

Article

Payload- and Energy-Aware Tactical Allocation Loop-Based Path-Planning Algorithm for Urban Fumigation Robots

Prithvi Krishna Chittoor ^{1,*}, Bhanu Priya Dandumahanti ^{1,2}, Abishegan M. ^{1,3}, Sriniketh Konduri ¹,
S. M. Bhagya P. Samarakoon ¹ and Mohan Rajesh Elara ¹

¹ Engineering Product Development Pillar, Singapore University of Technology and Design, Singapore 487372, Singapore; dd1920@srmist.edu.in (B.P.D.); am2189@srmist.edu.in (A.M.); sriniketh_konduri@sutd.edu.sg (S.K.); bhagya_samarakoon@sutd.edu.sg (S.M.B.P.S.); rajeshelara@sutd.edu.sg (M.R.E.)

² Department of Mechanical Engineering, SRM Institute of Science and Technology, Chennai 603203, India

³ Department of Computer Science and Engineering, SRM Institute of Science and Technology, Tiruchirappalli 621105, India

* Correspondence: prithvi_chittoor@sutd.edu.sg

Abstract: Fumigation effectively manages pests, yet manual spraying poses long-term health risks to operators, making autonomous fumigation robots safer and more efficient. Path planning is a crucial aspect of deploying autonomous robots; it primarily focuses on minimizing energy consumption and maximizing operational time. The Payload and Energy-aware Tactical Allocation Loop (PETAL) algorithm integrates a genetic algorithm to search for waypoint permutations, applies a 2-OPT (two-edge exchange) local search to refine those routes, and leverages an energy cost function that reflects payload weight changes during spraying. This combined strategy minimizes travel distance and reduces energy consumption across extended fumigation missions. To evaluate its effectiveness, a comparative study was performed between PETAL and prominent algorithms such as A*, a hybrid Dijkstra with A*, random search, and a greedy distance-first approach, using both randomly generated environments and a real-time map from an actual deployment site. The PETAL algorithm consistently performed better than baseline algorithms in simulations, demonstrating significant savings in energy usage and distance traveled. On a randomly generated map, the PETAL algorithm achieved 6.05% higher energy efficiency and 23.58% shorter travel distance than the baseline path-planning algorithm. It achieved 15.69% and 31.66% in energy efficiency and distance traveled saved on a real-time map, respectively. Such improvements can diminish operator exposure, extend mission durations, and foster safer, more efficient urban pest control.

Keywords: A*; Dijkstra; fumigation; greedy distance first; path planning; pest control

MSC: 68T20



Received: 9 February 2025

Revised: 10 March 2025

Accepted: 11 March 2025

Published: 13 March 2025

Citation: Chittoor, P.K.; Dandumahanti, B.P.; M., A.; Konduri, S.; Samarakoon, S.M.B.P.; Elara, M.R. Payload- and Energy-Aware Tactical Allocation Loop-Based Path-Planning Algorithm for Urban Fumigation Robots. *Mathematics* **2025**, *13*, 950. <https://doi.org/10.3390/math13060950>

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

Fumigation is a crucial pest control method used in urban landscapes, industrial facilities, and healthcare environments to eradicate pests, like mosquitoes, using gaseous chemicals. These pests are carriers of life-threatening diseases like malaria, dengue, and Zika, posing public health concerns. Proper fumigation is essential to protect urban populations and maintain biodiversity. It promotes healthy plant growth and adherence to safety regulations, as highlighted by global health authorities and the World Health Organization [1,2]. Urban terrain environments are often burdened with dense populations,

irregular structural layouts, and a lack of green space maintenance. These conditions create ideal breeding grounds for pests and demand regular fumigation to maintain public health and ecological balance [3,4]. Manual urban fumigation, often used by trained workers, exposes workers to significant health risks, including respiratory problems, neurological disorders, skin irritation, and even cancer (due to prolonged exposure) [5]. Furthermore, it is labor-intensive and prone to human error, leading to uneven chemical distribution and missed pest breeding areas [5]. An autonomous fumigation robot must be implemented in urban landscapes to mitigate health risks associated with manual fumigation. It offers a safer alternative to manual labor in hazardous environments. Also, equipped with advanced navigation systems, these robots can precisely identify and target breeding sites in complex urban terrains, ensuring uniform and efficient chemical dispersion. Autonomous fumigation systems integrate sensors and real-time monitoring systems and can operate continuously with toxic chemicals, minimizing human health risks and improving operational safety by proactively detecting and addressing pest infestations.

Fumigation robots employ two main spraying methods: continuous and precision spraying. Continuous spraying ensures uniform chemical dispersion across large areas, while precision spraying targets specific high-risk zones for effective pest control. Agricultural fumigation primarily employs a continuous spraying technique, where fumigation chemicals are uniformly applied over large areas to protect crops from pests. This method ensures widespread coverage, making it effective for large-scale farming, but it may lead to chemical overuse and environmental concerns. In contrast, urban fumigation uses precision spraying techniques, targeting specific areas such as mosquito breeding sites. Many researchers emphasize precision spraying to optimize chemical management, reduce waste, and enhance fumigation operations' efficiency. Yu Xuan Chen et al. [6] developed unmanned ground vehicles (UGVs) to detect mosquito breeding sites in urban ditches, finding mosquitoes in 20.7% of inspected areas. UGVs combined with insecticide spraying showed effective control, highlighting the potential for improved mosquito monitoring and prevention. Sidharth Jeyabal et al. [7] introduced a fumigation robot for urban pest control, combining sensor fusion and YOLOv8 for precise hotspot detection. It improves efficiency by 62.5% and reduces chemical usage by using precision spraying techniques, offering a safer, more effective pest management solution. In continuation, Sriniketh Konduri et al. [8] introduced prioritized path planning for charging mosquito catchers using autonomous precision fumigation robots in urban landscapes with a fusion of genetic algorithm with A*-based navigation.

In autonomous navigation, urban fumigation robots must target specific locations, such as areas near dustbins, stagnant water zones, and dense greenery. These robots navigate efficiently using predefined waypoints, ensuring precise coverage of high-risk areas for effective pest control. Path planning is pivotal in the autonomous navigation of fumigation robots across urban landscape terrains, especially during precision fumigation applications. Choosing the optimized path is essential for extending the robot's operating time and minimizing the wastage of fumigation chemicals, ensuring efficient and effective pest control operations, particularly when dealing with large or complex urban landscapes. The frequent need to return to a base station to refill the fumigation chemical tank is a significant operational disadvantage, often resulting in wasted time, increased energy consumption, increased distance traveled by the robot, and interrupted workflows. Energy-aware and distance-efficient path planning is crucial for autonomous fumigation robots, which rely on battery power. These planning strategies minimize power usage by designing routes that minimize unnecessary movements and avoid sharp turns. This helps extend operational periods, reduces downtime, and minimizes repetitive trips to the base station for refills.

2. Related Studies

To address the challenges of optimized path planning while maximizing battery efficiency and minimizing chemical wastage, numerous researchers [9–13] have conducted extensive studies and investigations. Jiaming Leng et al. [14] introduced a data-driven model for estimating power consumption in low-speed autonomous guided vehicles (AGV), focusing on key components like the chassis, sensors, and communication devices. The model improves accuracy over traditional methods, achieving 12% energy savings in path planning and a 10.1% reduction in computational energy. With significant energy efficiency and battery performance improvements, this approach demonstrates the potential for optimizing robot operations through advanced power profiling techniques. Haojie Zhang et al. [15] proposed energy-efficient path planning to integrate with Ackermann steering and the A* algorithm and energy-aware motion primitives for AGVs. It effectively balanced path length and energy consumption with 26.9% energy savings. Dongpo Yang et al. [16] proposed a real-time multi-objective optimization-guided model predictive control strategy for power-split hybrid electric buses, optimizing fuel efficiency, shifting stability, and real-time control. This model enhanced efficiency and stability, achieving 98.41% fuel economy, a 12.5% reduction in shifting times, and 93.97% faster computation. Guoming Huang et al. [17] proposed a three-dimensional (3D) multi-object path planning method to optimize energy consumption and distance using an energy consumption estimation model integrated with a distance estimation model in electric vehicles. The proposed method achieved significant energy savings compared to traditional distance-based simulation methods. Huanwei Wang et al. [18] proposed an improved A*-based path planning method incorporating expansion distance, bidirectional search, heuristic optimization, and smoothing, which addresses collision avoidance and reduced right-angle turns. V Sangeetha et al. [19] proposed a gain-based dynamic green ant colony optimization algorithm for energy-efficient path planning in dynamic environments. Using a gain function and Octrees for 3D mapping improves energy use, path length, and computation time. Dongdong Li et al. [20] introduced an algorithm based on terminal distance index and multi-step movement and optimization, improving flexibility for mobile robot path planning. Simulations show faster convergence and shorter paths compared to traditional ant colony optimization methods. Junseo Lee et al. [21] proposed the adaptive analytic hierarchy process path planning by integrating the A* algorithm and robot sensor detection for optimized relative importance matrices. Simulations demonstrated the proposed method's efficiency in reducing distance and rotations compared to existing methods. Rafal Szczepanski et al. [22] proposed a predictive artificial potential field algorithm enhanced with energy-efficient path planning for AGVs by predicting movements and introducing virtual obstacles to avoid local minima. The algorithm reduces energy usage, path length, and travel time while ensuring smoother navigation. Chenming Wu et al. [23] address energy-efficient path planning for AGVs on 3D surfaces using height-aware Fermat spirals. Integrating geodesic distances and energy-efficient heuristics reduces gravity costs and demonstrates the proposed method's effectiveness on various terrains. Similarly, Mohammad Mohammadpour et al. [24] proposed local path planning for AGVs by incorporating load position effects on energy consumption. Using machine learning, it predicts the center of mass and energy requirements, generating smoother, shorter paths and reducing energy usage. Phone Tniha Kyaw et al. [25] proposed an energy-efficient batch-informed trees algorithm for reconfigurable robots in complex environments. The algorithm improves sampling techniques for faster convergence and accounts for energy costs of reconfiguration using the rapidly exploring random tree method. ShaoFei Shan et al. [26] proposed an optimized A* artificial potential field algorithm for trajectory planning in autonomous mobile robots with node

pruning, hybrid search, Bezier curve smoothing, and virtual ellipse theory incorporation to improve trajectory optimality, safety, and efficiency.

For effective navigation of autonomous robots and precise spraying tasks, several researchers have focused on dynamic localization techniques and obstacle detection methods to enable robots to adapt and operate efficiently in ever-changing dynamic environments. Yaxuan Yan et al. [27] presented a real-time localization and mapping system for agricultural robots in greenhouses, integrating wheel odometry, inertial measurement unit (IMU), and visual–inertial odometry with extended Kalman filter (EKF). Using modified simultaneous localization and mapping (SLAM) improves localization accuracy and reconstructs dense 3D maps in challenging environments. Similarly, K.S. Arikumar [28] used a fusion of 3D light detection and ranging (LiDAR) point cloud data and camera images from the computational neural network (CNN) for enhanced recognition of obstacles for AGVs. Qipeng Li et al. [29] presented a LiDAR–IMU localization system combining data and model fusion with uncertainty estimation, adapting to dynamic motion and environmental changes. The proposed algorithm demonstrates an 8.3% improvement in localization accuracy over existing methods. Likewise, Leyuan Liu et al. [30] integrated a deep neural network for LiDAR and camera fusion in 3D object detection and recognition, combining camera features with LiDAR point clouds. Yiyi Cai et al. [31] enhanced robot navigation using SLAM by integrating LiDAR-inertial odometry, visual-inertial odometry, and advanced IMU methods for robust mapping in dynamic environments. A lightweight object-detection network excludes transient objects, improving mapping accuracy and reliability in complex scenarios. On the other hand, Javier Mendez et al. [32] proposed a multi-level sensor fusion model optimized for the Google Coral tensor processing unit, combining it with LiDAR and camera data for target detection in autonomous vehicle navigation. It achieves high accuracy with reduced latency and power consumption. Yanjie Liu et al. [33] proposed an improved adaptive Monte Carlo localization method integrating 3D LiDAR, IMU, and odometer data using EKF for enhanced accuracy. Experiments show improved positioning accuracy and stability over traditional methods using point-to-line iterative closest-point refinement techniques.

Several challenges remain unaddressed with numerous advancements in autonomous navigation and fumigation robot technologies, including the following:

- Many studies focus on developing fumigation robots for agricultural applications, but there is a lack of research on fumigation robots designed for urban landscapes.
- Most agricultural fumigation robots follow straight-line paths on muddy terrains, whereas urban landscape fumigation robots require adaptive path planning to navigate in dynamic terrains, narrow junctions, and irregular pathways with dynamic obstacle avoidance, which remains unexplored in existing research.
- While energy-efficient path planning has been widely studied, there is minimal research on integrating minimal distance traveled with payload and energy-aware path planning for fumigation tasks.
- Existing studies fail to address the avoidance of redundant fumigation or strategies to minimize chemical wastage effectively.

To address these challenges comprehensively, the objectives of the current study are focused on the following:

1. Develop a payload and energy-aware tactical allocation loop (PETAL) algorithm that merges a genetic algorithm, a 2-OPT local search, and a payload-sensitive energy–distance cost model. This will enable autonomous navigation for fumigation robots under real-world constraints.
2. Design and integrate an energy–distance cost function that accurately reflects the robot’s changing payload (chemical usage), minimizing both overall distance traveled

and redundant spraying while ensuring efficient energy consumption over extended missions.

3. To implement the proposed algorithm on a randomly generated map and a map obtained from a 3D LiDAR scan, ensuring optimized and effective path planning for practical applications.

The key novelties of this study are as follows:

- **Energy- and Payload-Aware Synergy:** The manuscript introduces a path-planning framework that explicitly factors in the fumigation robot's chemical payload and battery limitations, ensuring extended operational time by continuously optimizing travel routes based on residual payload levels and energy usage.
- **Streamlined Base Station Cycles:** By strategically integrating chemical refilling and battery recharging into the navigation plan, the robot reduces idle time and travel overhead, resulting in more reliable, efficient missions and maximized mission time.

This manuscript focuses on developing an optimized path-planning algorithm for autonomous navigation in fumigation robots. Section 3 details the formulation and explanation of the proposed path-planning algorithm. Section 4 presents a case study demonstrating the random map generation with waypoints and deployment of the proposed algorithm. Section 5 highlights the development of the fumigation robot, with an emphasis on energy consumption analysis using the proposed algorithm in the real-time map and waypoints. Finally, Section 6 concludes the paper by summarizing the key findings and insights.

3. Distance and Energy Cost Model

This algorithm optimizes robot-based fumigation pathfinding and node selection within a structured area using binary maps generated from portable gray map images or grid-based maps with obstacles. The main focus is zone-based waypoint traversal, energy efficiency, and autonomous route planning with A* pathfinding while considering chemical weight constraints.

3.1. Problem Formulation

Consider a set of waypoints $W = \{0, 1, 2, \dots, N\}$. The waypoint 0 denotes a base station or refueling station, and the other waypoints $\{1, \dots, N\}$ represent target locations for a spraying robot to visit. The primary goal is to determine a route that minimizes a weighted combination of total energy and total distance traveled by the robot. Minimizing distance ensures shorter travel times and fewer redundant movements, while minimizing energy consumption is crucial for prolonging mission endurance, lowering operational costs, and reducing overall environmental impact. Additionally, payload constraints require the robot to return to the base station after servicing a limited number of waypoints before refilling is to be maintained (i.e., till the chemical tank is empty).

Let, $p = (p_1, p_2, \dots, p_N)$ be a permutation of the target waypoints $\{1, \dots, N\}$. Base station W_0 is inserted at the start of each mission segment and at the end of each segment for refilling, according to a specified limit on the number of waypoints that can be visited consecutively.

- The map is converted from a grayscale image to a binary map where the obstacles are marked as 1 and free space as 0, as presented in Equation (1).

$$\text{Binary_Map}(x, y) = \begin{cases} 1 & \text{if } \text{Pixel}(x, y) < T_{\text{obstacle}} \\ 0 & \text{if } \text{Pixel}(x, y) \geq T_{\text{obstacle}} \end{cases} \quad (1)$$

where T_{obstacle} is the obstacle threshold and Pixel (x, y) represents the pixel intensity at coordination (x, y) .

- Waypoints and Base Station, $W = \{0, 1, \dots, N\}$, $0 \in W$ = Base station and $p_i \in \{1, \dots, N\}$ = The i th waypoint in the visiting order.
- A fumigation limit F (for example, $F = 7$) is the maximum number of waypoints the robot can visit before returning to the base station for a refill. Hence, the permutation p is segmented into missions of size at most F . Each mission begins and ends at waypoint 0.
- Distance Function: Let $A_STAR(s, e)$ be a function returning a path of grid coordinates from waypoint s to waypoint e , determined via A^* in a binary map M as presented in Equation (2).
- If $A_STAR(s, e)$ is non-empty, we define the following:

$$d(s, e) = |A_STAR(s, e)| \times \text{UNIT_TO_METER} \quad (2)$$

That is the path length in meters (the number of A^* steps multiplied by a scaling factor determined experimentally).

Energy Consumption Model is given by the following:

- Travel Rate: Let $\rho(\text{payload_ml})$ be a linear function as presented in Equation (3). Also, ($\text{Payload_ml} = \text{Robot base weight} + \text{chemical payload}$). Then we obtain the following:

$$\rho(\text{payload_ml}) = a \times \text{payload_ml} + b \quad (3)$$

With constants a and b . Here, the $a = 0.0000020888$ and $b = 0.0000008853$ were determined experimentally by operating the robot during fully loaded and no-load conditions to evaluate a travel rate energy consumption function.

- Travel Energy: For a leg $(s \rightarrow e)$ with payload μ (in mL) as presented in Equation (4):

$$E_{\text{travel}}(s, e, \mu) = \rho(\mu) \times d(s, e) \quad (4)$$

- Tank Capacity: The maximum chemical volume the robot can carry.
- Spray Rate: The robot uses 660 mL of chemical per waypoint (considering 2 min of fumigation and 330 mL per minute of chemical dispersion). When the robot visits up to F waypoints, it depletes part of its payload. Upon finishing a segment, it must return to waypoint 0 to refill before continuing to the next segment.
- Cost Function: Let $\alpha \in [0, 1]$ be a weighting parameter between energy and distance. Define the total distance $D(p)$ and total energy $E(p)$ for the entire route as presented in Equation (5). The objective is written as follows:

$$\text{Cost}(p) = \alpha E(p) + (1 - \alpha) D(p) \quad (5)$$

- The aim is to minimize $\text{Cost}(p)$.

3.2. Assumptions

- Path Existence: For any leg $(s \rightarrow e)$, the function $A_STAR(s, e)$ may return an empty path (i.e., no feasible route through the map), in which case that segment is deemed invalid (or triggers an early mission termination).
- Base Station Reachability: It is typically assumed that W_0 is reachable from any waypoint s . If not, the mission cannot be successfully concluded.
- Fumigation Rate: Each waypoint fumigation requires a fixed volume of chemical (e.g., 660 mL) and a fixed energy consumption during the spraying time.

- Single Visit: Each non-base waypoint appears exactly once in the permutation p .

3.3. Objective

The objective is to find a permutation p^* (of the waypoints $\{1, 2, \dots, N\}$) that minimizes the cost function as presented in Equation (6).

$$\text{Minimize } \{\alpha E(p) + (1 - \alpha)D(p)\} \quad (6)$$

The above is subject to the following:

- Permutation: $p = (p_1, p_2, \dots, p_N)$ with $p_i \neq p_j$ for $i \neq j$.
- Fumigation Limit: Partition p into segments of at most F waypoints, returning to W_0 after each.
- Tank Capacity: The required chemical volume in each segment.
- Map Constraints: Each leg ($s \rightarrow e$) must have a valid path from $A_STAR(s, e)$. Otherwise, that route is infeasible.
- Output: Best Permutation $p^* \in \{1, \dots, n\}$ indicating the visiting order of waypoints.

3.4. PETAL Algorithm

The proposed PETAL algorithm is a hybrid optimization approach designed to plan efficient routes for a spraying robot navigating through multiple waypoints. It integrates a genetic algorithm with A* pathfinding and a 2-OPT (two-edge exchange) local search to minimize the weighted cost of total distance and energy consumption [34]. In this framework, as illustrated in Figure 1, the cost function (COST_OF_PERM) simulates the entire route for a given permutation of waypoints, computing path distances via A*, tracking energy usage through a payload-based travel rate, and accounting for fumigation time and partial refills after a limited number of visits. The 2-OPT local search then refines any route by reversing small segments if it reduces the overall cost. Finally, the genetic algorithm evolves a population of such permutations over several generations, using crossover, mutation, and selection to explore diverse solutions, while 2-OPT provides local improvements. PETAL discovers near-optimal routes that balance distance traveled, energy consumed, and mission constraints, such as payload capacity and refilling requirements, through these subroutines working in tandem.

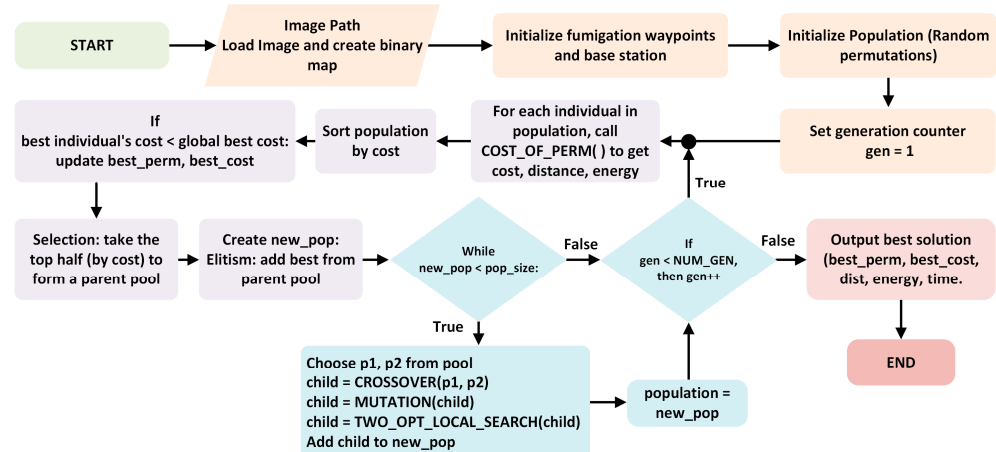


Figure 1. Flowchart of the proposed PETAL algorithm.

3.4.1. Cost of Permutation Calculation

The Subroutine 1 function calculates the total cost of a given waypoint permutation. First, the permutation is split into segments of size FUMIG_LIMIT, each framed by the

base station W0 to model partial refills. For each chunk, it calls A* to find a path between consecutive waypoints and accumulates distance, time, and energy. The travel energy depends on the current payload level, which decreases each time the robot fumigates a waypoint. When returning to the base station, the function increments a refill count. After traversing all mission segments, it sums distance (D) and energy (E) into a single cost value using a weighted formula, as presented in Equation (5).

This function returns both the individual components (distance, energy, time, refills) and the combined cost. By simulating the entire route, COST_OF_PERM provides a crucial fitness metric for the genetic algorithm and local search phases, enabling them to compare alternative permutations accurately and select or refine the most efficient paths.

Subroutine 1: Pseudocode to compute the cost of each permutation

Require: permutations (perm), Waypoints, binary_map

Constants: α , TANK_CAPACITY, SPRAY_RATE, FUMIG_LIMIT (Maximum number of fumigation points), UNIT_TO_METER (Number of units in map to actual meters determined experimentally), SPEED (of robot), FUMI_ENERGY_RATE

Ensure: $\text{Cost}(\text{perm}) = \alpha E(\text{perm}) + (1 - \alpha) D(\text{perm})$ \ \ Equation (5)

Pseudocode:

```

1.   route_of_wp ← [0]
2.   I ← 0
3.   while i < |perm|:
4.     chunk ← perm[i : i + FUMIG_LIMIT]
5.     route_of_wp.extend(chunk); route_of_wp.append(0)
6.     i ← i + FUMIG_LIMIT
7.     total_dist ← 0; total_energy ← 0; total_time ← 0; refills ← 0
8.     missions ← PARSE_MISSIONS(route_of_wp)
9.     for each mission in missions:
10.    M ← count(non-base waypoints in mission)
11.    Payload ← min(M × (SPRAY_RATE × 2), TANK_CAPACITY)
12.    for j ← 0 to |mission| − 2:
13.      s ← mission[j], e ← mission[j + 1]
14.      path ← A_STAR (Waypoints[s], Waypoints[e], binary_map)
15.      dist_units ← |path|; d ← dist_units × UNIT_TO_METER
16.      travel_rate ← 0.0000020888 × payload_ml + 0.0000008853
17.      travel_energy ← travel_rate × d
18.      total_dist += d; total_energy += travel_energy; total_time += d/SPEED
19.      if e ≠ 0:
20.        total_time += 120; total_energy += (120 × FUMI_ENERGY_RATE);
21.        payload -= (SPRAY_RATE × 2)
22.      if payload < 0:
23.        payload ← 0
24.      else refills += 1
25.    Cost ←  $\alpha \times \text{total\_energy} + (1 - \alpha) \times \text{total\_dist}$ 
26.    return (Cost, total_dist, total_energy, total_time, refills)

```

3.4.2. 2-OPT Local Search

The Subroutine 2 refines the waypoint permutation by reversing small sections of the route to reduce crossing paths and lower cost. It starts by computing an initial cost from `COST_OF_PERM`. Then, it iteratively looks for pairs of indices (i, j) such that reversing the segment between them decreases overall cost. If a potential reversal yields a better cost, it updates the best solution found so far and restarts the search within a capped number of iterations (e.g., 20). When no more improvements are detected or the iteration limit is reached, the function returns the final, improved permutation. This method is a simple yet effective local search strategy, removing route overlaps and shortening travel distance. Because it depends on the cost function's feedback, it can also account for energy usage and refills, not just pure distance, making it well suited to the PETAL framework.

Subroutine 2: Pseudocode to try and compute smaller segments from Subroutine 1

Require: perm, `COST_OF_PERM`

Ensure: Improved permutation with reduced cost

Pseudocode:

```

1.   (best_cost, ...) ← COST_OF_PERM(perm); best ← perm
2.   improved ← true; iter ← 0; MAX_ITER ← 20 (given by user)
3.   while improved ∧ iter < MAX_ITER do:
4.     improved ← false; iter++
5.     for i ← 0 to |best| − 2:
6.       for j ← i + 2 to |best| − 1:
7.         new_seq ← best
8.         Reverse (new_seq[i+1..j])
9.         (new_cost, ...) ← COST_OF_PERM(new_seq)
10.        if new_cost < best_cost:
11.          best ← new_seq;
12.          best_cost ← new_cost;
13.          improved ← true;
14.        break
15.    if improved:
16.      break
17.    return best

```

3.4.3. Complete Solution

Algorithm 1 implements a genetic algorithm that evolves a population of permutations. It initializes a random set of solutions (optionally forcing a particular waypoint first). Each generation calculates the cost of every individual using `COST_OF_PERM`. After sorting by ascending cost, the top performers are selected as parents. A new population is generated by applying crossover and mutation operators, then refining each offspring with `TWO_OPT_LOCAL_SEARCH`. Elitism is used to retain the single best individual. Over multiple generations, the population converges toward permutations with lower costs. The final result is the best solution discovered, along with its associated cost, distance, energy, time, and refill count. This integration of genetic algorithm techniques with local search ensures a balance between global exploration (through crossover and mutation) and local exploitation (through 2-OPT), yielding near-optimal routes for the PETAL problem.

Algorithm 1: Pseudocode to run the genetic algorithm and compute best permutation

Require: pop_size, num_gen, mutation_rate, COST_OF_PERM, TWO_OPT

Ensure: Best solution over generations

Pseudocode:

```

1.   population  $\leftarrow$  INIT_POPULATION (pop_size)
2.   best_cost  $\leftarrow \infty$ ; best_perm  $\leftarrow \emptyset$ 
3.   for gen  $\leftarrow 1$  to num_gen:
4.     scored  $\leftarrow []$ 
5.     for indiv in population:
6.       (c, ...)  $\leftarrow$  COST_OF_PERM(indiv);   scored.push(indiv,c)
7.     sort scored by c ascending;
8.     if scored[0].c < best_cost:
9.       best_cost  $\leftarrow$  scored[0].c;
10.    best_perm  $\leftarrow$  scored[0].indiv
11.    selected  $\leftarrow$  top half of scored
12.    new_pop  $\leftarrow$  [selected[0].indiv]    // Elitism
13.    while |new_pop| < pop_size:
14.      p1, p2  $\leftarrow$  choose two from selected
15.      child  $\leftarrow$  MUTATE(CROSSOVER(p1, p2), mutation_rate)
16.      child  $\leftarrow$  TWO_OPT_LOCAL_SEARCH(child)
17.      new_pop.push(child)
18.      population  $\leftarrow$  new_pop
19.    return (best_perm,best_cost)

```

4. Simulation and Validation of the PETAL Algorithm

This section covers two key sections: (i) random map generation and waypoint fixation and (ii) comparing the proposed algorithm with the existing path planning techniques. The first section focuses on generating randomized maps and waypoints essential for navigation. The second section involves applying the proposed PETAL and the existing path-planning algorithms to identify the most optimized path-planning technique.

4.1. Random Map Generation

In this study, a random map is generated with dense obstacles in a static environment, as shown in Figure 2a. Similarly, 50 random waypoints [W0, W1, W2, ..., W48, W49] are fixated, as illustrated in Figure 2b. The waypoint W0 serves as the base station or initial point, where the robot starts and returns for chemical refilling. The remaining 49 waypoints [W1, W2, ..., W48, W49] represent fumigation hotspots, where the robot must navigate and perform fumigation tasks. Given the full payload capacity of the robot, it can fumigate up to seven waypoints before requiring a return to the base station W0 for refueling. Consequently, to cover all 49 waypoints, the robot must complete seven separate missions from the base station W0. The challenge lies in determining the most efficient path for the robot to visit all fumigation hotspots while minimizing travel distance and energy consumption. This randomly generated map serves as the testing environment for evaluating the performance of the proposed and existing path-planning algorithms. The objective is to analyze and compare different path-planning techniques to identify the most optimized path-planning strategy, ensuring efficient navigation while considering obstacle avoidance, energy constraints, distance traveled, and operational effectiveness.

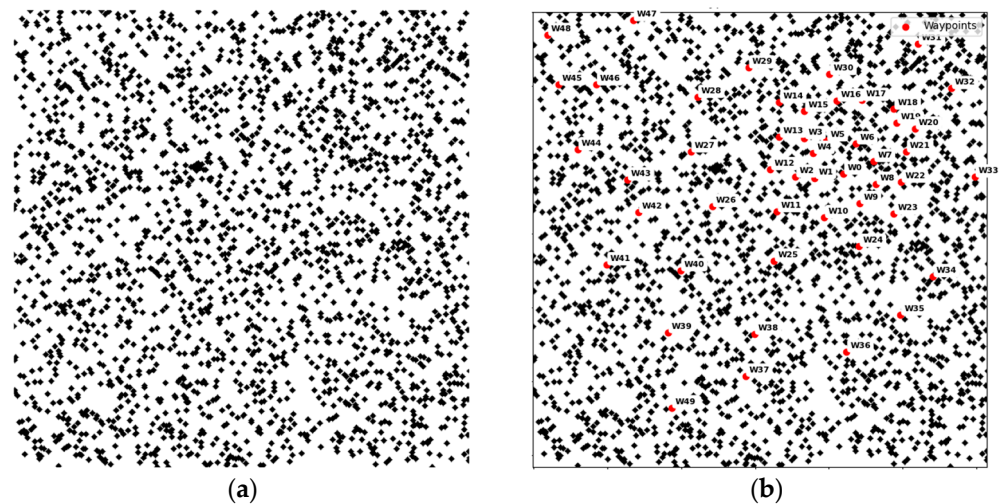


Figure 2. (a) Randomly generated map; (b) Randomly generated map with waypoints.

4.2. Comparing with Existing Path-Planning Techniques

In this study, a total of five path-planning algorithms are considered, including the proposed path-planning algorithm. The selected algorithms are the A* algorithm, Dijkstra's algorithm incorporated with A*, the greedy distance algorithm, the random search algorithm, and the proposed PETAL algorithm. Each algorithm is evaluated based on its efficiency in navigating through the randomly generated map while considering obstacles and optimizing travel distance. The performance of these algorithms is compared to determine the most effective path-planning strategy for fumigation tasks, ensuring optimal navigation, reduced energy consumption, and improved operational efficiency in the given environment.

Robotic path planning involves global and local planners working together to ensure efficient navigation. Global planners generate optimal paths, considering static obstacles and terrain constraints, while local planners make real-time adjustments using sensor feedback. In this study, the fumigation robot follows a global planner approach for overall map generation, ensuring efficient fumigation waypoint coverage. To adapt to dynamic obstacles, a local planner is integrated, utilizing 3D LiDAR and depth perception cameras for real-time obstacle detection and path adjustments. The local planner recalculates the path when an obstacle is detected and temporarily stops the robot to let the pedestrians or dynamic obstacles pass, ensuring smooth and safe navigation with minimal deviation from the planned route. Additionally, the robot employs real-time obstacle avoidance to navigate urban environments effectively, adapting to temporary obstructions without compromising efficiency. However, the dynamic obstacles are completely unpredictable during the planning stage of the mission.

During the mission execution, whenever a path needs to be found between two waypoints, the A* algorithm is employed to determine an efficient route through the environment. The A* path planning algorithm is a widely used graph-based search algorithm that efficiently finds the shortest path between a start and goal point. It works by evaluating nodes using a cost function, $f(n) = g(n) + h(n)$, where $g(n)$ represents the actual cost from the start node to the current node, and $h(n)$ is the heuristic estimate of the cost from the current node to the goal. By prioritizing nodes with the lowest $f(n)$ value, A* balances optimality and efficiency, making it ideal for robot navigation, obstacle avoidance, and path optimization in complex environments. Figure 3a illustrates the A* path planning strategy, showcasing the route from the base station (W0) to all other 49 waypoints in the randomly generated map.

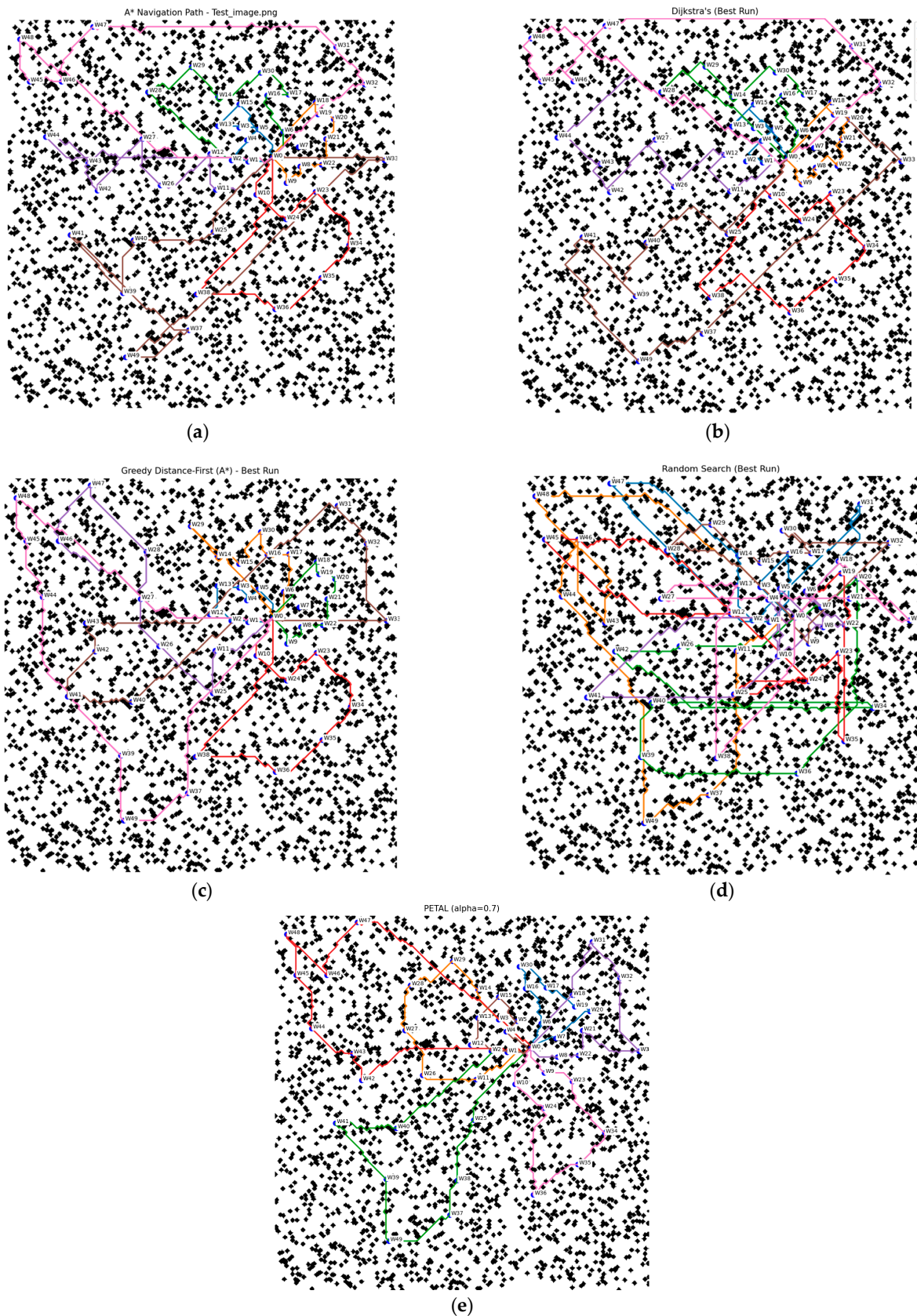


Figure 3. (a) Implication of A* path planning to the map; (b) Implication of Dijkstra's with A* path planning to the map; (c) Implication of greedy distance path planning to the map; (d) Implication of random search path planning to the map; (e) Implication of proposed PETAL algorithm path planning to the randomly generated map.

Dijkstra's path planning algorithm is also a graph-based search algorithm that finds the shortest path from a start node to all other nodes by systematically exploring paths with the lowest cost first. It initializes all node distances as infinity, except the start node W0, which is set to zero. The algorithm selects the node with the smallest distance, updates its neighboring nodes with the lowest possible cost, and repeats this process until all nodes are visited. The Dijkstra's algorithm incorporated with the A* path planning method combines the strengths of both algorithms for efficient navigation. Dijkstra's algorithm is used for waypoint selection, ensuring the shortest and most optimal waypoints are identified by evaluating all possible paths from the start node to the goal node based on minimal cost. Once waypoints are determined, the A* algorithm is applied for path planning, calculating the shortest and most efficient route between selected waypoints. Here also, A* uses the same heuristic cost function, $f(n) = g(n) + h(n)$. This hybrid approach ensures the robot follows an optimal route while efficiently navigating obstacles. Integrating Dijkstra's for waypoint selection and A* for path planning significantly improves efficiency, making it well suited for applications requiring optimized navigation. Similarly, Figure 3b illustrates the path planning strategy using Dijkstra's algorithm incorporated with A*, starting from the base station W0 and navigating to all 49 waypoints in seven missions.

Next, the greedy distance path planning algorithm follows a greedy approach, as shown in Figure 3c, selecting the closest node to the current position at each step without considering the overall path cost. It prioritizes immediate distance minimization, making decisions based only on local optimization rather than the entire path. The algorithm chooses the nearest unvisited waypoint, moves to it, and repeats the process until reaching the goal. While this approach is computationally efficient and simple to implement, it can lead to suboptimal paths, such as becoming trapped in local minima. It is mainly used for quick navigation solutions where path optimality is not critical.

Similarly, the random search path planning algorithm explores paths by selecting random waypoints without following a fixed heuristic or cost function. The robot or agent randomly selects a direction or waypoint, moves towards it, and continues this process until it reaches the goal. Unlike structured algorithms like A* or Dijkstra's, random search lacks optimization and may result in inefficient, lengthy paths. However, it is helpful in unknown or dynamic environments where predefined heuristics may not apply. While simple to implement, it often requires multiple iterations to find an optimal path, making it less reliable for time-sensitive or energy-constrained applications. In this random search method, a total of 50 factorial paths are randomly generated for the 50 waypoints. However, 3000 iterations were performed to reduce computational demands, as shown in Figure 4. From these iterations, the paths with the least energy consumption were selected and considered for analysis in this study, as presented in Figure 3d. This approach balances exploration of possible paths with computational efficiency, ensuring practical evaluation while minimizing resource usage.

Finally, the proposed PETAL algorithm is applied to the randomly generated map for path planning. In this approach, the alpha value in the cost function is set to 0.7, balancing both energy consumption and distance traveled by the robot. A total of 300 iterations were executed, as illustrated in Figure 5. After several iterations, energy consumption and distance traveled values stabilized, indicating convergence toward optimal paths. The least energy-consuming and shortest-distance paths were selected for further analysis, as shown in Figure 3e. This process leads to a reduced or optimal cost function, ensuring efficient navigation while minimizing both energy use and travel distance. By adjusting the alpha value, the algorithm prioritizes these two critical factors, making it suitable for applications requiring a balance between energy efficiency and travel distance.

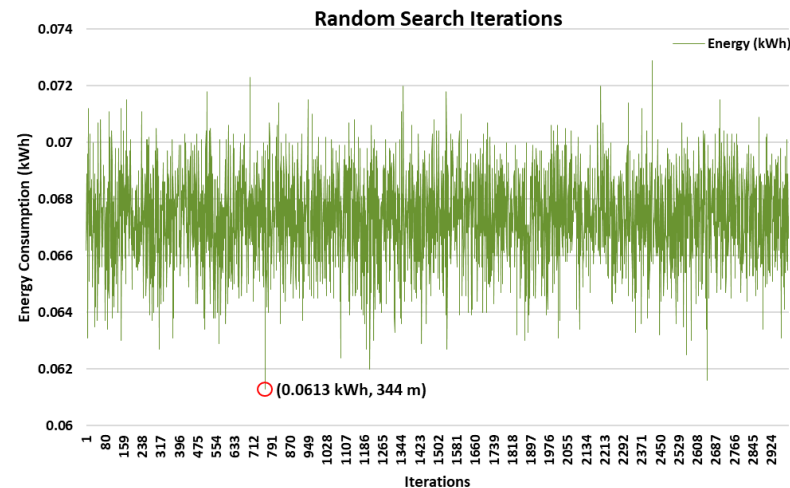


Figure 4. Random search algorithm iterations for least energy consumed path.

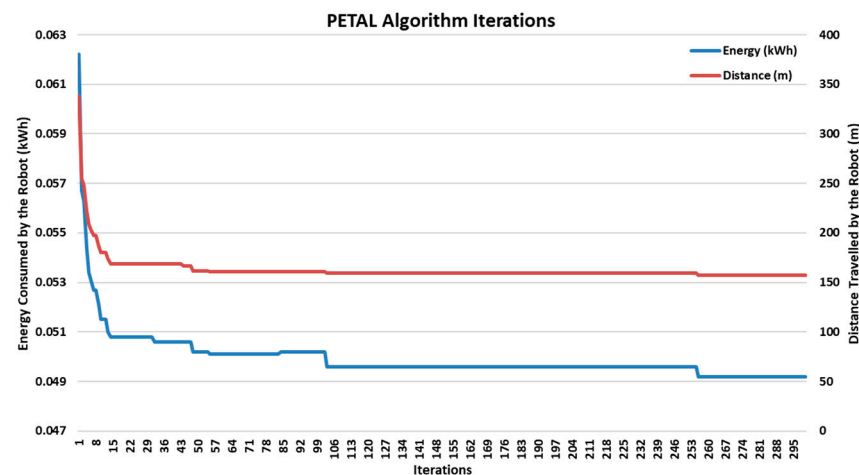


Figure 5. Proposed PETAL algorithm iterations for least energy consumed and distance traveled path.

Overall, Figure 3a–e illustrates different path planning strategies of the fumigation robot, where each colored line in those figures represents a separate mission. The robot starts from the base station (W0) and navigates through 49 waypoints that require fumigation. Given the robot's fumigation chemical payload capacity, it can fumigate up to seven waypoints for each mission before returning to the base station for refilling. As a result, the robot takes seven missions or seven paths to fumigate all 49 waypoints, each represented in a different color for clarity. This visualization highlights the optimized path planning, ensuring minimal energy consumption and reduced travel distance while efficiently covering all designated fumigation waypoints.

Table 1 compares five path-planning algorithms based on their energy consumption (in kWh) and distance traveled (in meters). Among all the algorithms, the proposed PETAL algorithm demonstrates the most efficient performance, consuming only 0.0492 kWh of energy and covering a distance of 156.93 m, making it the most optimized solution in terms of both energy and distance. The A* algorithm consumes 0.052011 kWh and travels 199.89 m, showing higher energy usage and longer distances compared to the proposed method. Similarly, Dijkstra's with A* consumes 0.052118 kWh and covers 194.33 m, indicating slightly better distance travel efficiency than A*, but is still more energy-intensive and requires more travel compared to the proposed approach. The greedy distance algorithm has an energy consumption of 0.052177 kWh and a distance of 193.93 m, which is close to the Dijkstra's with A* algorithm but still not as efficient as the proposed algorithm. The

random search algorithm shows the least efficiency, with a consumption of 0.061300 kWh and a distance traveled of 344 m, significantly higher than all other algorithms in both energy use and travel distance.

Table 1. Energy consumption and distance traveled comparison in random map with five path planning algorithms.

Algorithm	Computing Time (s)	Energy Consumption (kWh)	Distance Traveled (m)	% Change in Energy w.r.t. PETAL	% Change in Distance Traveled w.r.t. PETAL
A* algorithm	5	0.052011	199.89	5.71	27.38
Dijkstra's with A* algorithm	6	0.052118	194.33	5.93	23.83
Greedy Distance First	5	0.052177	193.93	6.05	23.58
Random Search	6 h (for 3000 iterations)	0.061300	344	24.59	119.21
PETAL algorithm	51 (for finding the least cost path)	0.0492	156.93	-	-

Figure 6 presents the percentage change in energy consumption relative to the proposed PETAL algorithm across different path-planning strategies. The A* algorithm shows a 5.71% increase in energy consumption compared to the proposed method, making it less efficient. The Dijkstra's with A* algorithm follows with a 5.93% slightly higher energy consumption than A*. The greedy distance algorithm consumes 6.05% more energy, similar to Dijkstra's with A*, but is still more energy-intensive than the proposed approach. The random search algorithm exhibits the highest energy increase, with a 24.59% higher energy consumption than the proposed PETAL method, indicating significant inefficiency in both energy use and path planning.

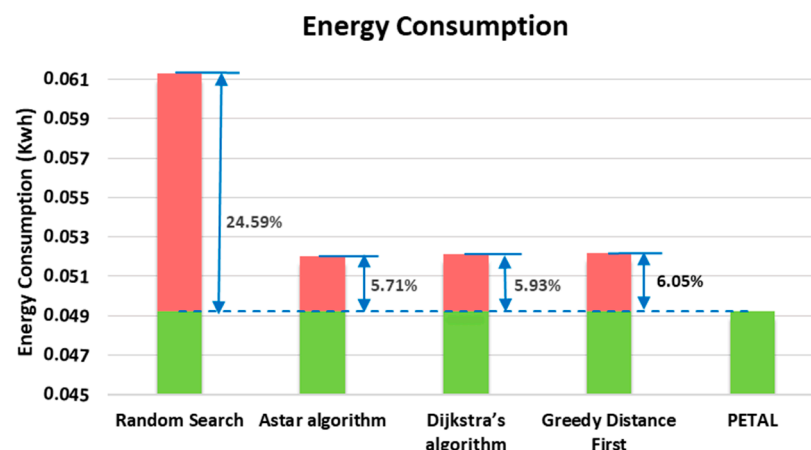


Figure 6. Energy consumption comparison in random map with five path planning algorithms.

Figure 7 shows the percentage change in distance traveled relative to the proposed PETAL algorithm for various path-planning algorithms. The A* algorithm requires 27.38% more distance than the proposed method, indicating a less efficient path. The Dijkstra's with A* algorithm follows with a 23.83% increase in distance. In comparison, the greedy distance algorithm shows a 23.58% higher distance, making it slightly more distance traveled than A* but is still less optimal than the proposed algorithm. The random search algorithm is the least efficient, with a 119.21% increase in distance, suggesting a significant inefficiency in both travel distance and path planning compared to the proposed approach.

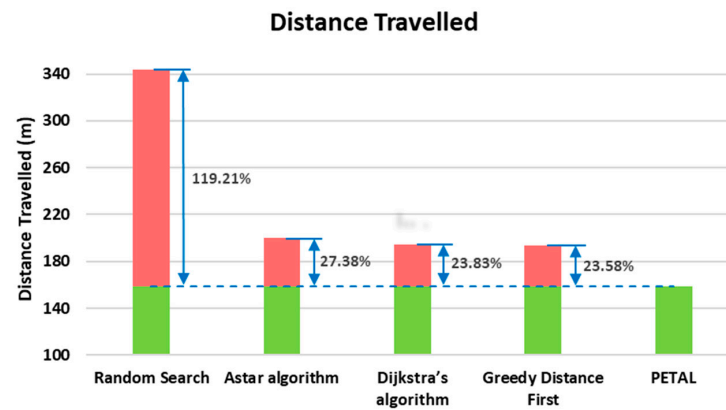


Figure 7. Distance traveled comparison in random map with five path planning algorithms.

Overall, the proposed PETAL algorithm performs better in terms of energy efficiency and distance traveled than other path-planning algorithms. It consumes the least energy (0.0492 kWh) and covers the shortest distance (156.93 m), making it the most optimized solution for path-planning tasks. While A*, Dijkstra's with A*, and greedy distance algorithms show moderate improvements, they still consume more energy and travel greater distances. The random search algorithm is the least efficient, exhibiting significantly higher energy consumption and longer travel distances. Therefore, the proposed PETAL algorithm is the most effective for achieving optimal path planning, significantly reducing energy consumption and distance traveled for a given payload capacity.

5. Case Study on Developed Fumigation Robot

This section is divided into four key subsections. First, it elaborates on the developed fumigation robot and its specifications. Second, it explains the case study map used to analyze the fumigation robot's energy consumption in that environment. The third subsection discusses the implementation of five path-planning algorithms, including the proposed algorithm, on the given case study map to determine the most optimized path-planning technique. These sections comprehensively understand the robot's efficiency and performance in fumigation tasks. Finally, it ends with limitations of the current algorithm and a brief on future scopes.

5.1. Developed Autonomous Fumigation Robot

The fumigation robot has been specifically designed to address the challenges of urban pest control, with a focus on efficiency, continuous fumigation, and adaptability. The key specifications of the developed fumigation robot are presented in Table 2. A differential drive system powered by brushless DC (BLDC) motors is chosen for smooth navigation of the robot, which is suitable for flat surfaces, i.e., common in urban infrastructure. The motors BLHM450KC-30 (Oriental Motor Company, Singapore) are used for precise locomotion, enhancing maneuverability in complex terrains. The robot is incorporated with advanced sensing technologies, including a SICK TiM-571 (SICK, Singapore) two-dimensional (2D) LiDAR and a Vectornav VN-100 IMU (VectorNav Technologies, Singapore) sensor. These tools enable effective environmental mapping, obstacle detection, and real-time path planning. This ensures that the robot operates safely and efficiently in dynamic urban landscapes. The robot's processing power is driven by a Nuvo-10108GC-RTX3080 (Voltrium Systems, Singapore) industrial-grade processor, which supports advanced tasks such as multi-sensor integration, navigation, object detection, and real-time decision-making. A 48V 25Ah lithium iron phosphate battery provides up to four hours of continuous operation, optimizing the robot's productivity. The fumigation system employs a continuous spraying

method to provide uniform chemical coverage, ideal for large areas. The fumigation system employs a continuous spraying method to provide uniform chemical coverage, ideal for large areas. It is equipped with a 5 L fumigation chemical tank with a spray gun that delivers fine droplets of 50 microns at a flow rate of 330 mL/min. The fumigation spray gun is mounted with one end connected to a linear actuator and the other secured to a 3D-printed mount, enabling precise tilt motion. The opposite side of the linear actuator attaches to a metal shaft, which is paired with a stepper motor to provide rotational panning. This configuration allows the fumigation spray gun to achieve a full 360° rotation, effectively covering areas up to 4.5 m in height and within a range of 2.5 to 6 m. The integrated tilt and pan mechanism ensures complete and efficient coverage of the target area for optimal fumigation performance. Figure 8 illustrates the final developed autonomous fumigation robot used in this study for deploying the proposed algorithm.

Table 2. Key specifications of the developed fumigation robot.

Parameters	Developed Fumigation Robot Key Specifications
Terrain	Urban landscape
Drive Type	Differential drive with two BLDC motors BLHM450KC-30
Size (mm)	692 (L) × 517 (W) × 1080 (H)
Weight (kg)	30
Sensors	SICK TiM-571 2D LiDAR, Vectornav VN-100 IMU
CPU Processor	Industrial Processor: Nuvo-10108GC-RTX3080
Battery	48 V, 25 Ah
Fumigation Type	Continuous spraying
Tank Capacity (L)	5
Spray Flow Rate	330 mL/min



Figure 8. Developed autonomous fumigation robot.

The proposed payload and energy-aware path planning algorithm enable a detailed analysis of energy consumption and distance traveled for the developed autonomous fumigation robot. Energy consumption is measured based on the distance and power consumed during the waypoint coverage process. In contrast, time consumption accounts for both the travel time and delays caused by refilling the fumigation chemical tank, which impacts overall operation efficiency. The robot is navigated autonomously from point A to point B to evaluate these parameters under precision fumigation conditions. During this trial, the energy consumed and the distance covered are recorded. The distance from point A to point B is approximately 10 m. On average, this corresponds to 270 units, where 1 unit equals 1/27 m, meaning 1 m is equivalent to 27 units. Subsequently, the robot is navigated over the same distance without fumigation to simulate the scenario of returning to the base station to refill the fumigation chemical tank and recharge the battery. The

energy consumption from this scenario is recorded. The collected data are then normalized for both scenarios. These normalized values are applied in simulations to calculate the overall distance traveled, energy, and time consumption for the robot's operations under the existing and proposed PETAL algorithms.

5.2. Explanation of the Case Study Map

This study implements the proposed algorithm on an autonomous fumigation robot within a case study map to assess its effectiveness. Figure 9 presents both the real-time case study map and the 3D LiDAR-generated map. To enhance understanding, 21 random waypoints [W0, W1, W2, . . . , W19, W20] are designated and shown in Figure 10a,b, along with their coordinates. Similarly, the waypoint W0 serves as the base station, where the robot starts its mission and returns for chemical refilling. The remaining 20 waypoints act as fumigation hotspots, requiring the robot to navigate and perform fumigation tasks. Given the robot's full payload capacity of 5 L, it can fumigate up to seven waypoints per mission before returning to W0 for refueling. As a result, completing the fumigation of all 20 waypoints requires three separate missions from the base station. To systematically manage navigation, the map is divided into three zones as shown in Figure 10a: Zone 1 (W0, W1, W2), Zone 2 (W3, W4, W5, W10, W17, W18, W19, W20), and Zone 3 (W6, W7, W8, W9, W11, W12, W13, W14, W15, W16). Here, Zone 1 is designated as the nearest region, which holds all the closest waypoints; Zone 2 represents waypoints with moderate distance, and Zone 3 contains the farthest waypoints from the base station, as presented in Figure 10a.

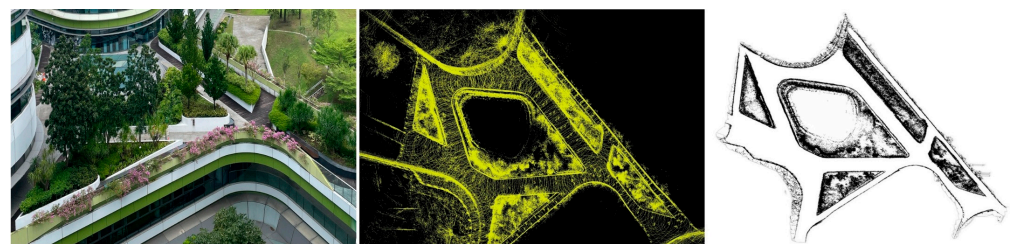


Figure 9. Case study map used for analyzing the proposed path planning algorithm.

All five path planning algorithms, including the proposed algorithm, are implemented within this case study map while adhering to the given constraints. The primary challenge is determining the most efficient path, allowing the robot to visit all fumigation hotspots while minimizing travel distance and energy consumption. This case study serves as a testbed for evaluating the performance of both existing and proposed path-planning algorithms with the considered autonomous fumigation robot.

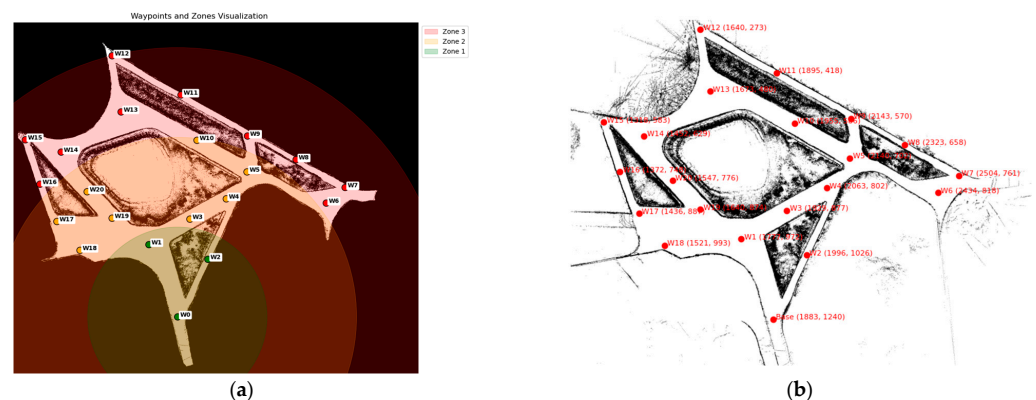


Figure 10. (a) Case study map displaying waypoints and zonal visualization. (b) Waypoint with their coordinates.

5.3. Deploying Path-Planning Algorithms on the Map

This study evaluates five path-planning algorithms, including the proposed PETAL algorithm, for their effectiveness. The selected algorithms are A*, Dijkstra's algorithm combined with A*, the greedy distance algorithm, the random search algorithm, and the proposed PETAL algorithm. Each algorithm is assessed based on its ability to navigate through the case study map, efficiently avoid obstacles, and minimize travel distance. Their performances are compared to identify the most optimized path-planning strategy for fumigation tasks, ensuring efficient navigation, reduced energy consumption, and enhanced operational effectiveness in the given environment. Figure 11a–e represent the implication of the five path planning algorithms to the given case study map to navigate the autonomous fumigation robot to all of the waypoints.

The proposed PETAL algorithm is applied to the case study map for path planning, where the energy–distance cost value stabilized after 20 iterations, as illustrated in Figure 12. Similarly, in the case study map, the alpha value in the cost function is set to 0.7, ensuring a balance between energy consumption and distance traveled by the robot. Over time, the values for energy consumption and distance traveled reached stability as a randomly generated map, indicating that the algorithm was converging toward optimal paths. The paths with the least energy consumption and shortest distance were then selected for further analysis, as shown in Figure 11e. This process resulted in a reduced or optimal cost function, facilitating efficient navigation while minimizing both energy usage and travel distance. By adjusting the alpha value, the algorithm effectively prioritizes both factors, making it well suited for applications that require an optimal balance between energy efficiency and travel distance.

Table 3 compares five path-planning algorithms based on their energy consumption (in kWh) and distance traveled (in meters). The PETAL algorithm demonstrates the best performance in terms of both energy efficiency and travel distance, consuming 0.0306 kWh of energy and covering a distance of 215.48 m. It stands out as the most optimized solution among the algorithms, offering a balanced approach to minimizing energy usage and distance. The A* algorithm, while efficient, consumes 0.0322 kWh and covers 236.19 m, resulting in a 5.23% higher energy consumption and a 9.61% longer travel distance compared to PETAL. In comparison, the Dijkstra's with A* algorithm consumes 0.0354 kWh and covers 283.70 m, showing a significant increase in both energy consumption (15.69%) and distance traveled (31.66%) compared to PETAL, indicating less efficiency in both respects. The greedy distance algorithm shows similar results to Dijkstra's, with identical energy consumption (0.0354 kWh) and distance traveled (283.70 m), meaning it is equally inefficient when compared to the PETAL algorithm. The random search algorithm is the least efficient, consuming 0.0397 kWh and traveling 342.59 m, leading to 29.74% more energy consumption and 58.99% greater distance traveled than PETAL, as illustrated in Figures 13 and 14. Overall, the PETAL algorithm consistently outperforms the other algorithms by offering the lowest energy consumption and a reasonable travel distance, making it the most optimized and efficient solution for the given task.

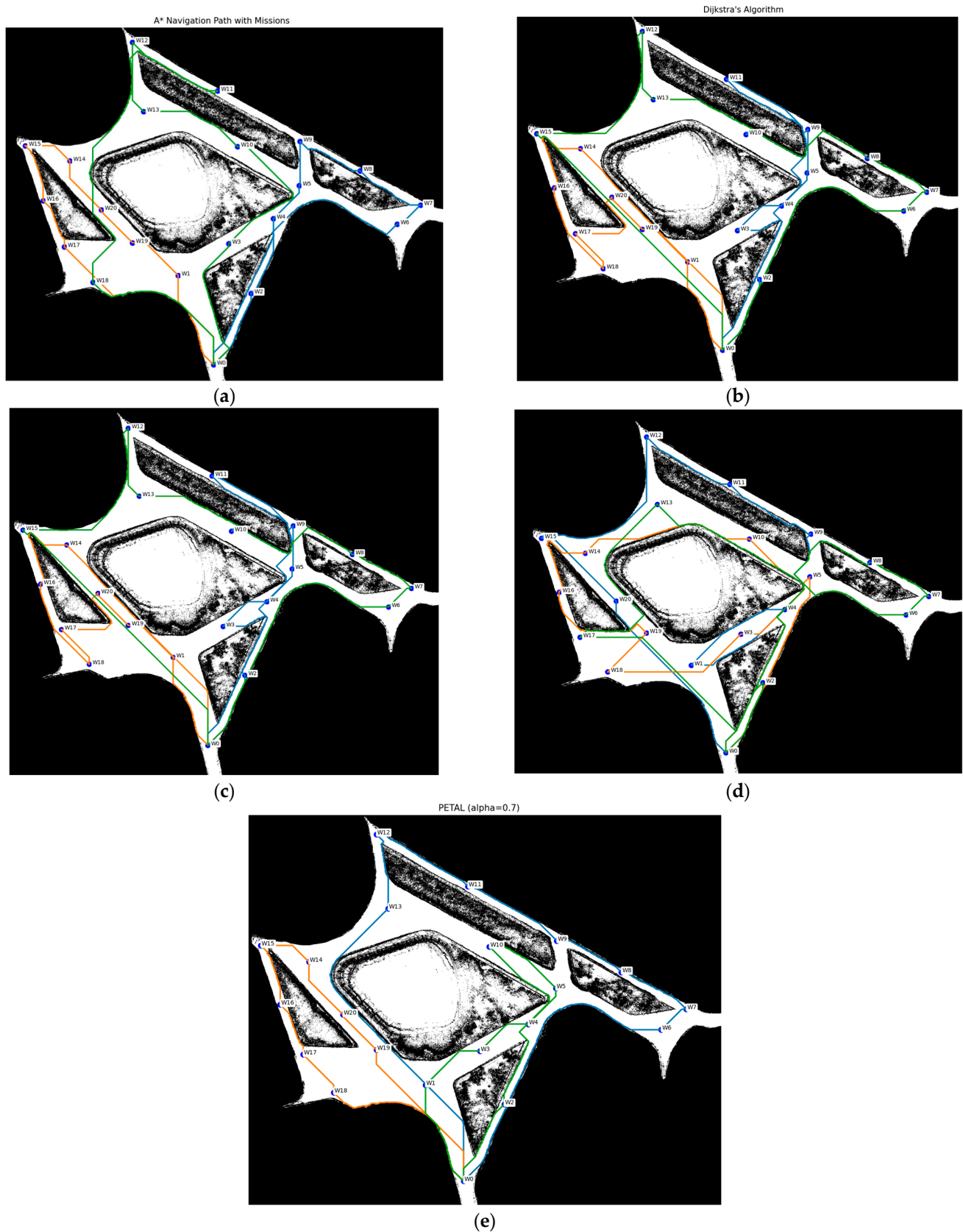


Figure 11. (a) Implication of A* path planning to the case study map; (b) Implication of Dijkstra's with A* path planning to the case study map; (c) Implication of greedy distance path planning to the case study map; (d) Implication of random search path planning to the case study map; (e) Implication of proposed PETAL algorithm path planning to the case study map.

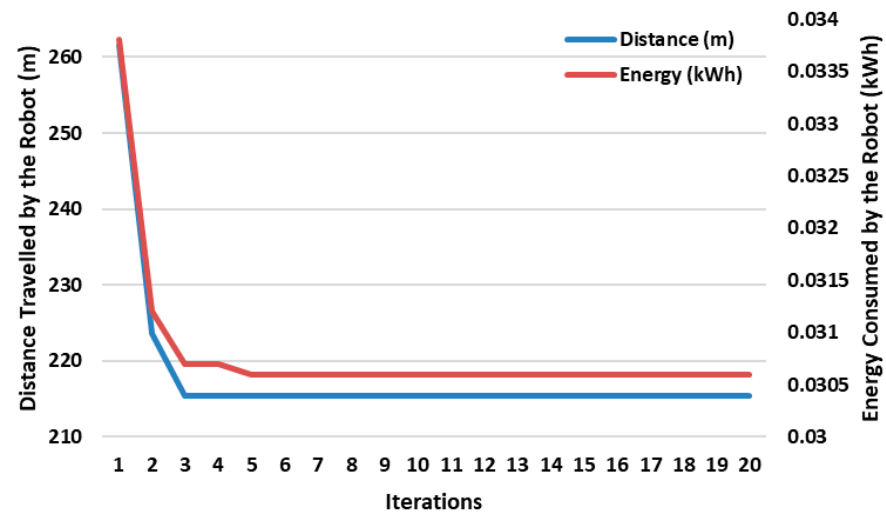


Figure 12. Proposed PETAL algorithm iterations for least energy consumed and distance traveled path in case study map.

Table 3. Energy consumption and distance traveled comparison in real-time map with five path planning algorithms.

Algorithm	Computing Time (s)	Energy Consumption (kWh)	Distance Traveled (m)	% Change in Energy w.r.t. PETAL	% Change in Distance Traveled w.r.t. PETAL
A* algorithm	3	0.0322	236.19	5.23	9.61
Dijkstra's with A* algorithm	4	0.0354	283.70	15.69	31.66
Greedy distance first	3	0.0354	283.70	15.69	31.66
Random search	6 h (for 3000 iterations)	0.0397	342.59	29.74	58.99
PETAL algorithm	32 (for finding the least cost path)	0.0306	215.48	-	-

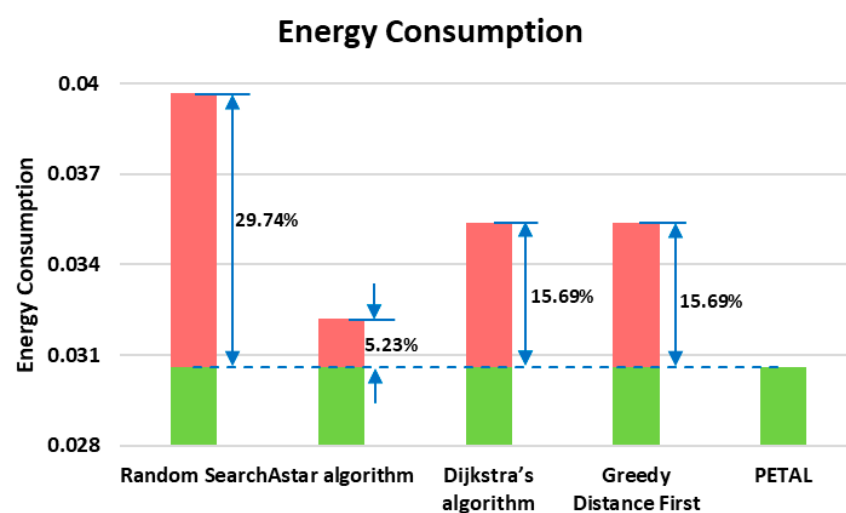


Figure 13. Energy consumption comparison in real-time map with five path planning algorithms.

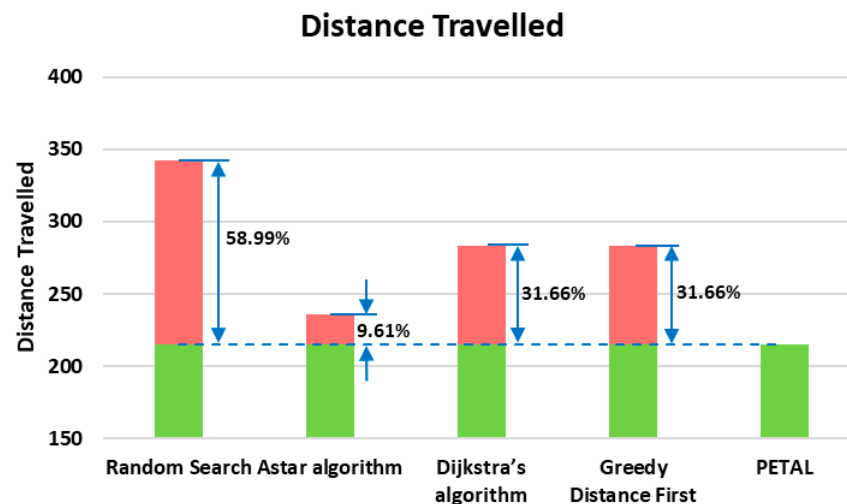


Figure 14. Distance traveled comparison in real-time map with five path planning algorithms.

Although the PETAL algorithm yields better results than the other algorithms, there are certain time and space complexity tradeoffs. With regard to time complexity, Dijkstra's algorithm operates in $O((V + E) \log V)$ time complexity at best case and $O((V^2) \log V)$ at worst case, where E and V represent the number of edges and vertices, respectively, on graph representations. Similarly, for A* search, the time and space complexity is given by $O(b^m)$, where b is the branching and m is the maximum search tree depth. The greedy distance-first algorithm selects the nearest first using Euclidean distance, leading to minimal time complexity (3 s in this case), but at a cost of fewer global optimizations. Random search performs poorly as it has to examine thousands of iterations, consuming hours, reflecting its brute-force nature. PETAL's memetic approach carries higher computational time due to the evolving and refining permutations and achieves the most efficient route. PETAL's computation is majorly due to the repeated evaluation of the population permutations over multiple generations, each requiring path cost calculation using the A* algorithm (T_{Astar}). Each permutation invokes N times the A* ($N \times T_{Astar}$) per segment per mission for a population size of P , number of generations G , and number of waypoints N . Furthermore, the 2-OPT local search adds a constant overhead factor c per offspring, making the approximate time complexity $O(c \times P \times G \times N \times T_{Astar})$, where T_{Astar} is approximately equal to $(V + E) \log V$ (or b^m in search tree view). In terms of space complexity, the A* and Dijkstra's algorithms store the open and closed sets proportional to the node space, while the random search maintains a large pool of solutions. PETAL stores a population of permutations $O(P \times N)$ and A*'s open or closed sets, approximating the space requirement as $O(P \times N + V)$. While this is higher than simple A* or greedy approaches, PETAL's evolutionary search and local optimization synergy yields more effective distance-energy trade-offs.

Therefore, the PETAL algorithm stands out as the most effective for optimal path planning in fumigation service robot applications, offering a significant reduction in both energy consumption and travel distance for a given payload capacity, making it the ideal choice for efficient and resource-conscious navigation in autonomous fumigation robots.

5.4. Limitations and Future Scope

Despite PETAL's potential, certain limitations hamper widespread adoption and necessitate future improvements. The algorithm presumes a static environment, uses a single-objective approach balancing energy and distance, and employs a 2-OPT local search that might be insufficient for large, complex route structures. Additionally, large-scale deployments can pose computational challenges even with caching. Addressing these

shortcomings paves the way for advanced multi-objective formulations, dynamic scenario adaptations, and more refined local searches. Ultimately, the future of PETAL lies in robust real-time planning, extended objective definitions, and distributed or parallel implementations for scalable performance. These expansions will broaden PETAL's applicability. A summary of the current limitations is listed below:

Current Limitations:

- **Static Environment Assumption:** PETAL relies on a pre-processed binary map for pathfinding and cost calculations, assuming that obstacles and navigable areas remain static throughout the mission. If the operational environment experiences dynamic changes, the algorithm may yield suboptimal routes or fail to adapt promptly.
- **Single-Objective Formulation:** Although PETAL addresses a cost function balancing distance and energy, it treats the trade-off through a single weight (α). Missions requiring additional objectives, such as time-critical deliveries, fleet coordination, or risk minimization, remain outside its current scope, reducing its applicability in highly multi-objective scenarios.
- **Limited Local Refinement:** The 2-OPT local search is computationally cheap but focuses primarily on reducing crossing edges in a route. To ensure deeper local improvements, complex route structures or nuanced energy profiles might need more sophisticated neighborhood explorations (e.g., 3-OPT or Lin-Kernighan).
- **Scalability Constraints:** While caching A* paths accelerates repeated distance lookups, the underlying genetic algorithm can still exhibit high computational demands for large problem instances (hundreds of waypoints), potentially limiting real-time or on-the-fly mission planning.

Future Development Scope:

- **Dynamic and Real-Time Extensions:** Incorporating online updates of obstacle locations and real-time waypoint modifications would allow PETAL to adjust routes instantly whenever environmental changes occur, enhancing robustness in rapidly evolving domains such as disaster relief or precision agriculture.
- **Multi-objective Optimization:** Extending PETAL to a fully multi-objective genetic algorithm would simultaneously capture additional mission goals (such as cost, time, coverage, or safety), providing decision-makers with a Pareto front of equally optimal solutions instead of a single weighted outcome.
- **Advanced Local Search Operators:** Incorporating more sophisticated local search heuristics or hybridizing 2-OPT with large neighborhood search could yield improved solution quality in less time. Such improvements would be especially beneficial for problems involving highly irregular terrains or intricate constraints.
- **Adaptive Parameter Tuning:** Enabling on-the-fly adjustment of genetic parameters (e.g., crossover rates, mutation probabilities, or population sizes) based on progress indicators or feedback from mission performance could accelerate convergence and reduce parameter selection overhead.
- **Distributed or Parallel Implementations:** Leveraging parallel computing platforms, whether multi-core CPUs, GPUs, or distributed cloud environments, would improve scalability. Large populations, extensive path caching, and multiple local searches could be processed in parallel, reducing runtime and supporting much larger mission scenarios.
- **Multi-Objective Optimization:** Integrating additional parameters such as operational time, coverage efficiency, and safety risks alongside energy efficiency and distance traveled will enhance real-world applicability. Improvements in path planning, dynamic obstacle avoidance, and speed regulation will contribute to a more efficient, adaptive, and safer fumigation robot for complex urban environments.

6. Conclusions

The proposed PETAL algorithm integrates a genetic algorithm, a 2-OPT local search, and an energy-based cost function to optimize path planning for urban fumigation robots. PETAL's emphasis on payload dynamics and strategic trajectory planning ensures minimal travel distance and energy consumption. Numerical evaluations conducted on both randomly generated maps and a 3D LiDAR-based real-time environment consistently demonstrate PETAL's performance over competing algorithms (A*, Dijkstra with A*, greedy distance first, and random search). Specifically, in the randomly generated map, PETAL reduced energy usage by up to 24.59% and travel distance by up to 119.21% compared to the random search approach, achieving 5.71–6.05% improvements over baseline algorithms. On the real-time map, PETAL outperformed alternatives by 29.74% in energy and 58.99% in the distance traveled compared to the random search approach and 5.23–15.69% energy saving over stronger baseline algorithms. A key component of this research involved developing a functional fumigation robot. This system features a 5 L chemical tank with a continuous spraying rate of 330 mL/min, accommodating extended missions in urban areas. By demonstrating that PETAL's route optimization algorithm effectively conserves energy and shortens travel distance, the results indicate that the proposed approach can reduce operator exposure while increasing operational efficiency, pointing toward a scalable solution for real-world pest management applications.

Author Contributions: Conceptualization, P.K.C., B.P.D., S.K., S.M.B.P.S. and M.R.E.; methodology, P.K.C., B.P.D. and A.M.; software, P.K.C. and A.M.; writing—original draft, P.K.C., B.P.D. and A.M.; resources, supervision, and funding acquisition, M.R.E. All authors have read and agreed to the published version of the manuscript.

Funding: This research is supported by the National Robotics Programme under its National Robotics Programme (NRP) BAU, Ermine III: Deployable Reconfigurable Robots, Award No. M22NBK0054 and also supported by the SUTD Growth Plan (SGP) Grant, Grant Ref. No. PIE-SGP-DZ-2023-01.

Data Availability Statement: The data presented in this study are available from the corresponding author upon request. The data are not publicly available due to privacy concerns, proprietary or confidential information, and intellectual property belonging to the organization that restricts its public dissemination.

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

Abbreviations

The following abbreviations are used in this manuscript:

PETAL	Payload and Energy-aware Tactical Allocation Loop
2-OPT	Two-edge exchange
UGV	Unmanned ground vehicles
AGV	Autonomous guided vehicles
IMU	Inertial measurement unit
EKF	Extended Kalman filter
SLAM	Simultaneous localization and mapping
LiDAR	Light detection and ranging
CNN	Computational neural network
BLDC	Brushless DC

References

1. World Health Organization. *Disease Outbreak News: Annual Compendium 2022*; World Health Organization: Geneva, Switzerland, 2024.
2. Abdullah, M.S.; Islam, M.J.; Hasan, M.M.; Sarkar, D.; Rana, M.S.; Das, S.S.; Hossian, M. Impact of Waste Management on Infectious Disease Control: Evaluating Strategies to Mitigate Dengue Transmission and Mosquito Breeding Sites—A Systematic Review. *J. Angiother.* **2024**, *8*, 1–12.
3. Onen, H.; Luzala, M.M.; Kigozi, S.; Sikumbili, R.M.; Muanga, C.-J.K.; Zola, E.N.; Wendji, S.N.; Buya, A.B.; Balciunaitiene, A.; Viškelis, J. Mosquito-Borne Diseases and Their Control Strategies: An Overview Focused on Green Synthesized Plant-Based Metallic Nanoparticles. *Insects* **2023**, *14*, 221. [\[CrossRef\]](#) [\[PubMed\]](#)
4. Abdullah, N.A.M.H.; Dom, N.C.; Salleh, S.A.; Dapari, R.; Precha, N. Dengue's Climate Conundrum: How Vegetation and Temperature Shape Mosquito Populations and Disease Outbreaks. *BMC Public Health* **2025**, *25*, 4. [\[CrossRef\]](#)
5. Choi, J.; Cha, W.; Park, M.-G. Evaluation of the Effect of Photoplethysmograms on Workers' Exposure to Methyl Bromide Using Second Derivative. *Front. Public Health* **2023**, *11*, 1224143. [\[CrossRef\]](#) [\[PubMed\]](#)
6. Chen, Y.-X.; Pan, C.-Y.; Chen, B.-Y.; Jeng, S.-W.; Chen, C.-H.; Huang, J.-J.; Chen, C.-D.; Liu, W.-L. Use of Unmanned Ground Vehicle Systems in Urbanized Zones: A study of Vector Mosquito Surveillance in Kaohsiung. *PLoS Negl. Trop. Dis.* **2023**, *17*, e0011346. [\[CrossRef\]](#) [\[PubMed\]](#)
7. Jeyabal, S.; Vikram, C.; Chittoor, P.K.; Elara, M.R. Revolutionizing Urban Pest Management with Sensor Fusion and Precision Fumigation Robotics. *Appl. Sci.* **2024**, *14*, 7382. [\[CrossRef\]](#)
8. Konduri, S.; Chittoor, P.K.; Dandumahanti, B.P.; Yang, Z.; Elara, M.R.; Jaichandar, G.H. Boa Fumigator: An Intelligent Robotic Approach for Mosquito Control. *Technologies* **2024**, *12*, 255. [\[CrossRef\]](#)
9. Sabry, A.H.; Nordin, F.H.; Sabry, A.H.; Ab Kadir, M.Z.A. Fault Detection and Diagnosis of Industrial Robot Based on Power Consumption Modeling. *IEEE Trans. Ind. Electron.* **2019**, *67*, 7929–7940. [\[CrossRef\]](#)
10. Ma, Q.; Li, M.; Huang, G.; Ullah, S. Overtaking Path Planning for CAV based on Improved Artificial Potential Field. *IEEE Trans. Veh. Technol.* **2023**, *73*, 1611–1622. [\[CrossRef\]](#)
11. Kumar, P.; Bensekrane, I.; Merzouki, R. Power Consumption Modeling of Wheeled Mobile Robots with Multiple Driving Modes. *IEEE Trans. Ind. Electron.* **2022**, *70*, 10282–10291. [\[CrossRef\]](#)
12. Pathmakumar, T.; Muthugala, M.V.J.; Samarakoon, S.B.P.; Gómez, B.F.; Elara, M.R. A Novel Path Planning Strategy for a Cleaning Audit Robot Using Geometrical Features and Swarm Algorithms. *Sensors* **2022**, *22*, 5317. [\[CrossRef\]](#) [\[PubMed\]](#)
13. Le, A.V.; Veerajagadheswar, P.; Kyaw, P.T.; Muthugala, M.V.J.; Elara, M.R.; Kuma, M.; Nhan, N.H.K. Towards Optimal Hydro-Blasting in Reconfigurable Climbing System for Corroded Ship Hull Cleaning and Maintenance. *Expert Syst. Appl.* **2021**, *170*, 114519. [\[CrossRef\]](#)
14. Leng, J.; Peng, J.; Liu, J.; Zhang, Y.; Ji, J.; Zhang, Y. Profiling Power Consumption in Low-Speed Autonomous Guided Vehicles. *IEEE Robot. Autom. Lett.* **2024**, *9*, 6027–6034. [\[CrossRef\]](#)
15. Zhang, H.; Zhang, Y.; Liu, C.; Zhang, Z. Energy Efficient Path Planning for Autonomous Ground Vehicles with Ackermann Steering. *Robot. Auton. Syst.* **2023**, *162*, 104366. [\[CrossRef\]](#)
16. Yang, D.; Liu, T.; Song, D.; Zhang, X.; Zeng, X. A Real Time Multi-Objective Optimization Guided-MPC Strategy for Power-Split Hybrid Electric Bus Based on Velocity Prediction. *Energy* **2023**, *276*, 127583. [\[CrossRef\]](#)
17. Huang, G.; Yuan, X.; Shi, K.; Liu, Z.; Wu, X. A 3-D Multi-Object Path Planning Method for Electric Vehicle Considering the Energy Consumption and Distance. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 7508–7520. [\[CrossRef\]](#)
18. Wang, H.; Qi, X.; Lou, S.; Jing, J.; He, H.; Liu, W. An Efficient and Robust Improved A* Algorithm for Path Planning. *Symmetry* **2021**, *13*, 2213. [\[CrossRef\]](#)
19. Sangeetha, V.; Krishankumar, R.; Ravichandran, K.; Kar, S. Energy-Efficient Green Ant Colony Optimization for Path Planning in Dynamic 3d Environments. *Soft Comput.* **2021**, *25*, 4749–4769. [\[CrossRef\]](#)
20. Li, D.; Wang, L.; Cai, J.; Ma, K.; Tan, T. Research on Terminal Distance Index-Based Multi-Step Ant Colony Optimization for Mobile Robot Path Planning. *IEEE Trans. Autom. Sci. Eng.* **2022**, *20*, 2321–2337. [\[CrossRef\]](#)
21. Lee, J.; Kim, C. Development of an Adaptive AHP Path Planning Method considering the Mobile Robot Driving Environment. *IEEE Access* **2024**, *12*, 95565–95575. [\[CrossRef\]](#)
22. Szczepanski, R.; Tarczewski, T.; Erwinski, K. Energy Efficient Local Path Planning Algorithm Based on Predictive Artificial Potential Field. *IEEE Access* **2022**, *10*, 39729–39742. [\[CrossRef\]](#)
23. Wu, C.; Dai, C.; Gong, X.; Liu, Y.-J.; Wang, J.; Gu, X.D.; Wang, C.C. Energy-Efficient Coverage Path Planning for General Terrain Surfaces. *IEEE Robot. Autom. Lett.* **2019**, *4*, 2584–2591. [\[CrossRef\]](#)
24. Mohammadpour, M.; Kelouwani, S.; Gaudreau, M.-A.; Allani, B.; Zeghmi, L.; Amamou, A.; Graba, M. Energy-Efficient Local Path Planning of a Self-Guided Vehicle by Considering the Load Position. *IEEE Access* **2022**, *10*, 112669–112685. [\[CrossRef\]](#)
25. Kyaw, P.T.; Le, A.V.; Veerajagadheswar, P.; Elara, M.R.; Thu, T.T.; Nhan, N.H.K.; Van Duc, P.; Vu, M.B. Energy-Efficient Path Planning of Reconfigurable Robots in Complex Environments. *IEEE Trans. Robot.* **2022**, *38*, 2481–2494. [\[CrossRef\]](#)

26. Shan, S.; Shao, J.; Zhang, H.; Xie, S.; Sun, F. Research and Validation of Self-Driving Path Planning Algorithm Based on Optimized A*-Artificial Potential Field Method. *IEEE Sens. J.* **2024**, *24*, 24708–24722. [[CrossRef](#)]
27. Yan, Y.; Zhang, B.; Zhou, J.; Zhang, Y.; Liu, X.a. Real-Time Localization and Mapping Utilizing Multi-Sensor Fusion and Visual-Imu-Wheel Odometry for Agricultural Robots in Unstructured, Dynamic and Gps-Denied Greenhouse Environments. *Agronomy* **2022**, *12*, 1740. [[CrossRef](#)]
28. Arikumar, K.; Deepak Kumar, A.; Gadekallu, T.R.; Prathiba, S.B.; Tamilarasi, K. Real-time 3D Object Detection and Classification in Autonomous Driving Environment Using 3D Lidar and Camera Sensors. *Electronics* **2022**, *11*, 4203. [[CrossRef](#)]
29. Li, Q.; Zhuang, Y.; Huai, J.; Wang, X.; Wang, B.; Cao, Y. A Robust Data-Model Dual-Driven Fusion with Uncertainty Estimation for LiDAR-IMU Localization System. *ISPRS J. Photogramm. Remote Sens.* **2024**, *210*, 128–140. [[CrossRef](#)]
30. Liu, L.; He, J.; Ren, K.; Xiao, Z.; Hou, Y. A LiDAR-Camera Fusion 3D Object Detection Algorithm. *Information* **2022**, *13*, 169. [[CrossRef](#)]
31. Cai, Y.; Ou, Y.; Qin, T. Improving SLAM Techniques with Integrated Multi-Sensor Fusion for 3D Reconstruction. *Sensors* **2024**, *24*, 2033. [[CrossRef](#)]
32. Mendez, J.; Molina, M.; Rodriguez, N.; Cuellar, M.P.; Morales, D.P. Camera-LiDAR Multi-Level Sensor Fusion for Target Detection at the Network Edge. *Sensors* **2021**, *21*, 3992. [[CrossRef](#)] [[PubMed](#)]
33. Liu, Y.; Wang, C.; Wu, H.; Wei, Y.; Ren, M.; Zhao, C. Improved LiDAR Localization Method for Mobile Robots Based on Multi-sensing. *Remote Sens.* **2022**, *14*, 6133. [[CrossRef](#)]
34. Li, J.; Hu, Y.; Yang, S.X. A Novel Knowledge-Based Genetic Algorithm for Robot Path Planning in Complex Environments. *IEEE Trans. Evol. Comput.* **2025**, early access. [[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.