drones

Article

Efficient 3D Exploration with Distributed Multi-UAV Teams: Integrating Frontier-Based and Next-Best-View Planning

André Ribeiro and Meysam Basiri *

Institute for Systems and Robotics, Instituto Superior Técnico (IST), 1049-001 Lisboa, Portugal; andre.m.ribeiro.2000@tecnico.ulisboa.pt

* Correspondence: meysam.basiri@tecnico.ulisboa.pt

Abstract: Autonomous exploration of unknown environments poses many challenges in robotics, particularly when dealing with vast and complex landscapes. This paper presents a novel framework tailored for distributed multi-robot systems, harnessing the 3D mobility capabilities of Unmanned Aerial Vehicles (UAVs) equipped with advanced LiDAR sensors for the rapid and effective exploration of uncharted territories. The proposed approach uniquely integrates the robustness of frontier-based exploration with the precision of Next-Best-View (NBV) planning, supplemented by a distance-based assignment cooperative strategy, offering a comprehensive and adaptive strategy for these systems. Through extensive experiments conducted across distinct environments using up to three UAVs, the efficacy of the exploration planner and cooperative strategy is rigorously validated. Benchmarking against existing methods further underscores the superiority of the proposed approach. The results demonstrate successful navigation through complex 3D landscapes, showcasing the framework's capability in both single- and multi-UAV scenarios. While the benefits of employing multiple UAVs are evident, exhibiting reductions in exploration time and individual travel distance, this study also reveals findings regarding the optimal number of UAVs, particularly in smaller and wider environments.

Keywords: autonomous 3D exploration; cooperative exploration; distributed robot systems; multi-UAV exploration; 3D mapping

check for updates

Citation: Ribeiro, A.; Basiri, M. Efficient 3D Exploration with Distributed Multi-UAV Teams: Integrating Frontier-Based and Next-Best-View Planning. *Drones* **2024**, *8*, 630. https://doi.org/10.3390/drones8110630

Academic Editors: Ziyang Zhen and Maarten Uijt de Haag

Received: 9 September 2024
Revised: 17 October 2024
Accepted: 28 October 2024
Published: 31 October 2024

1. Introduction

Exploration and mapping of unknown environments using mobile ground robots and Unmanned Aerial Vehicles (UAVs) have become pivotal in a wide range of applications, including environmental monitoring [1,2], disaster response [3,4], and industrial inspection [5–7]. For such applications, multi-robot systems offer significant advantages over single robots by increasing adaptability and efficiency through task division and larger area coverage [8]. Moreover, distributed multi-robot systems further enhance resilience and operational continuity, proving invaluable in many applications. The growing demand for UAVs in such fields has driven extensive research into developing autonomous exploration algorithms, which optimize UAV trajectories to maximize area coverage, minimize exploration time, and ensure efficient resource utilization. However, achieving these goals in real time, particularly with teams of UAVs, presents significant challenges related to scalability, communication, and decision-making.

Effective exploration strategies require a delicate balance between mapping and path planning, where mapping helps robots understand their surroundings, and path planning guides them through the environment. Autonomous exploration is a rapidly evolving field that focuses on efficiently uncovering unknown regions using single or multi-robot systems [9–11]. Key exploration strategies include frontier-based exploration and sampling-based exploration, with recent developments combining these methods into hybrid approaches. Frontier-based exploration, introduced by Yamauchi [12], directs robots towards the “frontier”, the boundary between known and unknown areas. While this approach is computationally efficient and straightforward, it often fails to consider the overall context of the environment which can lead to sub-optimal exploration paths when the environment changes dynamically, such as when obstacles appear or when the targets move. Several works have thus optimized this approach for faster and safer exploration suitable for UAVs by either selecting frontiers that maximize flight speed or using hierarchical planning for global coverage [13,14].

Conversely, sampling-based strategies like Next-Best-View (NBV) planning guide robots to informative viewpoints, optimizing data collection and map accuracy. Pioneered by Connolly [15], and later adapted to 3D environments using Rapidly exploring Random Trees (RRTs) [16], these methods improve exploration in complex environments by selecting viewpoints that reduce uncertainty. While this approach can provide significant advantages in terms of information gain, it can also introduce computational overhead, particularly in real-time applications. The complexity of calculating the optimal next viewpoint often results in increased latency in decision-making, which can be critical in time-sensitive scenarios. Additionally, hybrid approaches have also been developed, whether integrating both frontier detection and information gain metrics [17], or using frontiers to optimize global coverage paths [14].

Coordination among multiple robots enhances exploration by improving coverage and adaptability. One method involves centralized systems, where a single entity coordinates actions, and distributed systems, where robots make autonomous decisions based on local information. Centralized systems, such as those described by Burgard et al. [18], offer comprehensive control but suffer from scalability and single-point-of-failure problems. One the other hand, distributed systems, as demonstrated by Yamauchi [19], are more scalable and resilient but face communication constraints and coordination complexities. Additionally, cooperative strategies, such as environmental segmentation, auction-based assignment, and Sequential Greedy Assignment (SGA), optimize resource allocation and coordination in multi-robot exploration. Environment segmentation divides the exploration area into distinct regions assigned to individual robots, improving coverage and reducing redundancy [20]. The auction-based approach allocates tasks through a bidding process, where robots submit bids based on factors like proximity and information gain, ensuring efficient resource utilization and balanced task allocation [21]. SGA sequentially assigns tasks based on priority or utility, while the Distributed Sequential Greedy Assignment (DSGA) algorithm enhances this method for large teams, using distributed computation and online planning to maintain performance in complex environments.

In recent years, deep learning (DL) and reinforcement learning (RL) techniques have been explored to address some of the limitations of traditional methods [22–24] . Deep learning models, particularly convolutional neural networks (CNNs), have shown great promise in improving perception and decision-making tasks in robot exploration. For example, deep models can be trained to predict optimal exploration paths or learn from previous exploration missions. However, these methods typically require extensive training data and high computational resources, making real-time application challenging. Additionally, they often lack transparency, which can be a drawback in critical missions where interpretability and reliability are crucial. Reinforcement learning approaches can enable robots to learn optimal exploration policies through interaction with their environment; however, they face challenges in terms of training time, exploration–exploitation trade-offs, and scalability, particularly in large, multi-robot systems where communication and coordination become key bottlenecks [25].

In summary, autonomous exploration strategies have advanced from traditional frontier-based and sampling-based methods to include state-of-the-art techniques such as deep learning, reinforcement learning, and hybrid cooperative approaches. These newer methods offer promising improvements in exploration efficiency but also introduce challenges related to scalability, computational demands, and real-time deployment, particularly in larger environments. Additionally, in specific scenarios like low-light conditions, traditional sensors such as RGB-D cameras may underperform, necessitating enhanced sensing techniques to maintain performance. Addressing these challenges remains crucial for improving exploration in diverse environments. This research addresses this gap using 3D LiDAR-equipped distributed multi-UAV systems with the following objectives:

- Develop an efficient exploration strategy;
- Design a framework for cooperative exploration;
- Leverage 3D LiDAR sensors for environmental perception.

By creating an innovative 3D exploration framework, improving resource allocation efficiency, and integrating UAVs for 3D movement, this work advances the field of autonomous exploration. This work extends our previous conference paper [26] by improving the proposed method and expanding the experimental results, providing a more thorough validation of the system and additional comparisons with relevant benchmarks. The developed code and additional resources related to this work are available on GitHub at the following link: https://github.com/andremribeiro/thesis (accessed on 1 October 2024).

Finally, this work is organized as follows: Section 2 analyzes the proposed framework, Section 3 presents simulation experiments and results, and Section 4 concludes the paper by summarizing key findings and suggesting future research directions.

2. Materials and Methods

The proposed framework for autonomous UAV exploration, shown in Figure 1, utilizes a distributed and embedded system involving multiple UAVs equipped with 3D LiDAR sensors and GPS for environmental perception and localization.

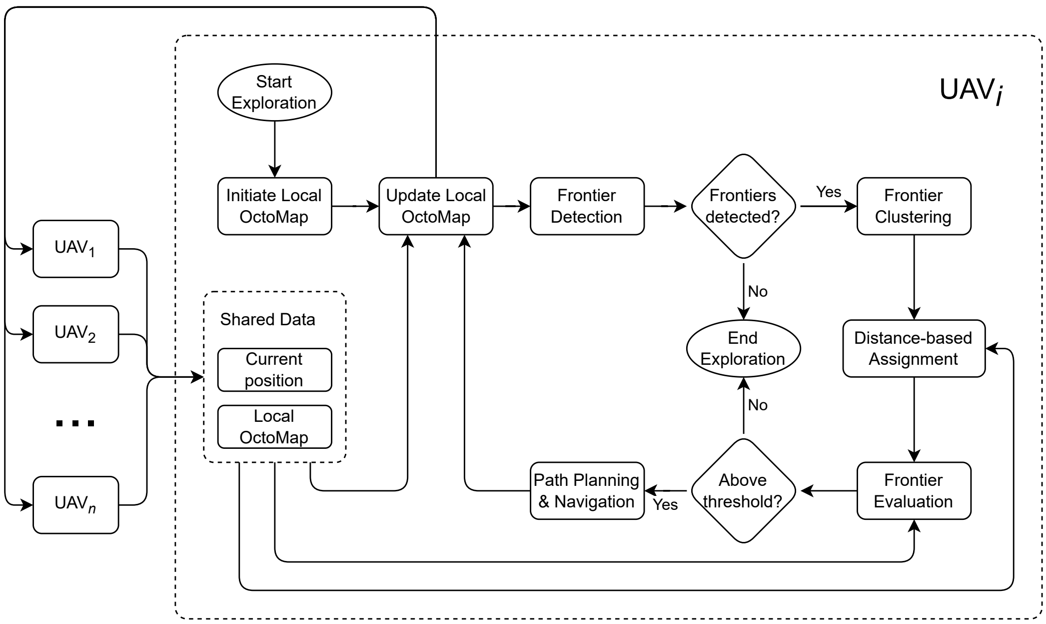


Figure 1. Proposed framework for UAV$_i$.

In this system, each UAV independently initializes an OctoMap [27] to understand its surroundings, and periodically checks for neighboring robots to share and merge OctoMaps, creating a unified map of the environment. Subsequently, frontier detection identifies boundary points between explored and unexplored regions, which are then clustered and assigned to UAVs through a simplified auction-based approach to prevent overlapping efforts. Each UAV evaluates these frontiers based on various criteria and navigates towards optimal waypoints, continuously updating their paths as new data become available.

The system operates independently of any SLAM algorithm, relying on LiDAR and GPS data for mapping and position estimation. It prioritizes flexibility and can integrate with occupancy grid-based path planning algorithms, ensuring effective navigation in diverse environments. The UAVs maintain both local and global reference frames to facilitate accurate global positioning and mapping, with transformations applied regarding a pre-defined common GPS origin to ensure consistency across all robots. This cooperative strategy streamlines communication by focusing on the exchange of global positions and OctoMaps, enabling efficient and coordinated exploration.

2.1. OctoMap Integration

This section discusses the integration of OctoMap [27], a probabilistic 3D occupancy grid mapping framework, within this exploration planner. Using Octrees for data representation, OctoMap excels in generating volumetric 3D models of environments, which are essential for tasks like map building, frontier detection, and collision avoidance. This integration aids the algorithm's spatial understanding, improving both efficiency and safety.

Octrees are a key part of the OctoMap framework, efficiently organizing 3D space. An Octree is a tree structure where each internal node splits into eight children (see Figure 2). This partitions 3D space, enabling fast spatial indexing and data access.

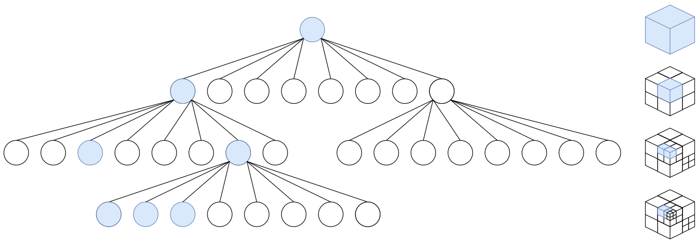


Figure 2. Octree structure.

In this study, Octrees are used to store and retrieve 3D environmental data. Each node represents a cubic volume, or voxel, with its depth corresponding to the resolution of the space. This hierarchical setup balances detail with computational efficiency.

The Octree's resolution is adaptive. Smaller areas needing detail, like indoor environments, use finer resolutions with smaller voxels, while open spaces with fewer obstacles are coarser. The maximum depth is usually 16, setting the smallest voxel size and enabling precise environment representation. This adaptability optimizes memory and computation by adjusting detail based on environmental complexity, crucial for real-time applications.

As previously mentioned, OctoMap employs a grid structure composed of voxels v. These voxels represent specific cubic volumes within the environment and encapsulate a probability value indicating the likelihood of occupancy within that space. This probability is derived from sensor data, the LiDAR's point cloud in this study, and a probabilistic sensor model, which updates with new sensor information, as defined in (1). This framework handles the inherent uncertainty present in real-world sensor data, facilitating informed decision-making even with noisy or incomplete sensor readings.

$$P(o_v \mid z_{1:t}) = \left[1 + \frac{1 - P(o_v \mid z_t)}{P(o_v \mid z_t)} \frac{1 - P(o_v \mid z_{1:t-1})}{P(o_v \mid z_{1:t-1})} \frac{P(o_v)}{1 - P(o_v)}\right]^{-1}, \tag{1}$$

where $P(o_v|z_{1:t})$ is the probability of occupancy of voxel v given all sensor measurements $z_{1:t}$, z_t is the current sensor measurement, $P(o_v)$ is a prior probability, $P(o_v \mid z_{1:t-1})$ is the prior estimate of voxel occupancy, and $P(o_v \mid z_t)$ is the probability of occupancy of voxel v given the current sensor measurement.

Furthermore, OctoMap assumes a uniform prior probability, resulting in $P(o_v) = 0.5$, and uses logarithmic notation to convert the multiplication of probabilities into additions, allowing for faster updates. The corresponding equation for updating the log-odds representation $L(o_v \mid z_{1:t})$ is defined in (2):

$$L(o_v \mid z_{1:t}) = L(o_v \mid z_{1:t-1}) + L(o_v \mid z_t), \tag{2}$$

where $L(o_v \mid z_{1:t})$ is the log-odds of occupancy given all sensor measurements up to time t, $L(o_v \mid z_{1:t-1})$ is the log-odds of occupancy given all sensor measurements up to time $t-1$, and $L(o_v \mid z_t)$ is the log-odds of occupancy given the current sensor measurement. This logarithmic approach simplifies the update process, making it computationally efficient.

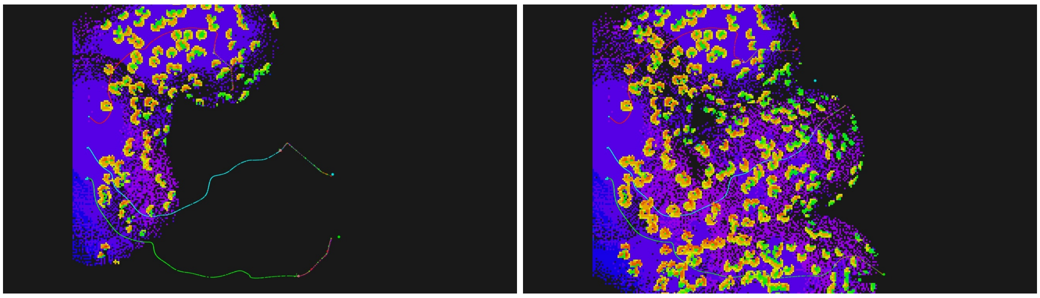
The occupancy status of each voxel categorizes the three-dimensional space $V \subset \mathbb{R}^3$ into three distinct subspaces: free $V_{free} \subset V$, occupied $V_{occ} \subset V$, and unknown $V_{un} \subset V$. The union of these subspaces defines the entirety of the environment $V \equiv V_{free} \cup V_{occ} \cup V_{un}$. Free space comprises voxels determined to be unobstructed, while occupied space denotes those containing obstacles. The unknown subspace represents areas where occupancy status cannot be definitively determined. An environment is considered fully explored when the unknown subspace is empty, or $V_{un} = \emptyset$. However, certain areas may remain unobserved due to sensor limitations or occlusion, resulting in an approximation rather than an exact representation of the environment.

Moreover, through collaborative mapping, UAVs share their individual maps with each other to create a comprehensive and accurate representation of the environment. In the context of this approach, each UAV updates its own map by incorporating the log-odds values from the received maps. If a voxel does not exist in the local map, it is created and initialized with the received log-odds value. The equation for updating the log-odds $L_i(o_v \mid z_{1:t})$ of a voxel v for UAV i after receiving information from UAV j is given by

$$L_i(o_v \mid z_{1:t}) = L_i(o_v \mid z_{1:t-1}) + L_j(o_v \mid z_{1:t}), \tag{3}$$

where $L_j(o_v \mid z_{1:t})$ is the log-odds value received from UAV j. This equation ensures that the updates are consistent and the occupancy probabilities are correctly propagated across the network of collaborating UAVs.

Through this collaborative process, the UAVs achieve a more efficient and effective exploration of the environment, minimizing redundant efforts and maximizing coverage. The efficiency gained through collaboration is particularly valuable in large or complex environments, where single-UAV mapping might be impractical or time-consuming. This process is depicted in Figure 3.



(a) Map before the top UAV enters communication range with the middle UAV.

(b) Updated map after merging with the middle UAV's map, which already includes the bottom UAV's map.

Figure 3. Example of a UAV updating its own OctoMap when entering communication range with other UAVs. The maps show the top UAV's view in a simulated forest environment, with its trajectory in red.

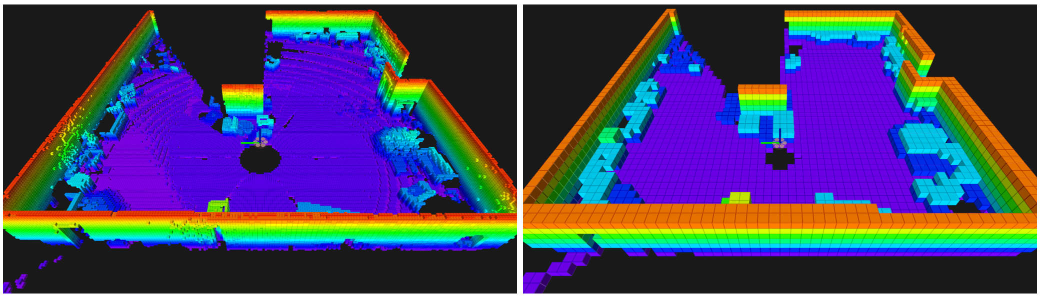
2.2. Frontier Detection

The proposed exploration strategy uses a frontier-based approach as a first step for independent exploration of unknown environments. Frontiers, defined as the boundaries between known and unknown areas, are identified using an occupancy grid divided into discrete voxels. In this work, a frontier voxel is defined as a free voxel adjacent to at least one unknown neighbor (see Equation (4)).

$$v_f = \{v \in V_{free} : \exists v' \in Neighbors(v) \in V_{un}\}, \tag{4}$$

Here, v_f denotes a frontier voxel, v represents any voxel, $Neighbors(v)$ denotes its 26 neighboring voxels, and v' is a neighboring voxel in the unknown set. Frontier voxels are identified by their center, called frontier points.

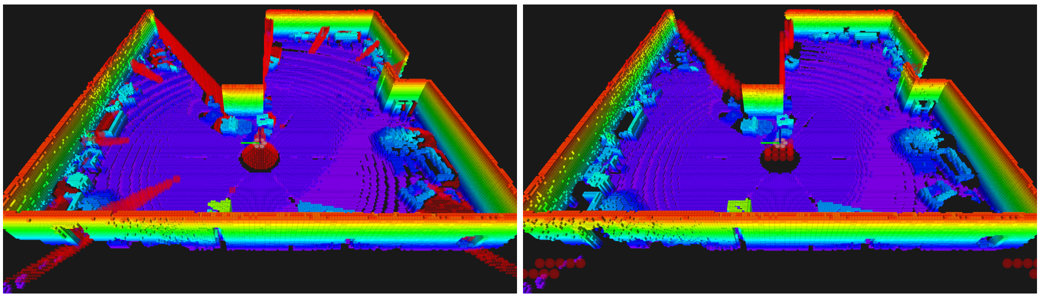
As a means of streamlining exploration, this work leverages the OctoMap's Octree structure. In this structure, each node in the Octree typically reaches a maximum depth $d_{max} = 16$, determining the map's resolution r_{map}. Figure 4 shows an indoor environment map at two different Octree depths.



(**a**) Octree depth $d = 16$. (**b**) Octree depth $d = 14$.

Figure 4. Example of an OctoMap with a $r_{map} = 0.1$ m at different depths.

The proposed optimization lies in performing detection at a reduced depth d_{exp} within the Octree. This approach conservatively uses the maximum child occupancy for inner nodes, reducing the number of cells processed and closing off narrow passages that may hinder navigation (see Figure 5).



(**a**) Octree depth $d = 16$. (**b**) Octree depth $d = 14$.

Figure 5. Frontier points (red spheres) detected at different depths.

The OctoMap depth used for exploration is adaptable to the environment: coarser resolutions for open-spaced environments and finer resolutions for cluttered layouts. Depth adjustment ensures an optimal balance between computational efficiency and exploration effectiveness. However, reducing the depth excessively can prematurely close off optimal frontiers due to oversimplification. By using OctoMap's multi-resolution capabilities, this strategy enhances exploration efficiency across diverse environments.

The detailed steps of this algorithm are provided in Algorithm 1.

Algorithm 1 Frontier detection at d_{exp}

procedure FRONTIERDETECTION
  $frontiers \leftarrow \emptyset$
  $neighbors \leftarrow \emptyset$
  for each cell **in** octree at depth d_{exp} **do**
    $current\_key \leftarrow$ getKey(cell)
    $current\_point \leftarrow$ keyToCoord$(current_key, d_{\text{exp}})$

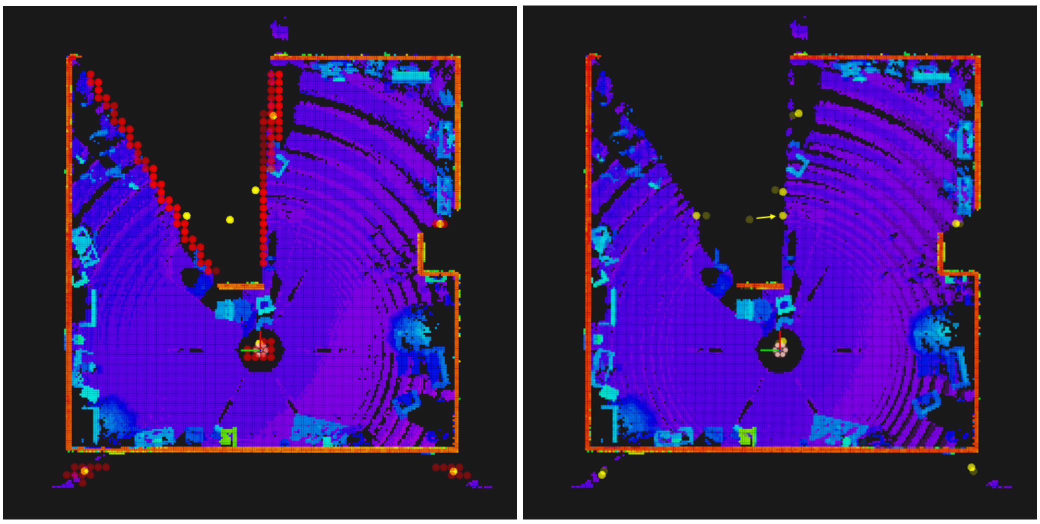
Algorithm 1 *Cont.*

    if current point is out of bounds **then**
        skip to next cell
    end if
    $current\_node \leftarrow \text{search}(current\_key, d_{\text{exp}})$
    if current node is null **or** cell is occupied **then**
        skip to next cell
    end if
    $unknown \leftarrow$ **false**
    getNeighbor(current_key, neighbors)
    for each neighbor **in** neighbors **do**
        if neighbor is out of bounds **then**
            skip to next neighbor
        end if
        $neighbor\_node \leftarrow \text{search}(neighbor, d_{\text{exp}})$
        if neighbor node is null **then**
            $unknown \leftarrow$ **true**
            stop checking neighbors
        end if
    end for
    if $unknown$ is **true then**
        add $current\_key$ to $frontiers$
    end if
  end for
end procedure

2.3. Frontier Clustering

In autonomous exploration, where efficient path planning is crucial, clustering algorithms offer valuable tools for grouping frontier points. Mean-shift clustering, introduced in [28], stands out for its flexibility in dynamically identifying clusters based on spatial proximity without needing a predetermined number of clusters.

Mean-shift clustering iteratively shifts points towards dense regions of the data distribution, converging to cluster centers (see Figure 6a). The bandwidth parameter in the Gaussian kernel function controls the scale of the kernel and influences the range over which neighboring frontier voxels contribute to the computation of these cluster centers.



(**a**) Frontier points (red spheres) and cluster centers (yellow spheres) obtained using mean-shift clustering.

(**b**) Frontier candidates derived from cluster centers for exploration.

Figure 6. Frontier clustering process.

The computational complexity primarily stems from identifying a voxel's neighbors and selecting an optimal bandwidth. Smaller bandwidths yield finer clustering granularity, while larger bandwidths consider more distant neighbors, which increases computational cost.

After determining cluster centers, the next step involves selecting representative frontier candidates for exploration. Frontier candidates are chosen by selecting the nearest frontier voxel to each cluster center, as depicted in Figure 6b.This ensures the frontier candidate is a free voxel and is thus safe for navigation.

2.4. Frontier Evaluation

Efficient frontier evaluation is critical for autonomous exploration systems, determining the most promising targets by assessing factors like information gain, proximity to other UAVs, and the cost of reaching a candidate. This process ensures UAVs make informed decisions, maximizing exploration efficiency and coverage.

The initial phase employs a simplified auction-based concept to assign frontier candidates to UAVs in a decentralized manner. Each frontier candidate is assigned to a UAV based on proximity, simulating a competitive bidding scenario. This distance-based assignment promotes balanced coverage and minimizes overlap, streamlining the process and facilitating quick decision-making.

After assignment, each UAV evaluates frontier candidates based on potential information gain, the cost of reaching them, and the distances of other UAVs to the candidate. This is achieved using a utility function adapted from [29], defined as

$$U_i(v_c) = \frac{100 \cdot I(v_c)}{e^{\lambda\left(d(\mathbf{p}i, v_c) + \frac{N}{\sum_{j=1}^{N} d(\mathbf{p}_j, v_c)}\right)}}, \tag{5}$$

where $U_i(v_c)$ is the utility of exploring voxel v_c for UAV i, $I(v_c)$ is the information gain, $d(\mathbf{p}_i, v_c)$ is the distance between UAV i and voxel v_c, $d(\mathbf{p}_j, v_c)$ is the distance between neighboring UAV j and voxel v_c, N is the number of UAVs within communication range, and λ is a positive constant scaling the influence of distance and neighboring UAVs.

The utility function incorporates several factors. Information gain $I(v_c)$ represents potential knowledge gain by exploring voxel v_c, estimating the proportion of unknown voxels within the sensor's field of view. Distance influence combines the distance penalty for the UAV and the influence of neighboring UAVs, encouraging UAVs to spread out and explore less-crowded areas. Exponential decay balances information gain against the cost of movement and proximity to other UAVs.

Each UAV computes a utility score for every frontier candidate and selects the one with the highest score as the next exploration target. To ensure efficient exploration, a minimum utility score threshold ($U_{\min}$) is set, scaling with the number of UAVs:

$$U_{\min} = \frac{1}{e^{N_{\text{UAV}}}} \tag{6}$$

Additionally, a minimum coverage threshold (typically over 90%) ensures thorough exploration.

Finally, when a UAV encounters a local minima (unreachable waypoint), the system dynamically adjusts waypoint priority. Waypoints that the path planner cannot reach are added to a local minima list, and their utility scores are reduced each time they are found unreachable, reducing their priority but keeping them in consideration for future attempts. This mechanism incentivizes exploration towards reachable waypoints with higher scores, avoiding getting stuck in local minima.

The algorithm used for this evaluation is detailed in Algorithm 2.

Algorithm 2 Frontier evaluation at d_{exp}

```
procedure FRONTIEREVALUATION
    best_score ← 0
    best_candidate ← ∅
    for each candidate in candidates do
        candidate_point ← keyToCoord(candidate)
        distance ← calculateDistance(position, candidate_point)
        distance_sum ← 0
        skip_candidate ← false
        for each other_uav in other_uavs do
            distance_sum ← distance_sum + other_uav.distance
            if distance between other UAV and candidate is less than distance then
                skip_candidate ← true
                break
            end if
        end for
        if skip_candidate is true then
            continue
        end if
        info_gain ← calculateInfoGain(candidate_point)
        utility_score ← 100 × info_gain × exp(−λ × (distance + N / distance_sum))
        if candidate is in local minima then
            reduce utility_score by a factor based on local minima
        end if
        if utility_score > best_score then
            best_score ← utility_score
            best_candidate ← candidate_point
        end if
    end for
    if no best candidate found then
        waypoint ← position
        waypoint_score ← 0
    else
        waypoint ← best_candidate
        waypoint_score ← best_score
    end if
end procedure
```

2.5. Path Planning

Integrating frontier information into path planning and navigation is essential for autonomous exploration. This section details how frontier data influence navigation decisions and describes the path planning algorithm used to guide UAVs efficiently and safely to their selected waypoints. After selecting its next waypoint, a UAV uses the MRS SubT Planner [5] to plan a path to that frontier point.

The A* algorithm finds the optimal path from a start node to a goal node by minimizing the total cost $f(n)$ at each node n, where

$$f(n) = g(n) + h(n) \tag{7}$$

Here, $g(n)$ is the cost from the start node to n, and $h(n)$ is a heuristic function estimating the cost from n to the goal node. In this path planning algorithm, the heuristic used is the Euclidean distance between node n and the goal node. This provides an informed estimate of the shortest possible path in a 3D space, guiding the UAV efficiently towards the goal.

The planner uses an occupancy grid map to calculate $g(n)$ based on the traversable and obstructed areas, and adapts $h(n)$ for UAV navigation by incorporating environmental factors such as distance to obstacles, sensor range, and terrain features. It considers the 26 adjacent voxels as potential neighbors for each voxel, providing more flexibility in exploring the surrounding space.

In the case of UAV navigation, dynamic obstacles and real-time environmental changes are considered, requiring frequent re-evaluation of the path. The MRS SubT Planner modifies the standard A* algorithm by introducing dynamic re-planning, which recalculates $f(n)$ as new data from the UAV's sensors become available. This ensures that the UAV can adapt to new obstacles or terrain changes.

Once a path is planned, the UAV follows it using its navigation system, which continuously monitors its position and adjusts its movements to stay on the planned path. Real-time feedback from the UAV's onboard sensors allows it to avoid obstacles and make necessary corrections.

The exploration process continues until either no frontiers are left to explore or no waypoints with a utility score above the minimum threshold remain, indicating that further exploration is unlikely to yield significant new information and approximate map completion. This integration of frontier information into path planning and navigation ensures efficient and thorough exploration, resulting in a comprehensive and accurate map.

The exploration process continues until no frontiers with utility scores above a certain threshold remain, at which point the UAV concludes that further exploration is unlikely to provide significant new information. Once the path is determined, the UAV uses real-time feedback from its navigation system to follow the planned trajectory, adjusting its movements to avoid obstacles and maintain path feasibility.

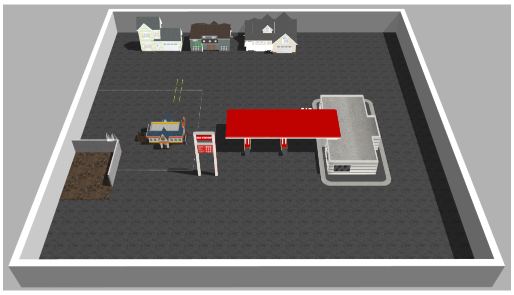
3. Results

For the exploration experiments, the comprehensive setup is described to ensure transparency and reproducibility. Simulations were performed within the Robot Operating System (ROS) framework and using the well-established MRS Multirotor Simulator [30], which is widely accepted by the research community. This simulator operates with a Software-In-The-Loop (SITL) configuration, utilizing actual PX4 autopilot software, Gazebo, and the Mavros framework. Such a setup is shown to have a high level of realism, closely mimicking real-world UAV behaviors, thus providing a robust environment for testing our algorithms.

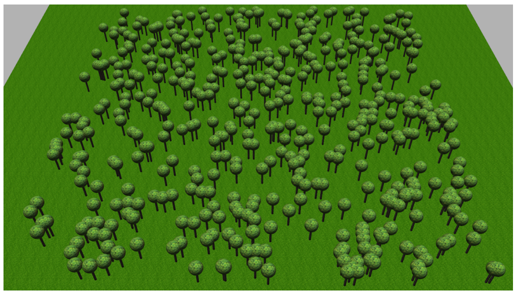
Path planning was facilitated by the MRS OctoMap Planner, which utilized a dynamic A* search algorithm. OctoMap was used for environmental mapping, enabling efficient 3D mapping within the Gazebo environment. The simulated UAVs were equipped with Ouster OS0-128 LiDAR sensors and RTK GPS systems, ensuring high-resolution 3D data collection and precise positioning. Simulations were conducted on an ASUS VivoBook S14/S15 laptop running ROS Noetic on Ubuntu 20.04 LTS.

The evaluation of the proposed approach has been conducted through five simulation studies for each n-UAV exploration scenario. Moreover, the exploration strategies were tested in two distinct environments: Test City (urban) and Forest (natural). These environments, depicted in Figure 7, provided diverse scenarios for evaluating our methodology's robustness and effectiveness.

In all simulation scenarios, the UAVs were initialized from the same side of the environment, with a spacing of 2.5 m between them in the urban environment and 5 m in the forested environment. This configuration reflects a realistic exploration scenario, where UAVs are typically deployed from a common starting point due to the unknown nature of the environment. Spreading the UAVs beforehand would not be practical or feasible, as the boundaries and layout of the environment are not known at the start.



(**a**) Test City.



(**b**) Forest.

Figure 7. Simulated environments.

Additionally, in order to ensure fair evaluation, consistent parameters across simulations were established as indicated in Table 1.

Table 1. Key system parameters.

Parameters	Value
Collision avoidance distance (m)	1.0
Take-off height (m)	2.0
Maximum speed (m/s)	4.0
Minimum sensor range (m)	0.7
Maximum sensor range (m)	20.0
Vertical FOV	$\pi/2$
Communication range (m)	40.0
Update rate (Hz)	2.0
Map resolution (m)	0.5
Minimum coverage threshold (%)	90

For evaluating exploration performance, a comprehensive set of metrics was used such as exploration time, path length, coverage, exploration rate (discovered volume per time), and path efficiency (discovered volume per distance). These metrics provided insights into the efficiency, effectiveness, and adaptability of the exploration strategy across various experimental scenarios.

3.1. Ablation Studies

This section is aimed at understanding how key tuning parameters affect the performance of the exploration planner. The parameters under study are OctoMap depth, trade-off constant λ, and mean-shift clustering bandwidth. The impact of these parameters is analyzed in different testing environments, with discussions on observed effects and key metrics.

Firstly, the optimal depth of the Octree for exploration is determined. The study evaluates various depths, focusing on exploration time, 3D path length, and coverage percentage, represented in Table 2.

Table 2. Ablation study of exploration depth.

Scene	Depth	Exploration Time (s)		Path Length (m)		Coverage (%)	
		Avg	Std	Avg	Std	Avg	Std
Test City	15	**49.34**	**2.42**	**109.26**	**2.42**	90.46	**0.30**
	14	85.68	18.26	141.79	18.83	**92.54**	1.41
Forest	15	**213.42**	**21.68**	**361.80**	**22.26**	94.21	**2.06**
	14	225.37	36.54	376.66	55.10	**95.37**	2.54

The bold numbers here simply is showing the smallest value between the trails and is trivials.

The results show that a depth of 15 generally leads to minimized exploration time and path length. However, depth 14 exhibits slightly higher coverage percentages. Further analysis reveals that depth 15 facilitates faster and more efficient exploration, as evidenced by higher exploration rates and path efficiency metrics. This is attributed to the availability of more detailed exploration options. However, depth 14 performs better in reaching initial coverage levels, especially in complex environments like forests.

While depth 15 offers better exploration efficiency, it demands significantly higher computational resources, posing challenges in larger or dynamic environments. On the other hand, depth 14 provides a more practical and scalable solution, making it suitable for real-time applications and multi-robot exploration scenarios. In conclusion, depth 15 strikes a balance between exploration efficiency and computational feasibility for smaller or moderate-sized maps. Meanwhile, depth 14 offers a more practical solution for larger or dynamic environments, despite potential trade-offs in exploration efficiency. Depths 16 and 13 are deemed unsuitable for real-time and comprehensive exploration, respectively.

Secondly, the tuning constant λ was examined. This constant dictates the balance between information gain and distance cost in the evaluation process.

Analyzing Table 3, the experiments revealed nuanced effects of λ on exploration metrics. In the urban environment, $\lambda = 0.10$ demonstrated superior performance, achieving the shortest exploration time (85.68 s) and path length (141.79 m) while maintaining high coverage (92.54%). Conversely, $\lambda = 0.05$ prioritized coverage but required significantly more time and distance. In the forested area, $\lambda = 0.20$ yielded the shortest exploration time (218.05 s) and path length (362.03 m), while $\lambda = 0.10$ achieved the highest coverage (97.87%), albeit with longer exploration metrics.

Table 3. Ablation study of λ.

Scene	λ	Exploration Time (s)		Path Length (m)		Coverage (%)	
		Avg	Std	Avg	Std	Avg	Std
Test City	0.05	143.56	14.81	265.98	13.65	**95.94**	**0.33**
	0.10	**85.68**	18.26	**141.79**	18.83	92.54	1.41
	0.15	114.60	16.06	172.76	21.03	92.61	1.50
	0.20	120.35	**10.93**	172.33	**13.45**	92.96	1.73
	0.25	113.88	17.91	173.28	15.25	92.14	0.84
Forest	0.05	247.88	34.47	427.05	53.35	96.75	1.56
	0.10	278.24	24.55	457.19	42.58	**97.87**	1.45
	0.15	225.37	36.54	376.66	55.10	95.37	2.54
	0.20	**218.05**	29.26	**362.03**	49.87	92.75	2.63
	0.25	221.07	**24.12**	368.97	**29.01**	90.98	**0.80**

The bold numbers here simply is showing the best value between the trails and is trivial.

Efficiency metrics further underscored the influence of λ. In urban settings, $\lambda = 0.10$ exhibited the highest exploration rate (163.23 m^3/s) and path efficiency (95.50 m^3/m), striking a balance between speed and coverage. Conversely, in the forest, $\lambda = 0.15$ provided the best exploration rate (140.48 m^3/s) and path efficiency (83.73 m^3/m), indicating a different optimal balance in complex environments. Our analysis of coverage milestones highlighted the effectiveness of different λ values. $\lambda = 0.05$ consistently achieved the fastest coverage milestones in both environments, indicating rapid initial exploration. However, higher λ values also performed well in the forest, demonstrating effectiveness in covering large areas despite longer exploration times. These findings underscore the importance of selecting an appropriate λ value based on environmental characteristics. While lower λ values strike a balance between efficiency and coverage in simpler environments, slightly higher values may be preferable in more complex settings to optimize exploration strategies.

Thirdly, the influence of mean-shift clustering bandwidth (h) was explored across varied bandwidth values ($h = 1.0, 2.0, 3.0$). Evaluation encompassed key metrics including exploration time, path length, coverage, efficiency, and computation time.

The results presented in Table 4 indicate that $h = 2.0$ emerged as the optimal choice across both urban and forest environments. In the urban setting, this bandwidth demonstrated the shortest exploration time (85.68 s) and path length (141.79 m), alongside notable coverage (92.54%). Similarly, in the forest environment, $h = 2.0$ showcased superior performance, minimizing exploration time (225.37 s) and path length (376.66 m), while ensuring efficient coverage.

Table 4. Ablation study of mean-shift clustering bandwidth.

Scene	h	Exploration Time (s)		Path Length (m)		Coverage (%)	
		Avg	Std	Avg	Std	Avg	Std
Test City	1.0	101.02	14.84	152.93	12.08	91.53	1.66
	2.0	**85.68**	18.26	**141.79**	18.83	92.54	1.41
	3.0	91.10	**7.36**	153.56	**8.46**	**93.41**	**1.23**
Forest	1.0	244.39	**27.09**	405.61	**33.48**	95.25	2.24
	2.0	**225.37**	36.54	**376.66**	55.10	95.37	2.54
	3.0	240.03	39.05	393.15	65.54	**96.13**	**2.01**

The bold numbers here simply is showing the best value between the trials.

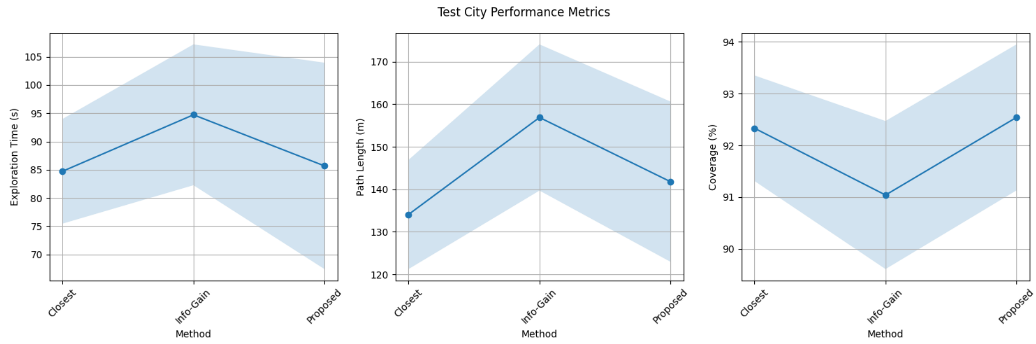
Efficiency metrics further emphasize its effectiveness, exhibiting the highest exploration rates and path efficiencies in both environments. Moreover, coverage analysis revealed $h = 2.0$ as proficient in rapid initial exploration and gap-filling, achieving significant coverage milestones swiftly. Computation time analysis indicated $h = 2.0$ as the most computationally efficient option, particularly suited for real-time applications, ensuring optimal resource utilization. The ablation study suggests $h = 2.0$ as the optimal mean-shift clustering bandwidth, providing a balanced trade-off between exploration performance, efficiency, coverage, and computational feasibility. Its adaptability to varied environmental complexities shows its robustness for diverse exploration scenarios. Adjusting h enables tailored exploration strategies, ensuring optimal performance under different constraints.

In summary, the results of the ablation studies provide insights into the practical impact of key parameters in real-world UAV exploration scenarios. The choice of OctoMap depth plays a vital role in balancing computational efficiency and mapping accuracy. Finer depths allow for detailed navigation in cluttered environments but demand more computational resources, while coarser depths provide faster performance suitable for larger, open environments. The λ parameter governs the trade-off between information gain and distance cost and is environment-dependent; lower values prioritize rapid coverage in time-sensitive missions, while higher values enhance precision in complex terrains. Similarly, the mean-shift bandwidth affects frontier clustering granularity, where smaller bandwidths excel in cluttered environments, and larger bandwidths improve computational efficiency in open spaces. These findings underscore the need for adaptable parameter tuning to optimize UAV performance in diverse operational conditions.

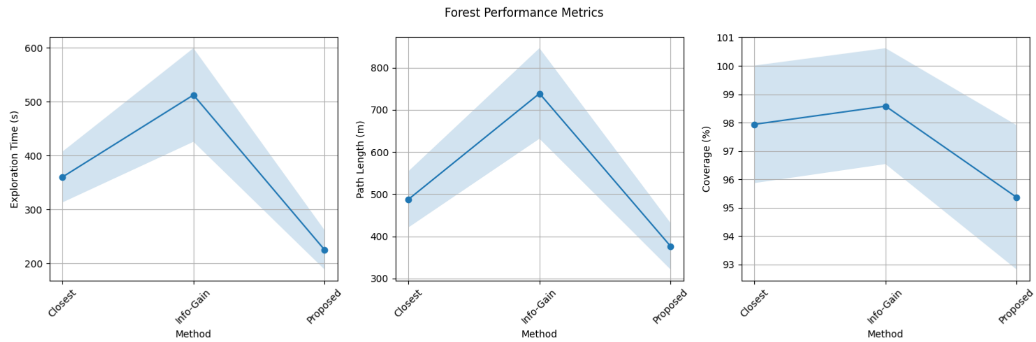
3.2. Benchmarking Analysis

In order to assess the efficacy of the proposed exploration strategy for 3D LiDAR exploration via UAVs, a benchmarking analysis was conducted to compare it against two established methods: the classical frontier-based approach (selecting the closest frontier point) and a greedy information gain strategy. The objective was to comprehensively assess performance across diverse exploration metrics in both urban and forest environments. Figure 8 presents performance metrics, including exploration time, path length, and coverage, for each method in the Test City and Forest environments.

In these simulations, the benchmark methods used for comparison do not incorporate a specific stopping criterion based on the value or importance of remaining frontier points. Instead, they continue exploring until all frontier points are exhausted, regardless of whether the additional exploration would yield significant new information. In contrast, the proposed method includes a mechanism to evaluate the worthiness of frontier points before continuing exploration, leading to a more efficient stopping condition once diminishing returns are detected.



(**a**) Test City.



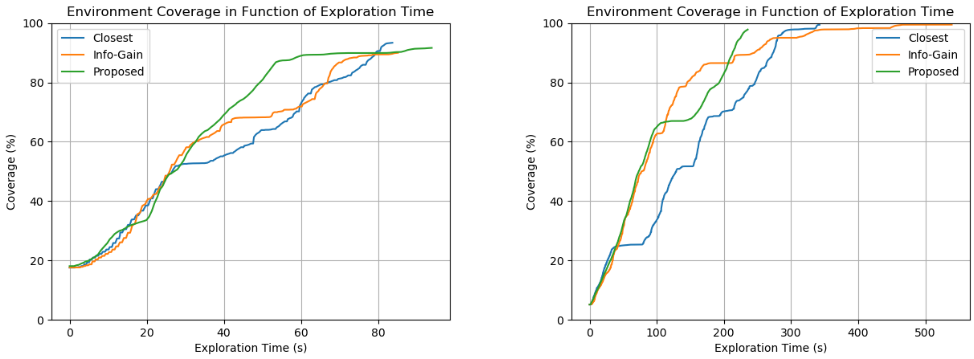
(**b**) Forest.

Figure 8. One-UAV exploration benchmarking analysis. The blue line represents the mean value, while the light blue shading indicates the standard deviation of 5 simulation runs.

In the Test City, the closest frontier method demonstrated an average exploration time of 84.72 s, path length of 134.08 m, and coverage of 92.33%. In contrast, the information gain method exhibited a slightly longer exploration time (94.72 s) and path length (156.88 m) but comparable coverage (91.04%). The proposed method struck a balance between efficiency and coverage, with an exploration time of 85.68 s, path length of 141.79 m, and coverage of 92.54%. In the Forest environment, the proposed method outperformed both benchmarks, achieving an exploration time of 225.37 s, path length of 376.66 m, and coverage of 95.37%.

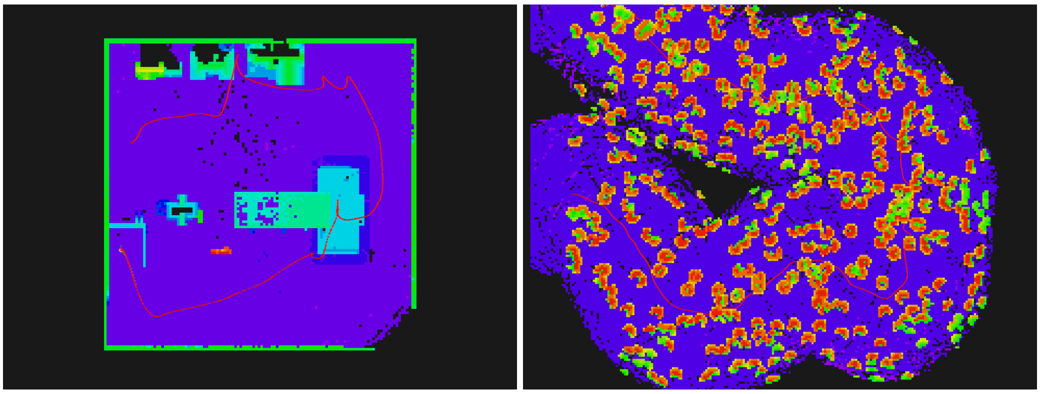
Moreover, the proposed method demonstrated superior exploration rates and path efficiencies in both environments. In the Test City, it achieved the highest exploration rate (163.23 m^3/s) and path efficiency (95.50 m^3/m). Similarly, in the Forest environment, the proposed method showcased superior performance with an exploration rate of 158.19 m^3/s and a path efficiency of 104.81 m^3/m. Regarding coverage metrics, the closest frontier method was fastest to reach 50% and 75% coverage in the Test City, while the proposed method demonstrated consistent performance across all coverage milestones. In the Forest environment, the proposed method surpassed the benchmarks in reaching 75% and 90% coverage.

Figures 9 and 10 depict the progressive coverage of the environment over time and the UAV trajectory overlaid on the map for each scenario, respectively. These figures show the proposed method's effectiveness in navigating complex environments, ensuring comprehensive coverage.



(**a**) Test City. (**b**) Forest.

Figure 9. Example of combined coverage of the environment over time for each scenario.



(**a**) Test City. (**b**) Forest.

Figure 10. Example of UAV trajectory overlaid on the map of the environment for each scenario.

The proposed method maintains consistent flight height and speed, essential for thorough coverage while minimizing exploration time. These adjustments demonstrate the UAV's adaptability to terrain and obstacles.

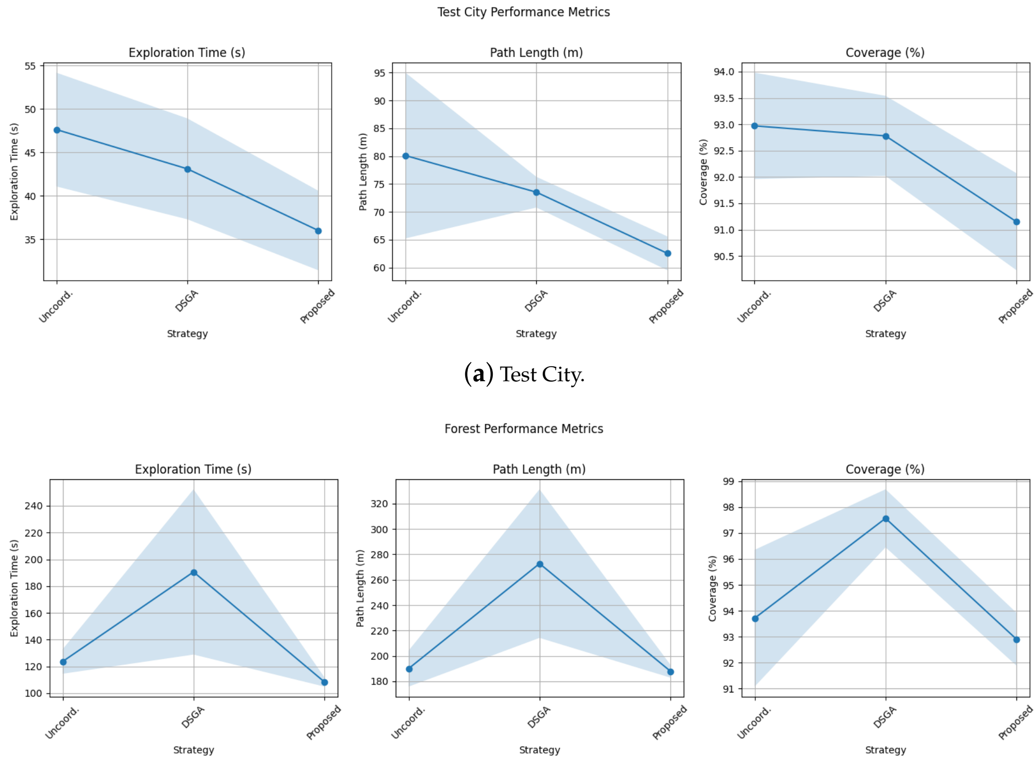
In summary, the benchmarking analysis validates the proposed exploration strategy's effectiveness across multiple metrics and environments. Its balanced performance, robustness, and adaptability make it a promising solution for 3D LiDAR exploration using UAVs, with potential applications in various domains.

3.3. Multi-UAV Exploration

This section presents a comprehensive benchmarking analysis of multi-UAV exploration strategies. The performance, efficiency, and coverage of three methods—Uncoordinated, DSGA, and the proposed strategy—are compared using both two and three UAVs in simulated environments. Uncoordinated refers to UAVs sharing environmental information but exploring independently without cooperative tactics. DSGA and the proposed strategy incorporate varying degrees of cooperation and optimization.

For these simulations, occasional variations in the final coverage values were observed, particularly when UAVs share their maps after being out of communication range. Once communication is re-established, the exchange of map data between UAVs can cause a sudden increase in the explored area, resulting in a noticeable jump in the coverage percentage. This explains the higher and more varied final coverage metrics in certain scenarios.

Starting with the addition of a second UAV, comparing various exploration strategies in different environments, Figure 11 highlights performance metrics like exploration time, path length, and coverage percentage.

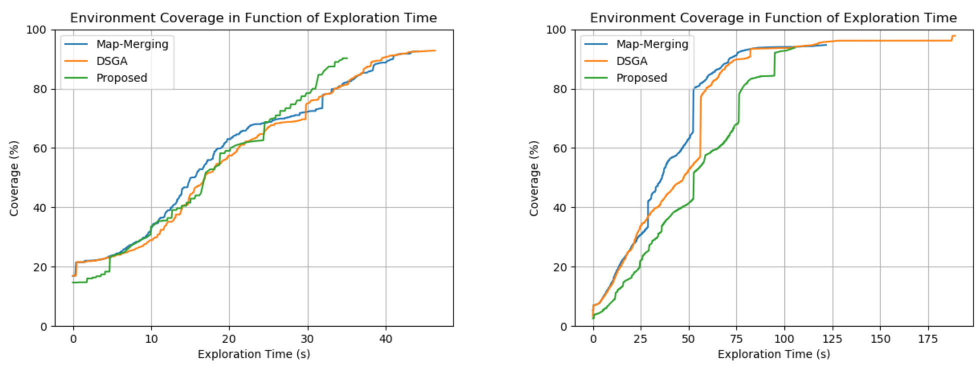


(**a**) Test City.

(**b**) Forest.

Figure 11. Two-UAV exploration benchmarking analysis. The blue line represents the mean value, while the light blue shading indicates the standard deviation of 5 simulation runs.

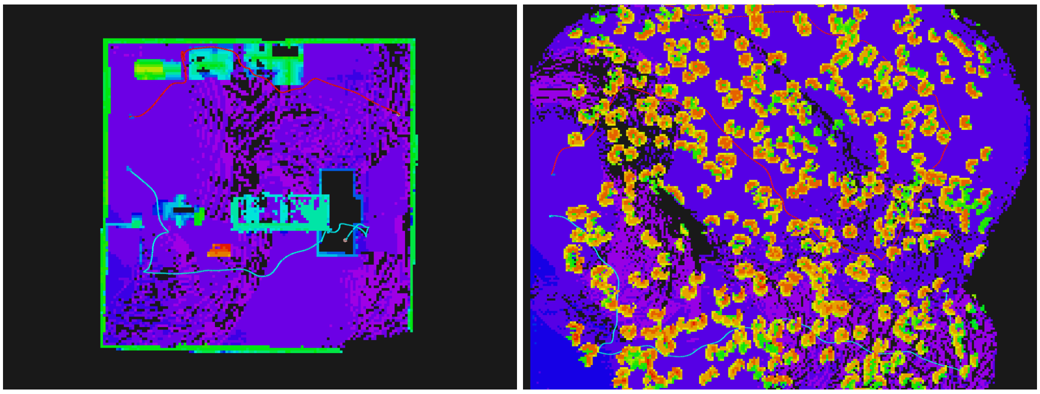
The proposed strategy outperforms others, especially in urban settings, with the shortest exploration time (36.01 s) and path length (62.56 m), though coverage percentages are similar. In Forest environments, the proposed strategy also excels, achieving the shortest exploration time (108.33 s) and path length (187.94 m). Coverage is slightly lower but competitive. Regarding efficiency metrics, the proposed strategy consistently leads in exploration rate and path efficiency. As for the time taken to reach different coverage levels, in the urban setting, the proposed strategy is fastest to 75% and 90% coverage. However, in the forest, it takes longer to achieve 50% coverage but remains competitive at higher coverage levels. Figure 12 illustrates progressive coverage over time, showing the proposed strategy's strength in urban areas but highlighting areas for improvement in Forest environments.



(**a**) Test City. (**b**) Forest.

Figure 12. Example of combined coverage of the environment over time for each scenario using 2 UAVs.

Mapping effectiveness is depicted in Figure 13, showcasing UAV trajectories and generated maps, indicating effective terrain handling but uneven workload distribution.

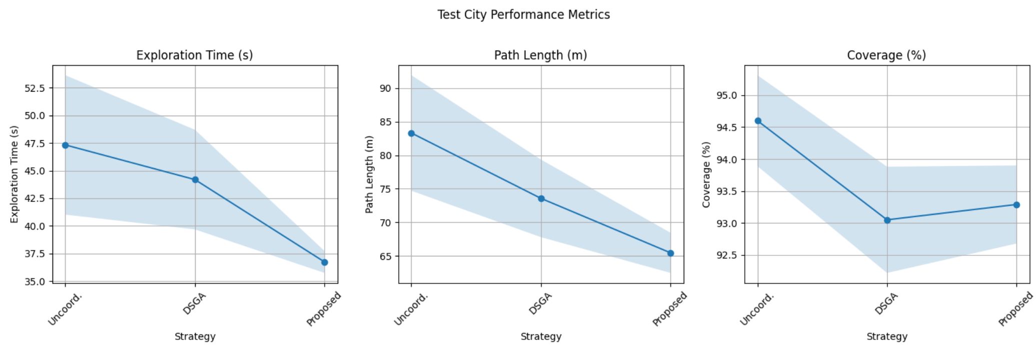


(**a**) Test City. (**b**) Forest.

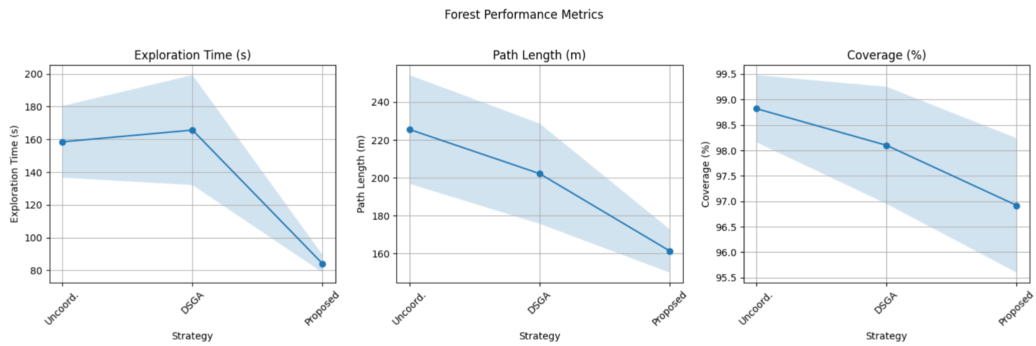
Figure 13. Example of trajectory and map of the environment for each scenario.

Overall, the proposed strategy enhances two-UAV exploration, particularly in urban settings, though improvements are needed for cluttered environments like forests.

Following this, a third UAV was added to the fleet to understand the dynamics and benefits of multi-UAV cooperation in exploration tasks. Due to hardware limitations, LiDAR data were downsampled for this scenario, though real-world applications would not require them due to dedicated onboard hardware for each UAV. Figure 14 compares the same cooperation strategies using key metrics: exploration time, path length, and coverage percentage.



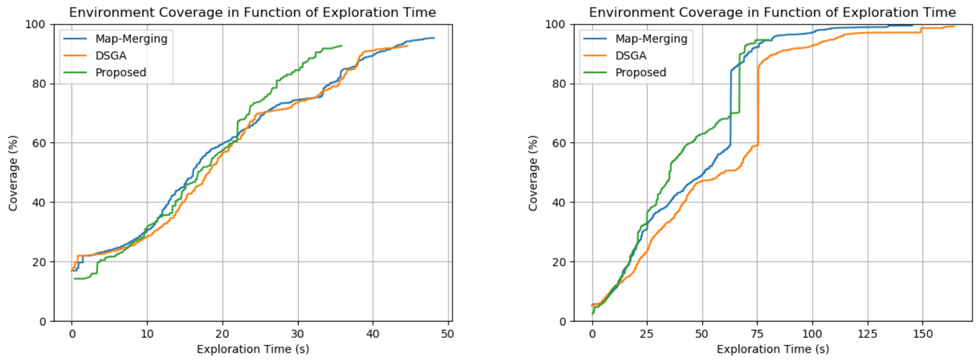
(**a**) Test City.



(**b**) Forest.

Figure 14. Three-UAV exploration benchmarking analysis. The blue line represents the mean value, while the light blue shading indicates the standard deviation of 5 simulation runs.

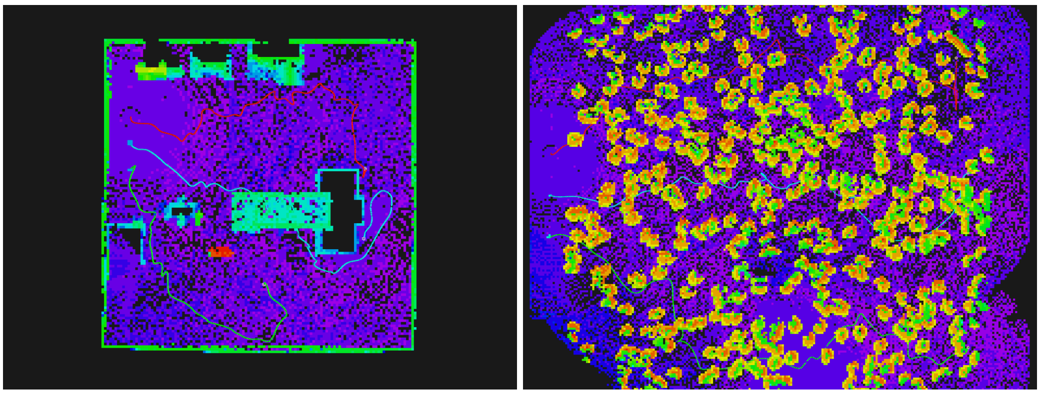
The proposed strategy outperforms others in both environments, achieving the shortest exploration time and path length while maintaining competitive coverage percentages. Moreover, the proposed strategy also excels in exploration rate and path efficiency, surpassing the other two in both environments. Regarding coverage thresholds, it achieves higher coverage levels faster than the other strategies, particularly in the Forest environment. Figure 15 illustrates the progressive coverage over time for each method, showing the proposed strategy's superiority in both environments.



(**a**) Test City. (**b**) Forest.

Figure 15. Example of combined coverage of the environment over time for each scenario by the three-UAV fleet.

Additionally, Figure 16 shows the UAV trajectories and generated maps for each scenario, highlighting the proposed method's effective navigation and comprehensive mapping despite LiDAR downsampling.



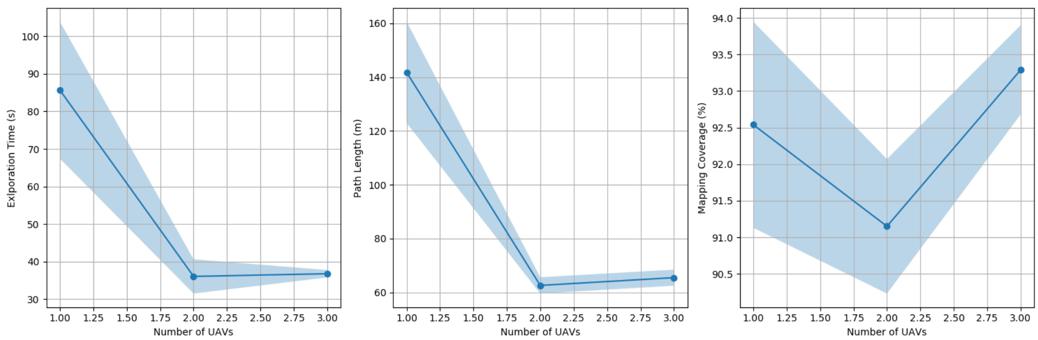
(**a**) Test City. (**b**) Forest.

Figure 16. Example of UAV trajectory overlaid on the map of the environment for each scenario.

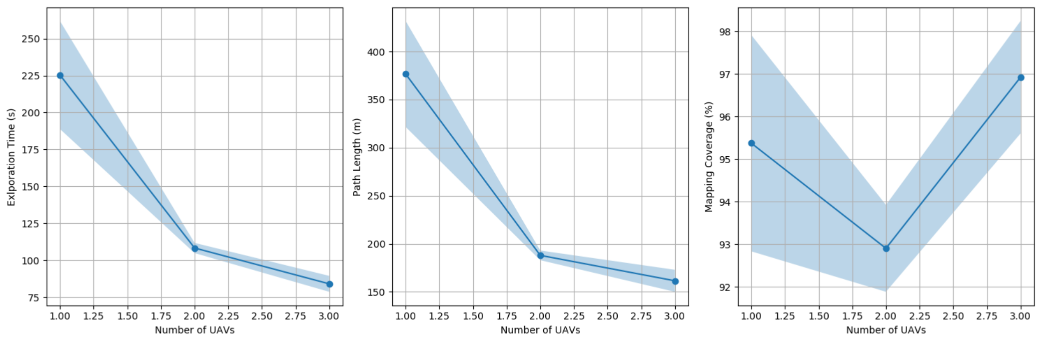
The three-UAV exploration analysis confirms the proposed strategy's superior performance and efficiency, making it the optimal choice for multi-UAV exploration missions in both urban and natural environments.

3.4. Comparative Analysis

This section compares the performance of single-, two-, and three-UAV configurations to understand how team size affects exploration efficiency. Key metrics such as exploration time, path length, coverage percentage, exploration rate, and path efficiency are analyzed in both urban and complex natural environments. Figure 17 shows the comparison of these metrics for each simulated environment.



(**a**) Test City.



(**b**) Forest.

Figure 17. Comparison between the number of UAVs in key metrics for each environment. The blue line represents the mean value, while the light blue shading indicates the standard deviation of 5 simulation runs.

In the Test City, a single UAV averages 80.34 s for exploration, two UAVs reduce this to 36.01 s, and three UAVs have a similar time of 36.73 s. The greatest improvement occurs when increasing from one to two UAVs, while adding a third UAV does not reduce exploration time. In the Forest, exploration time drops from 205.67 s with one UAV to 108.33 s with two, and further to 84.08 s with three, highlighting efficiency gains with additional UAVs in larger, more complex terrains. Average path length per UAV also decreases with from one to two UAVs but worsens when adding a third agent. In the Test City, a single UAV covers 125.45 m on average, while two UAVs reduce this to 62.56 m, and three UAVs to 65.44 m. In the Forest, the path length decreases from 310.92 m with one UAV to 187.94 m with two, and further to 161.47 m with three. Coverage percentages remain high across all configurations. While slight variations exist, overall coverage remains competitive regardless of the number of UAVs, indicating that additional UAVs can improve speed and efficiency without compromising coverage.

The analysis shows that the optimal number of UAVs for the simulated starting conditions depends on the environment's size and complexity. In the Test City, two UAVs provide the best balance of performance and efficiency, significantly reducing exploration time and path length while maintaining high coverage. In the Forest, three UAVs are optimal due to the environment's complexity, offering substantial gains in exploration rate and coverage time. Therefore, for smaller environments like the Test City, two UAVs are ideal, while for complex environments like the Forest, three UAVs maximize efficiency and resource utilization.

4. Conclusions

In this work, a cooperative strategy for MRS in 3D exploration and mapping tasks was introduced. This approach, based on distributed and sequential waypoint assignment, demonstrated significant improvements in exploration time and energy conservation. Our exploration planner played a crucial role in these results. It effectively managed the assignment of waypoints, ensuring optimal coverage of the environment and efficient use of resources. The planner's ability to dynamically adjust to changes in the environment and robot team composition was key to its success.

In the simulated environment, the effectiveness of our exploration planner was validated with a single-UAV setup. The introduction of a second UAV optimized the use of battery resources by reducing the exploration time and individual travel distance. However, it also led to a slight decrease in mapping coverage and an increase in standard deviation. These changes could be attributed to the challenges of coordinating multiple UAVs. As the number of UAVs in a system increases, so does the complexity of coordination and the challenges of collision avoidance, which may affect performance. The addition of a third UAV not only improved efficiency but also improved mapping completeness. The increased frequency of information sharing among the UAVs might have contributed to this improved performance, reducing discrepancies.

These findings highlight the importance of communication frequency in multi-UAV systems and indicate that more research is needed to fully understand these dynamics. The careful consideration of these factors is crucial when increasing the number of UAVs in a system due to potential trade-offs.

In conclusion, our work provides a solid foundation for future investigations in cooperative MRS. The exploration planner presented in this study has potential applications in a broad spectrum of MRS, underscoring the relevance of our work in this field. Future research will involve validating this framework in real-world scenarios to ensure its robustness and adaptability beyond simulations. We also plan to expand the scope of experiments to include more diverse environments and larger teams of UAVs. Additionally, improving communication protocols among UAVs, particularly to address network bandwidth and latency issues as team sizes grow, presents a critical avenue for enhancing system robustness. Another area worth exploring is the development of more efficient real-time path planning algorithms, specifically aimed at larger UAV teams, where the need for collision avoidance, task allocation, and synchronization grows significantly more complex. These advancements will be crucial for unlocking the full potential of MRS in dynamic and large-scale environments.

Author Contributions: Conceptualization, A.R. and M.B.; methodology, A.R. and M.B.; software, A.R.; validation, A.R. and M.B.; formal analysis, A.R. and M.B.; investigation, A.R. and M.B.; resources, A.R. and M.B.; data curation, A.R. and M.B.; writing—original draft preparation, A.R.; writing—review and editing, A.R. and M.B.; visualization, A.R. and M.B.; supervision, M.B.; project administration, M.B.; funding acquisition, M.B. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by LARSyS funding (DOI: 10.54499/LA/P/0083/2020, 10.54499/UIDP/50009/2020 and 10.54499/UIDB/50009/2020) and Aero.Next Portugal project (PRR-C645727867-00000066).

Data Availability Statement: The original contributions from this study are available in the article and on the GitHub repository https://github.com/andremribeiro/thesis (accessed on 1 October 2024). For further inquiries, please contact the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Manfreda, S.; McCabe, M.F.; Miller, P.E.; Lucas, R.; Pajuelo Madrigal, V.; Mallinis, G.; Ben Dor, E.; Helman, D.; Estes, L.; Ciraolo, G.; et al. On the Use of Unmanned Aerial Systems for Environmental Monitoring. *Remote Sens.* **2018**, *10*, 641. [CrossRef]
2. Bartolomei, L.; Teixeira, L.; Chli, M. Fast Multi-UAV Decentralized Exploration of Forests. *IEEE Robot. Autom. Lett.* **2023**, *8*, 5576–5583. [CrossRef]

3. Khan, A.; Gupta, S.; Gupta, S.K. Emerging UAV technology for disaster detection, mitigation, response, and preparedness. *J. Field Robot.* **2022**, *39*, 905–955. [CrossRef]
4. Wang, H.; Zhang, C.; Song, Y.; Pang, B.; Zhang, G. Three-Dimensional Reconstruction Based on Visual SLAM of Mobile Robot in Search and Rescue Disaster Scenarios. *Robotica* **2020**, *38*, 350–373. [CrossRef]
5. Krátký, V.; Petráček, P.; Báča, T.; Saska, M. An autonomous unmanned aerial vehicle system for fast exploration of large complex indoor environments. *J. Field Robot.* **2021**, *38*, 1036–1058. [CrossRef]
6. Lin, J.J.; Ibrahim, A.; Sarwade, S.; Golparvar-Fard, M. Bridge Inspection with Aerial Robots: Automating the Entire Pipeline of Visual Data Capture, 3D Mapping, Defect Detection, Analysis, and Reporting. *J. Comput. Civ. Eng.* **2021**, *35*, 04020064. [CrossRef]
7. Bettencourt, R.; Lewis, J.; Serra, R.; Basiri, M.; Vale, A.; Lima, P.U. GEERS: Georeferenced Enhanced EKF Using Point Cloud Registration and Segmentation. *IEEE Robot. Autom. Lett.* **2024**, *9*, 1803–1810. [CrossRef]
8. Lewis, J.; Lima, P.U.; Basiri, M. Collaborative 3D Scene Reconstruction in Large Outdoor Environments Using a Fleet of Mobile Ground Robots. *Sensors* **2023**, *23*, 375. [CrossRef] [PubMed]
9. Hardouin, G.; Moras, J.; Morbidi, F.; Marzat, J.; Mouaddib, E.M. Next-Best-View planning for surface reconstruction of large-scale 3D environments with multiple UAVs. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October–24 January 2021; pp. 1567–1574. [CrossRef]
10. Selin, M.; Tiger, M.; Duberg, D.; Heintz, F.; Jensfelt, P. Efficient Autonomous Exploration Planning of Large-Scale 3-D Environments. *IEEE Robot. Autom. Lett.* **2019**, *4*, 1699–1706. [CrossRef]
11. Zhang, H.; Wang, S.; Liu, Y.; Ji, P.; Yu, R.; Chao, T. EFP: Efficient Frontier-Based Autonomous UAV Exploration Strategy for Unknown Environments. *IEEE Robot. Autom. Lett.* **2024**, *9*, 2941–2948. [CrossRef]
12. Yamauchi, B. A frontier-based approach for autonomous exploration. In Proceedings of the Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97: 'Towards New Computational Principles for Robotics and Automation', Monterey, CA, USA, 10–11 July 1997; pp. 146–151. [CrossRef]
13. Cieslewski, T.; Kaufmann, E.; Scaramuzza, D. Rapid exploration with multi-rotors: A frontier selection method for high speed flight. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 2135–2142. [CrossRef]
14. Zhou, B.; Zhang, Y.; Chen, X.; Shen, S. FUEL: Fast UAV Exploration Using Incremental Frontier Structure and Hierarchical Planning. *IEEE Robot. Autom. Lett.* **2021**, *6*, 779–786. [CrossRef]
15. Connolly, C. The determination of next best views. In Proceedings of the 1985 IEEE International Conference on Robotics and Automation Proceedings, St. Louis, MO, USA, 25–28 March 1985; Volume 2, pp. 432–435. [CrossRef]
16. Bircher, A.; Kamel, M.; Alexis, K.; Oleynikova, H.; Siegwart, R. Receding Horizon "Next-Best-View" Planner for 3D Exploration. In Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016; pp. 1462–1468. [CrossRef]
17. Batinović, A.; Petrović, T.; Ivanovic, A.; Petric, F.; Bogdan, S. A Multi-Resolution Frontier-Based Planner for Autonomous 3D Exploration. *IEEE Robot. Autom. Lett.* **2021**, *6*, 4528–4535. [CrossRef]
18. Burgard, W.; Moors, M.; Stachniss, C.; Schneider, F. Coordinated multi-robot exploration. *IEEE Trans. Robot.* **2005**, *21*, 376–386. [CrossRef]
19. Yamauchi, B. Decentralized coordination for multirobot exploration. *Robot. Auton. Syst.* **1999**, *29*, 111–118. [CrossRef]
20. Wurm, K.; Stachniss, C.; Burgard, W. Coordinated multi-robot exploration using a segmentation of the environment. In Proceedings of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, Nice, France, 22–26 September 2008; pp. 1160–1165. [CrossRef]
21. Zlot, R.; Stentz, A.; Dias, M.; Thayer, S. Multi-robot exploration controlled by a market economy. In Proceedings of the Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292), Washington, DC, USA, 11–15 May 2002; Volume 3, pp. 3016–3023. [CrossRef]
22. Li, H.; Zhang, Q.; Zhao, D. Deep Reinforcement Learning-Based Automatic Exploration for Navigation in Unknown Environment. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *31*, 2064–2076. [CrossRef] [PubMed]
23. Issa, R.B.; Saferi Rahman, M.; Das, M.; Barua, M.; Rabiul Alam, M.G. Reinforcement Learning based Autonomous Vehicle for Exploration and Exploitation of Undiscovered Track. In Proceedings of the 2020 International Conference on Information Networking (ICOIN), Barcelona, Spain, 7–10 January 2020; pp. 276–281. [CrossRef]
24. Zhu, D.; Li, T.; Ho, D.; Wang, C.; Meng, M.Q.H. Deep Reinforcement Learning Supervised Autonomous Exploration in Office Environments. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 21–25 May 2018; pp. 7548–7555. [CrossRef]
25. Garaffa, L.C.; Basso, M.; Konzen, A.A.; de Freitas, E.P. Reinforcement Learning for Mobile Robotics Exploration: A Survey. *IEEE Trans. Neural Netw. Learn. Syst.* **2023**, *34*, 3796–3810. [CrossRef] [PubMed]
26. Ribeiro, A.; Basiri, M. Cooperative 3D Exploration and Mapping using Distributed Multi-Robot Teams. In Proceedings of the 2024 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC), Paredes de Coura, Portugal, 2–3 May 2024; pp. 132–137; ISSN: 2573-9387. [CrossRef]
27. Hornung, A.; Wurm, K.M.; Bennewitz, M.; Stachniss, C.; Burgard, W. OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Auton. Robot.* **2013**, *34*, 189–206. [CrossRef]

28. Fukunaga, K.; Hostetler, L. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Trans. Inf. Theory* **1975**, *21*, 32–40. [CrossRef]
29. González-Baños, H.H.; Latombe, J.C. Navigation Strategies for Exploring Indoor Environments. *Int. J. Robot. Res.* **2002**, *21*, 829–848. [CrossRef]
30. Baca, T.; Petrlik, M.; Vrba, M.; Spurny, V.; Penicka, R.; Hert, D.; Saska, M. The MRS UAV System: Pushing the Frontiers of Reproducible Research, Real-world Deployment, and Education with Autonomous Unmanned Aerial Vehicles. *J. Intell. Robot. Syst.* **2021**, *102*, 26. [CrossRef]