. [[CrossRef](#)]
42. Hassanat, A.; Almohammadi, K.; Alkafaween, E.; Abunawas, E.; Hammouri, A.; Prasath, V.B.S. Choosing Mutation and Crossover Ratios for Genetic Algorithms—A Review with a New Dynamic Approach. *Information* **2019**, *10*, 390. [[CrossRef](#)]
43. Preiss, J.A.; Honig, W.; Sukhatme, G.S.; Ayanian, N. CrazySwarm: A large nano-quadcopter swarm. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; IEEE: New York, NY, USA, 2017; pp. 3299–3304. [[CrossRef](#)]
44. Official CrazySwarm Tutorial. Available online: <https://crazyswarm.readthedocs.io/en/latest/tutorials/tutorials.html> (accessed on 12 November 2022).
45. Helland, J.; Whitaker, J.; Cowan, P.; Glass, S. *Autonomous Drone*; University of Utah Abstract: Salt Lake City, UT, USA, 2015. Available online: <https://my.ece.utah.edu/~kstevens/3992/reports/death-ray.pdf> (accessed on 23 November 2022).
46. Datasheet Crazyflie 2.1. Available online: [https://www.bitcraze.io/documentation/hardware/crazyflie\\_2\\_1/crazyflie\\_2\\_1-datasheet.pdf](https://www.bitcraze.io/documentation/hardware/crazyflie_2_1/crazyflie_2_1-datasheet.pdf) (accessed on 22 February 2023).
47. Battery and Charger for Crazyflie 2.1 Drone. Available online: <https://www.generationrobots.com/en/403752-240-mah-battery-and-charger-for-crazyflie-21-drone.html> (accessed on 22 February 2023).
48. Ren, Y.; Cai, Y.; Zhu, F.; Liang, S.; Zhang, F. ROG-Map: An Efficient Robocentric Occupancy Grid Map for Large-scene and High-resolution LiDAR-based Motion Planning. *arXiv* **2013**, arXiv:2302.14819. [[CrossRef](#)]
49. Chen, X.; Zhang, P.; Du, G.; Li, F. Ant Colony Optimization Based Memetic Algorithm to Solve Bi-Objective Multiple Traveling Salesmen Problem for Multi-Robot Systems. *IEEE Access* **2018**, *6*, 21745–21757. [[CrossRef](#)]
50. Tang, H.; Zhang, X.; Miao, C.; Zhang, J.; Ming, R.; Schnable, J.C.; Schnable, P.S.; Lyons, E.; Lu, J. ALLMAPS: Robust scaffold ordering based on multiple maps. *Genome Biol.* **2015**, *16*, 3. [[CrossRef](#)] [[PubMed](#)]
51. Al-Betar, M.A.  $\beta$ -hill climbing: An exploratory local search. *Neural Comput. Appl.* **2017**, *28*, 153–168. [[CrossRef](#)]
52. Katoch, S.; Chauhan, S.S.; Kumar, V. A review on genetic algorithm: Past, present, and future. *Multimed. Tools Appl.* **2021**, *80*, 8091–8126. [[CrossRef](#)] [[PubMed](#)]
53. Yusoff, M.; Roslan, N. Evaluation of Genetic Algorithm and Hybrid Genetic Algorithm-Hill Climbing with Elitist for Lecturer University Timetabling Problem. In *Advances in Swarm Intelligence*; Tan, Y., Shi, Y., Niu, B., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 363–373.
54. Hussain, A.; Ashas, H.; Shahid, A.; Qureshi, S.; Karrila, S. Hybrid Approach Involving Genetic Algorithm and Hill Climbing to Resolve the Timetable Scheduling for a University. In *Advances in Information and Communication*; Arai, K., Ed.; Springer International Publishing: Cham, Switzerland, 2024; pp. 72–83.

**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.