Efficient Frontier Selection via Reinforcement Learning for Exploring Unstructured Environments with Minimal Sensing

Javier Melero Deza *, Pedro Arias Perez, Guillermo GP-Lenza, Martín Molina, Pascual Campoy

Posted Date: 28 April 2026

doi: 10.20944/preprints202604.1959.v1

Keywords: reinforcement learning; zero-shot transfer; cross-attention; aerial robots; exploration

 doi:10.20944/preprints202604.1959.v1

Article

Efficient Frontier Selection via Reinforcement Learning for Exploring Unstructured Environments with Minimal Sensing

Javier Melero-Deza [1,2], Pedro Arias-Perez [3], Guillermo GP-Lenza [1,2], Martín Molina [2] and Pascual Campoy [1]

[1] Centre for Automation and Robotics C.A.R. (UPM-CSIC), Calle Jose Gutierrez Abascal 2, 28006 Madrid, Spain
[2] Department of Artificial Intelligence, Universidad Politécnica de Madrid (UPM), 28660 Madrid, Spain
[3] Department of Systems Engineering and Automation, Universidade de Vigo, 36310 Vigo, Spain
* Correspondence: javier.mdeza@upm.es

Highlights

What are the main findings?

- A novel cross-attention policy architecture fusing global map features with variable-length frontier features outperforms classical heuristics, yielding better efficiency in low, medium, and high density scenarios.
- The ablation study confirms the cross-attention module as the key contributor, yielding a stronger final policy with smoother convergence and lower variance than a concatenation-based baseline.

What are the implications of the main findings?

- Zero-shot sim-to-real transfer on a Bitcraze Crazyflie with minimal sensing demonstrates practical applicability for cost-effective search and rescue in unstructured environments.
- The architecture supports variable-sized occupancy grids without retraining, enabling scalable deployment across scenarios of different spatial scales in aerial robotics.

Abstract

In recent years, reinforcement learning (RL) has been applied to frontier-based exploration to enhance the robot's decision-making policy and improve exploration performance. In this work, we address this scenario with the aim of pushing forward the finding of the optimal frontier selection policy in unknown, unstructured environments with RL deployed for a minimal sensing drone setup. We propose a novel policy architecture, featuring an attention module that uses the global map features captured by a convolutional neural network together with local frontier features in the form of scalar values, trained end-to-end with a scoring network using the Proximal Policy Optimization algorithm over a 2D randomized unstructured environment. Our approach demonstrates a notable improvement in exploration efficiency as it surpasses purely heuristic-based frontier selection strategies used as baselines for other RL methods, achieving shorter paths than Nearest Frontier, Hybrid Approach, and TARE local horizon, as well as one-shot sim-to-real policy deployment.

Keywords: reinforcement learning; zero-shot transfer; cross-attention; aerial robots; exploration

1. Introduction

Frontier-based exploration has become a foundational strategy for autonomous mapping of unknown environments since Yamauchi's seminal work introduced the concept in 1997 [1]. In a 2D occupancy grid map, frontiers are defined as the boundaries between explored free space and unexplored regions. By moving to frontier cells in a 2D occupancy grid, a robot can systematically cover the environment, adding new sensor information and pushing back the "known/unknown" boundary until no frontier remains. This approach has been widely adopted in robotic exploration due

 doi:10.20944/preprints202604.1959.v1

to its simplicity and its capability to yield complete environment coverage [2–4]. However, classical frontier-based methods typically rely on hand-crafted heuristics (e.g., choosing the nearest or largest frontier), which may not yield optimal long-term exploration efficiency in more complex environments. This limitation has motivated research into more effective decision-making strategies for exploration.

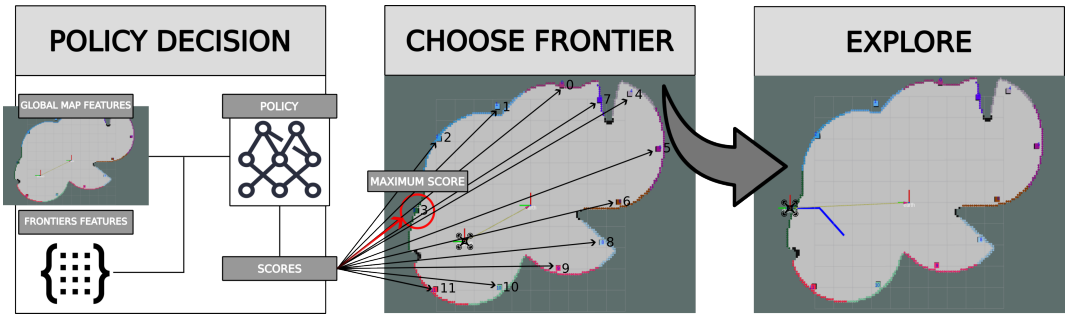


Figure 1. Frontier selection for efficient exploration of unstructured environments. Our policy employs an attention module that uses the global map features captured by a convolutional neural network and the local frontier features in the form of scalar values. Then, a scoring network chooses the next frontier to explore.

In recent years, reinforcement learning (RL) has been applied to frontier-based exploration to enhance the robot's decision-making policy and improve exploration performance [5]. By formulating exploration as a sequential decision problem, RL enables an agent to learn deciding which frontiers to select or which navigation actions to take in order to maximize mapping efficiency or information gain over time. The growing interest in RL-driven exploration is underscored by recent surveys [6,7], which conclude that learning-based strategies can adapt to complex 2D environments and effectively balance exploration-exploitation trade-offs. By building on the frontier paradigm with deep learned policies, these approaches enable autonomous robots to explore more intelligently, improving map coverage and efficiency in challenging scenarios.

In this work, we address this scenario with the aim of pushing forward the finding of the optimal exploration policy with RL by improving purely heuristic-based exploration strategies and existing RL methods. Even though the whole approach and pipeline are platform agnostic, our motivation is also driven by our goal of integrating these techniques into the aerial robotics domain, more specifically, minimal sensing equipped drones, to provide a better and more cost-effective solution for search and rescue in unstructured environment applications like forests, jungles, or even disaster zones. Thus, the research question for this work is: Can we improve state-of-the-art heuristic frontier-based methods with the latest RL approaches?

To answer this question, we integrate Arias-Perez et al. [8] exploration pipeline, which, in turn, is based on Aerostack2's [9] aerial framework, replacing its frontier explorer with a novel policy architecture approach. The main contributions of our RL-driven exploration approach are summarized as follows:

1. The proposed policy consists of a true variable frontier selector policy which also accepts a variable observation state processed by a scorer network [10], together with a Cross-Attention encoder module [11].
2. We provide a comparison with the different heuristic methods that the most recent RL frontier-based exploration approaches use as baselines, such as Nearest Frontier [1], Hybrid Approach (nearest and information gain [12]), and TARE Local [14].
3. Finally, to demonstrate the system works, we zero-shot transfer the policy to a real scenario, achieving consistent behavior, integrated in a real ROS2 [13] aerial framework.

2. Related Work

In autonomous exploration, frontier-based methods have long served as a fundamental strategy for mapping unknown environments, a concept first introduced by Yamauchi [1] where frontiers

 doi:10.20944/preprints202604.1959.v1

are defined as the boundaries between known free space and unexplored regions. Building on this pioneering work, Burgard et al. [15] extended the concept to multi-robot systems by incorporating utility and cost functions that coordinate exploration efficiently, ensuring that robots spread out over distinct frontiers to maximize coverage. González Baños would the introduce a ray casting non-discrete version of this information gain approach later in [16]. In an effort to overcome the limitations of exhaustive scanning in classical methods, Umari and Mukhopadhyay [17] proposed a hybrid approach that combines rapidly-exploring random trees (RRTs) with frontier detection, allowing for rapid generation and selection of candidate frontiers based on a utility heuristic balancing information gain and travel cost. Further refinement was achieved by Deng et al. [18] who introduced a differentiable formulation of the frontier information gain metric, enabling the use of gradient-based optimization to continuously improve exploration trajectories. Complementarily, Sun et al. [19] addressed the challenge of unreachable frontiers by integrating reachability analysis into the exploration process, thereby filtering out non-viable candidates and ensuring more robust navigation in 2D environments. More recently, Zhang et al. [20] advanced the state-of-the-art with a two-level approach that not only accelerates frontier detection through sampling but also incorporates an advanced viewpoint heuristic to optimize target selection, significantly enhancing mapping efficiency. Together, these studies represent a progressive evolution from simple, greedy frontier selection to sophisticated techniques used in real-world 2D mapping tasks.

Existing reinforcement learning (RL) approaches for frontier selection have significantly advanced autonomous exploration beyond classical heuristics [6]. Fan et al. [21] use an edge detection algorithm to find potential frontier candidates from the occupancy grid map. Then, 16 candidate points are randomly chosen from the detected edges. Once selected, these points get sorted clockwise. This ordered list, in the form of information gain and spatial features, is then mapped directly to the fixed-size output layer of the actor network. In B. Yu et al. [22]'s framework, frontiers are encoded as fixed channels in the frontier map—specifically, the action space is defined as 4 channels corresponding to potential frontier cells. The policy outputs a vector over these channels, using invalid action masking in case the number of frontiers generated is less than 4. These approaches assume a fixed, discrete action space of frontier choices, which means the agent can only pick from a predetermined number of frontier candidates at each decision step, an oversimplification due to the loss of granularity given that an ideal frontier generation module takes into consideration the length of the boundary between known and unknown space to generate a variable number of frontiers as the robot maps its environment. G. Iyer et al. [23] formulate frontier selection with continuous control, having the agent predict a continuous-valued goal location for the next frontier. A hybrid heuristic-RL approach is presented by Niroui's work [24], in which instead of outputting a fixed vector of probabilities, the actor produces a learned weight that feeds into a cost function combining normalized distance and information gain for each candidate, and the frontier with the lowest cost is chosen. In Cao et al. work [25] candidate frontiers are represented as nodes in an augmented collision-free graph built over the free area. The attention-based policy network with a pointer mechanism in its decoder dynamically computes weights over the variable set of neighboring nodes. While these approaches demonstrate the feasibility of RL-driven exploration, they do not fully capture the dynamic nature of the frontier set; the policy either operates on a constant-sized action set or must rely on auxiliary mechanisms to cope with varying frontier availability.

Introducing recent RL works that deal with the ability of dealing with the variable nature of frontier generation, Liu et al. [26] approach consists of applying rule-based imitation learning to then introduce a reinforcement learning training with the Soft-Actor-Critic (SAC) method, using a scoring network to get the output. Wang et al. [27] propose a CNN to capture both local frontier features and global features, where both inputs are fixed; if the global map is larger than the CNN input, they resize it to a lower resolution. After that, they perform sequential scoring (i.e., an iterative sequential process that scores each output individually using the scoring network), similar to a scoring network but without the ability of parallelizing due to sequential candidate scoring.

 doi:10.20944/preprints202604.1959.v1

These works leave room for improvement. While they capture the true variability of frontiers through the scoring network output, they rely on additional mechanisms that greatly influence the resulting behavior. Techniques such as imitation learning introduce a significant initial bias depending on the imitated policy, whereas graph structures, combined with the constraint of selecting only local neighbors as navigation nodes, lead to a loss of granularity. Additionally, the use of padded visual feature maps to capture local frontier features to match the same shape results in a loss of resolution as the map size increases. None of these approaches introduce techniques beyond a CNN architecture or a scoring network.

3. Exploration Robotics Framework

Before going further into the main contributions of this article, we aim to provide details on how the reinforcement learning environment, together with the policy is integrated in a real aerial robotics framework. As mentioned before, we integrate our RL environment into Arias-Perez et al. [8] minimal sensing ROS 2-based exploration framework, which is at the same time based on Aerostack2 [9]. The frontier allocator heuristic from the original architecture is replaced with an RL environment, which contains observation and action handler modules as ROS 2 nodes, along with the proposed novel policy.

Depending on wether the policy is being trained or tested in a real environment, the data source differs. When training we use Gazebo Simulator as Figure 2(**a**) depicts. The agent simulated in the Gazebo simulator utilizes readings from a 360° planar LiDAR sensor to create a scan message, which is collected by the mapping subsystem and encoded as a grid map. There is no control loop enabled, control is replaced for a teleport to speed up training.

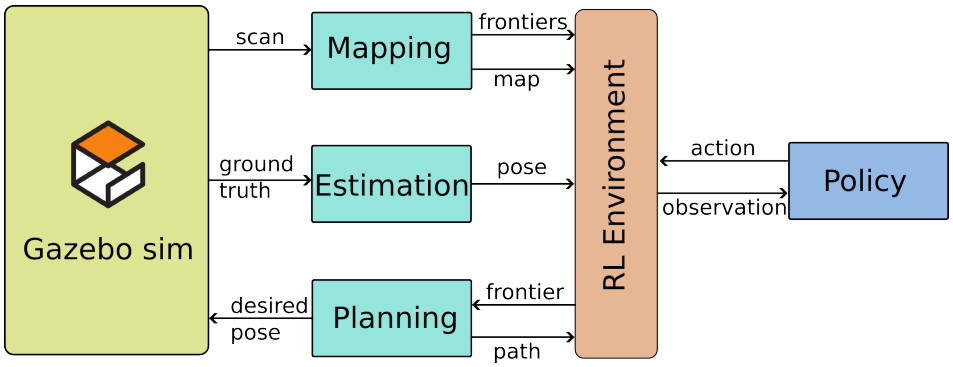
When testing in a real environment, simulated agent is substituted for a real nano-drone, as Figure 2(**b**) indicates. Instead of having a 360° planar LiDAR, the drone is equiped with a 4 beam rangefinder sensor. When it reaches a certain position, a 90° spin in place with this rangefinder sensor produces the same result as the simulated 360° LiDAR sensor. Control is back enabled, allowing the drone to reach the desired position with a PID based controller. Ground truth estimation information is replaced for a Motion Capture System.

This modular design ensures that the observation space remains invariant to the actual sensor modality: regardless of whether the scan originates from a simulated 360° LiDAR or a physical 4-beam rangefinder performing a 90° rotation, the resulting occupancy grid fed to the policy is structurally identical, which is the key enabler for the zero-shot sim-to-real transfer demonstrated in Section 5.7.

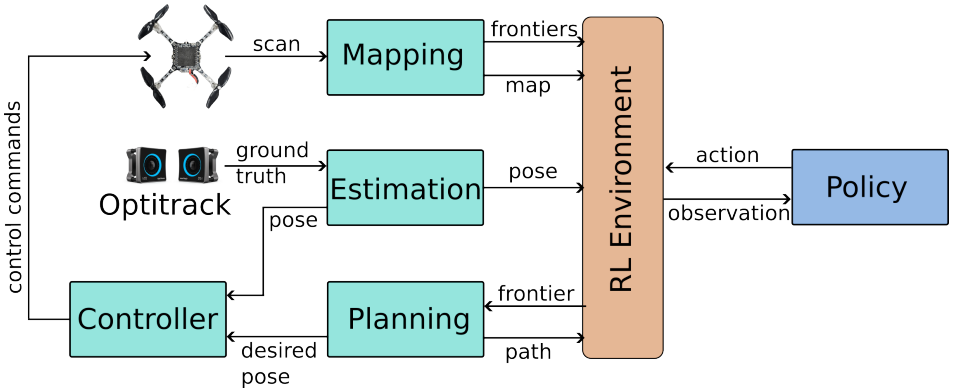
From the map structure, the set of available frontiers is calculated. The observation handler uses the map, the list of frontiers, and the agent's position to create the observation state, which is consumed by the policy to calculate the selected frontier (action). Once the policy has decided the next point to explore, the planning subsystem executes the planner algorithm to obtain a path to reach the frontier, which is used to compute the reward function. Finally, for training in simulation we set the new pose automatically in Gazebo to teleport the agent to the desired position to improve training speed. For the real experiments, the agent would use a PID controller to move into the desired frontier position.

Given that the sensor collects only 2D information about the environment, the map is encoded as an occupancy grid. Within the planning components, the frontier allocator node returns a set of spatial 2D points within the grid. Both the map and the set of frontier points serve for the RL environment observation purposes.

 doi:10.20944/preprints202604.1959.v1



(**a**) Simulation setup. Gazebo provides the source of data. There is no controller, Gazebo sets the pose whenever there is a requested desired pose.



(**b**) Real setup. The drone provides the lidar data. MoCap system provides ground truth estimation data. Whenever a desired position is given back by the RL environment, the controller sends control commands to the drone.

Figure 2. Exploration framework overview. Boxes represent each framework component, compose of one or more ROS 2 nodes. Solid arrows stand for communication interfaces such as ROS 2 topics or services. The agent generate a scan message, which is then passed to the Mapping component. Within these Mapping components, a scan-to-grid map node generates the grid map which is then passed via topic to a map server node. This grid map structure is used by the observation handler of the RL environment, together with the last mapping component which is the frontier generator node. The state estimator node receives the ground truth to provide the agent position to the environment. This position, together with the map and frontiers are used to build the observation state. Within the Planning components, the path planner node generates a path once the policy has chosen the next frontier point to explore. This path is used to calculate the reward once it is returned to the environment. The desired pose is passed to the controller or gazebo, and the whole process is repeated.

4. Reinforcement Learning Based Exploration Approach

Our approach introduces an attention module that uses the global map features captured by a CNN, along with local frontier features in the form of scalar values, trained end-to-end with a scoring network using Proximal Policy Optimization (PPO) [28] from Stable Baselines 3 [29]. This section includes the explanation of how the observation and action spaces are defined, together with the policy architecture details, which include architecture design and decisions, policy parameters and working mechanisms.

4.1. Observation and Action Spaces

We divide our observation space into two components. First, a grid map encodes the global map in a 5-channel binary tensor. Figure 3 depicts how these five channels are encoded, gathering free space, frontier centroids, and the agent's position information. This structured representation enables convolutional models to effectively extract relevant features for navigation and exploration. To further

 doi:10.20944/preprints202604.1959.v1

guide the agent toward informative regions, frontier cells and the agent's position are smoothed using Gaussian kernels, producing soft gradients that facilitate exploration behavior.

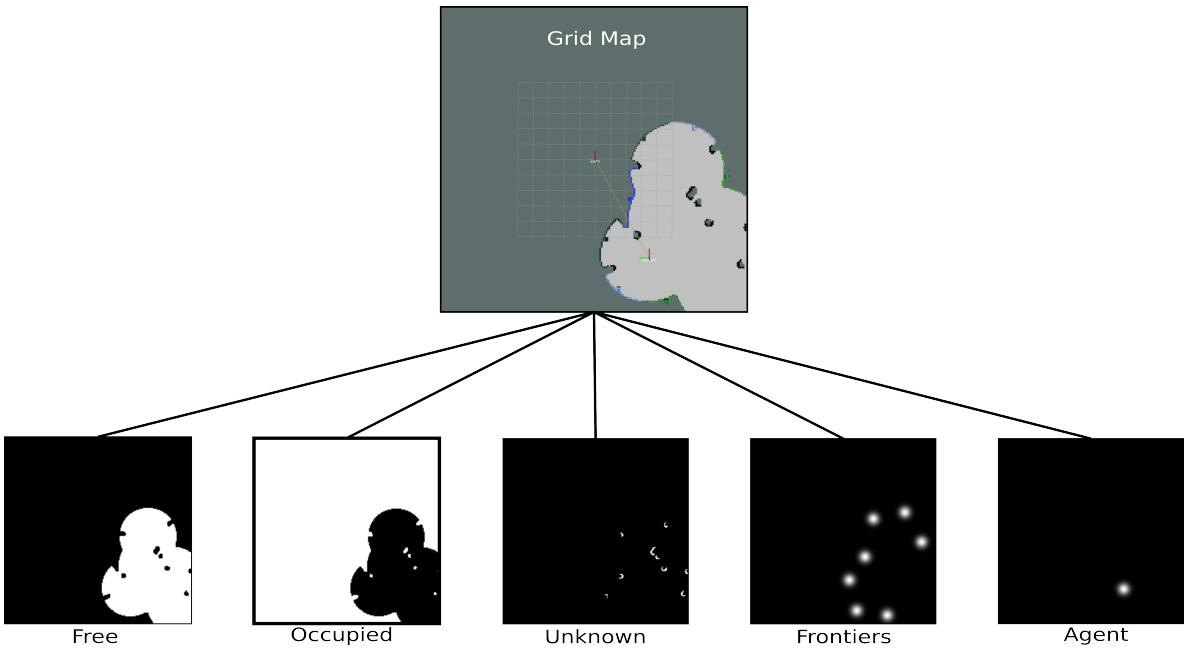


Figure 3. Grid Map with an integrated map representation with discrete semantic values: free, occupied, unknown, frontier, agent position, transformed into a 5-channel binary encoded tensor. Each channel corresponds to one semantic class, where cells belonging to a given class are marked as 1 in the respective channel and 0 elsewhere.

Second, a variable-length list of frontiers that includes a fixed-size vector with each frontier's features. As the number of frontiers keeps changing as the agent explores, the size of the list of frontier feature vectors also changes. Each frontier is represented by a feature vector of five normalized scalar values. Given an agent position (x_a, y_a), frontier position (x_f, y_f), and grid size S, the frontier features exploited are described in vector 1.

$$\left[\frac{x_f}{S}, \frac{y_f}{S}, \frac{\sqrt{(x_f - x_a)^2 + (y_f - y_a)^2}}{\sqrt{2}\,S}, \frac{x_a}{S}, \frac{y_a}{S}\right], \tag{1}$$

where the Euclidean distance between the agent and a frontier is normalized by $\sqrt{2}S$, the maximum distance in a square grid.

As per action space, the policy outputs an index pointing to the chosen frontier within the input vector. The shortest path to reach the selected frontier is then fed to the reward function, which will be discussed later.

4.2. Policy Architecture

In this work, we utilize an actor-critic architecture with several scenario-specific improvements. Figure 4 depicts a simplification of the overall architecture. To describe the architecture, we decompose it into three functional blocks: the context-augmented observation module (CAOM), the actor head, and the critic head. Note that while the CAOM is structurally common to both the actor and the critic, there are also some shared-weight configuration-specific components within the CAOM that use identical parameters across both pathways, rather than merely mirroring the same topology with independent weights.

We combine the features extracted from the global map extracted from a CNN together with hand-crafted scalar vectors of the features of each frontier point. We propose using an attention module that leverages global map embeddings to query the candidate frontier points by projecting them into key and value embeddings, generating an aggregated representation that reflects the information

 doi:10.20944/preprints202604.1959.v1

of the candidates most relevant to the overall environment. This aggregated representation is then processed by the actor and the critic to produce its respective outputs.

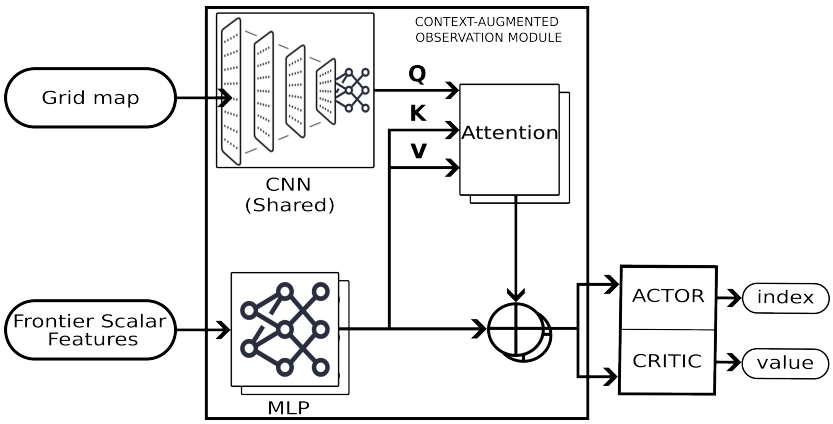


Figure 4. Overall policy architecture. The three functional blocks will be explained further away in the article, these are: context-augmented observation module, common structure for both the actor and the critic, and the actor and the critic themselves.

By subsequently adding (via a residual connection) the original candidate embeddings to this attention output, we enrich each frontier feature with contextual cues without discarding the original local information. This fusion combines the strengths of both representations: it preserves the inherent details of each candidate while integrating a global perspective that can highlight important inter-candidate relationships. The residual connection ensures that the information derived from the local candidate features is maintained alongside the aggregated context-aware adjustments.

4.3. Context-Augmented Observation Module

Within our crafted architecture, the map feature extractor (CNN) and the map encoder are shared between the actor and the critic. The intuition is that the CNN and map encoder extract a general spatial understanding of the environment, which is useful for both the actor and the critic equally. The candidate-specific processing via the attention mechanism is where the actor and critic diverge: the actor needs to score candidates relative to each other (which one to pick), while the critic needs to aggregate them into a single state value. These are different enough tasks to justify separate weights.

Sharing the CNN and map encoder between both pathways also provides a regularization effect: during PPO updates, gradients from both the actor and critic losses jointly update the shared spatial feature extractor, acting as a multi-task learning signal that encourages richer map representations and prevents overfitting to either objective alone. This design keeps the shared CAOM at roughly 4.08M parameters (see Table 1), avoiding the near-doubling that fully independent feature extractors would require.

The embedding dimension is set to $n = 64$ throughout the architecture. For the frontier embedding network, we use a single linear layer ($5 \rightarrow 64$) with ReLU activation, since the frontier features in Equation 1 are already hand-crafted and normalized, the representational heavy lifting is delegated to the cross-attention module.

 doi:10.20944/preprints202604.1959.v1

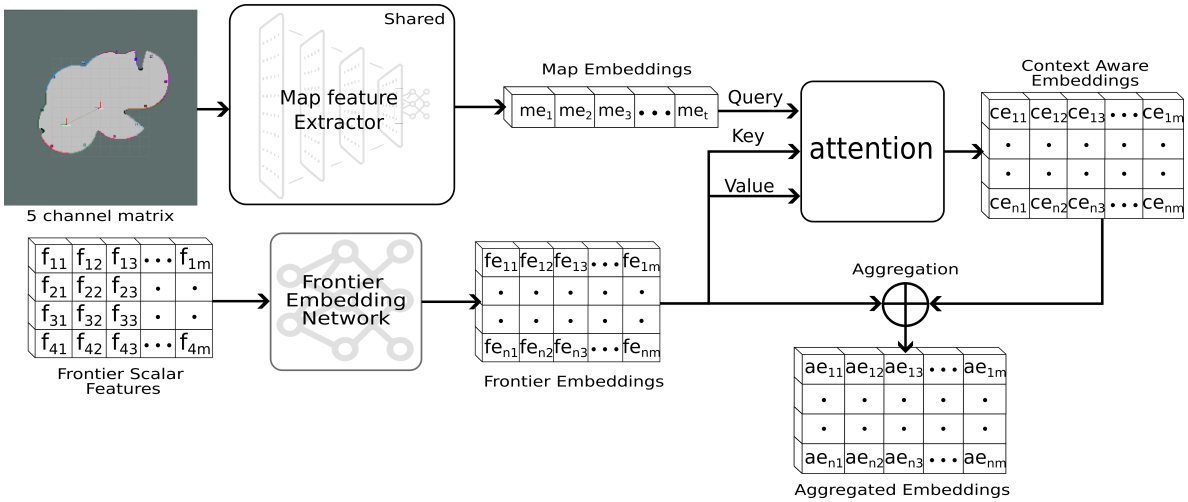


Figure 5. Context-Augmented Observation Module overview. In the same timestep, a preprocessed 5 channel matrix, which represent the global map, and a vector of scalar features for each of m frontiers act as the input for the policy. The 5 channel matrix gets processed by a map feature extractor (CNN) that returns the features which are then projected into the embedding space by an MLP. CNN feature extractor and map encoder is shared between the critic and the actor. These embeddings serves as the query for the attention module. Each vector of scalar frontier features gets also projected into embedding vectors of size n. This batch of features serves as the key and value input for the attention module. The attention output embedding gets added to every frontier embedding.

Table 1. Detailed network architecture. Layers marked with † are shared between actor and critic. The CNN input is a $5 \times 200 \times 200$ binary tensor. Each frontier is represented by a 5-dimensional scalar feature vector. m denotes the variable number of frontiers.

Component	Layer	Input Shape	Output Shape	Parameters
Map Feature Extractor (CNN)†				
	Conv2d (8×8, stride 4) + ReLU	$5 \times 200 \times 200$	$32 \times 49 \times 49$	10,272
	Conv2d (4×4, stride 2) + ReLU	$32 \times 49 \times 49$	$64 \times 23 \times 23$	32,832
	Conv2d (3×3, stride 2) + ReLU	$64 \times 23 \times 23$	$128 \times 11 \times 11$	73,856
	Flatten	$128 \times 11 \times 11$	15,488	0
	Linear + ReLU	15,488	256	3,965,184
Map Encoder†				
	Linear + ReLU	256	64	16,448
Frontier Embedding Network (shared encoder)				
	Flatten (Identity)	5	5	0
	Linear + ReLU	5	64	384
Actor Cross-Attention				
	MultiheadAttention (4 heads)	Q: 1×64, K/V: $m \times 64$	1×64	16,640
	Residual addition	$m \times 64$	$m \times 64$	0
Scoring Network				
	Linear + ReLU	64	64	4,160
	Linear (score)	64	1	65
	Softmax + Multinomial sampling	m	index j	0
Critic Cross-Attention				
	MultiheadAttention (4 heads)	Q: 1×64, K/V: $m \times 64$	1×64	16,640
	Residual addition	$m \times 64$	$m \times 64$	0
Critic Head				
	Mean pooling	$m \times 64$	64	0
	Linear + ReLU	64	64	4,160

Continued on next page

 doi:10.20944/preprints202604.1959.v1

Component	Layer	Input Shape	Output Shape	Parameters
	Linear (value)	64	1	65
Total trainable parameters				**~4.14M**

4.3.1. Actor

To select the next frontier point, we then use a scoring network. This design, depicted in Figure 6, treats the number of frontiers dimension as a batch input to the network, resulting in order-covariant outputs capable of handling varying frontiers. We then pass the head scores through a stochastic categorical distribution, thus, we turn the scorer output logits into probabilities using a softmax layer and then draw one sample from the multinomial distribution defined by those probabilities. The maximum value index is likely to be chosen and returned by the policy. In the end, the policy decides which frontier point to explore next by scoring each candidate based on both its features and the global map representation.

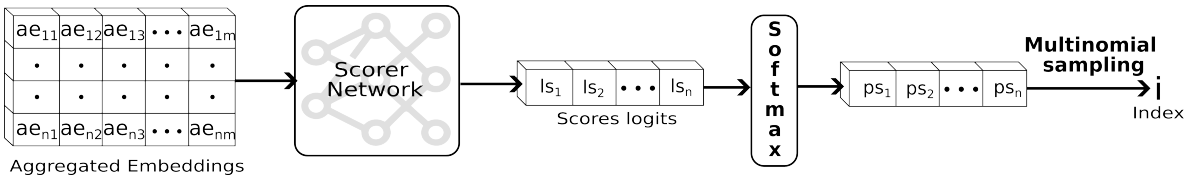


Figure 6. The aggregated embeddings gets processed by a simple 64x64 Relu MLP. The scoring network's output results in a single logit per batch (frontier), which are then turned into probabilities. An index is produced as a result of performing multinomial sampling from this probabilities vector.

4.3.2. Critic

To produce an estimated value, we use average mean pooling to average across all N frontier candidates, collapsing them into a single vector, so the critic is answering how good the overall state is, considering all frontiers together and not how good each individual frontier is. The critic estimates V(s), and PPO computes the advantage as $A = R + \gamma V(s') - V(s)$. That advantage is then used to weight the policy gradient for whichever action the actor sampled. The critic never needs to know which specific frontier was picked.

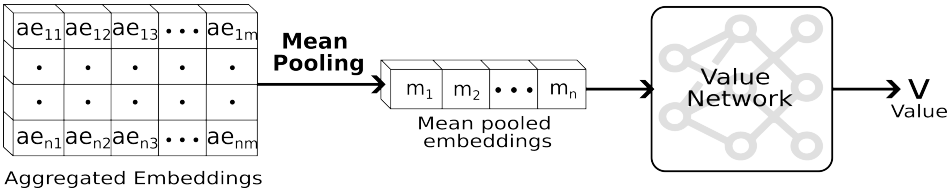


Figure 7. The aggregated embeddings gets mean pooled. The resultant vector is processed by a simple MLP which then goes through a linear network in order to produce a single value.

5. Experimental Results

In this section, we evaluate the performance of our proposed policy architecture within the exploration framework described in Section 4. Our goal is to demonstrate the efficiency of our method in reducing the total path length over the course of exploration compared to classical heuristic methods that serve as adapted learning-based baselines.

5.1. Training Setup

Our policy was trained using the PPO algorithm over a series of episodes in a simulated 2D unstructured environment. The agent receives observations in the form of a 200x200 pixel occupancy grid and frontier scalar features, as detailed in Equation 1. In the subsequent section, a comprehensive explanation of the reward mechanism's design is provided. Here, an *episode* refers to the entire exploration process. An episode concludes when the robot's frontier generator no longer provides

 doi:10.20944/preprints202604.1959.v1

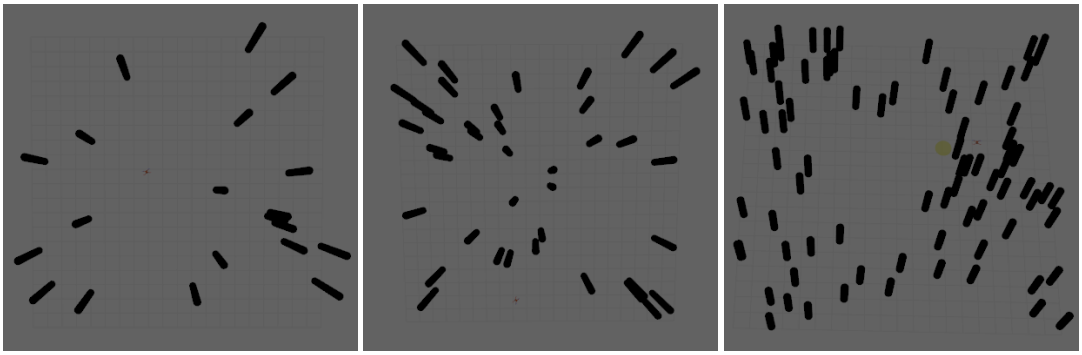
frontiers given the provided map. A *step* refers to the process of getting the policy output as the next frontier to explore, generating the path to the chosen frontier, and navigating to that point. To train our policy, we conducted training over **200k** steps, which lasted about 8 hours to finish training using an i7 13620H paired with a NVIDIA RTX 4060. Additional training hyper-parameters are found in Section 5.3.

Reward configuration: The reward mechanism is crucial, as it directly influences both the training efficiency and the overall effectiveness of deep reinforcement learning. To ensure that the rewards are not overly sparse, a reward is given after each exploration step. Specifically, the reward is defined as the negative value of the path length traversed by the robot during that step, as expressed in the following formula:

$$r_i = -l_i / \sqrt{2}S, \tag{2}$$

where r_i denotes the reward received and l_i represents the distance traveled by the robot at the i-th exploration step. l_i is normalized by the maximum possible straight distance (diagonal) $\sqrt{2}S$ in the squared grid.

Simulated environment: The simulation environment is executed using *Gazebo* simulator. We used a randomized unstructured 20x20 meter 2D map designed to mimic a forest-like unstructured environment with parameters such as varying numbers of obstacles, minimum separation between obstacles, and locations, as shown in Figure 8. These maps were randomly generated to evaluate the generalization capabilities of the proposed policy. In order to randomize the scenario during execution, we use a Gazebo service-to-service bridge to move, remove, and add obstacles within a range. We also ensure that the initial agent position is within a safety obstacle distance.



(**a**) Low (**b**) Medium (**c**) High

Figure 8. Different examples of unstructured scenario complexity. (a) is considered a low-density scenario with 20 obstacles. (b) is considered a medium-density scenario with 40 obstacles. (c) is considered a high-density scenario with a maximum of 80 obstacles.

5.2. Training Parameters

We have performed training with multiple combinations of hyper-parameters. The ones that worked best for us are in Table 2.

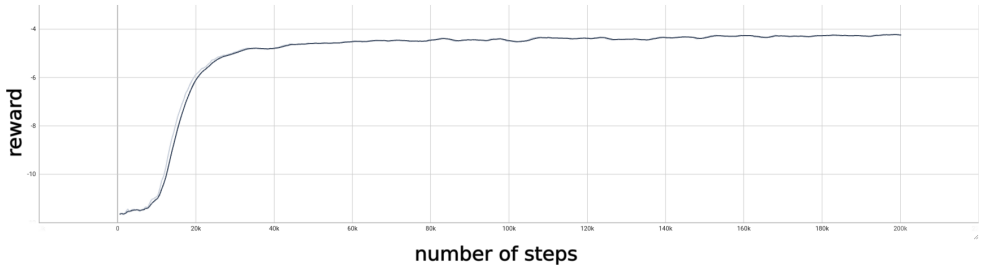
 doi:10.20944/preprints202604.1959.v1

Table 2. Training hyper-parameters.

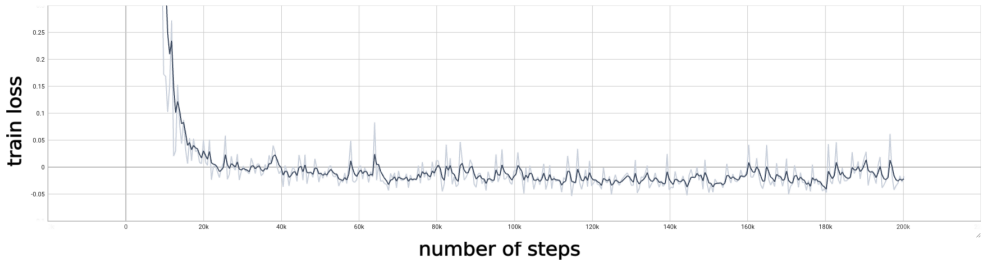
Parameter	Value
Learning rate	**0.0003**
Rollout steps (n_steps)	**512**
Batch size	**64**
Number of epochs	**5**
Discount factor (γ)	**0.99**
GAE lambda	**0.95**
Clip range	**0.2**
Clip range VF	**None**
Normalize advantage	**True**
Entropy coefficient	**0.0**
Value function coefficient	**0.5**
Max gradient norm	**0.5**

5.3. Final Training Metrics

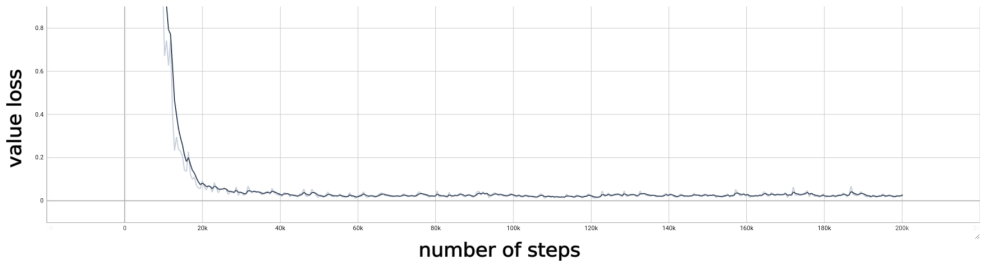
During training, the reward evolution is given by the following Figure 9(**a**). The loss of the policy network (actor) during training is given in Figure 9(**b**); lastly, the loss of the value network (critic) is given in Figure 9(**c**).



(**a**) Reward.



(**b**) Train Loss.



(**c**) Value Loss.

Figure 9. Training evolution results.

 doi:10.20944/preprints202604.1959.v1

5.4. Policy Evaluation

Let $\mathcal{F}$ denote the set of candidate frontier points, $\mathbf{p}_{\text{robot}} \in \mathbb{R}^3$ the current position of the robot, and $\mathbf{p}_f \in \mathbb{R}^3$ the position of a candidate frontier $f \in \mathcal{F}$. We compare the exploration efficiency of our proposed policy against the following baseline methods:

- **Random Sample:** The next frontier is drawn uniformly at random from the frontier set:

$$f^* = \text{UniformRandom}(\mathcal{F}) \tag{3}$$

  where f^* is the selected frontier.
- **Nearest Frontier:** A greedy strategy that selects the frontier closest to the robot in Euclidean distance:

$$f^* = \arg\min_{f \in \mathcal{F}} \left\| \mathbf{p}_{\text{robot}} - \mathbf{p}_f \right\|_2 \tag{4}$$

  where $\|\cdot\|_2$ denotes the L_2 (Euclidean) norm. [1].
- **Information Gain:** For each candidate frontier, the information gain is estimated as the number of unknown cells that would become visible from that frontier:

$$\text{IG}(f) = \left| \left\{ c \in \mathcal{C}_{\text{unknown}} \;\middle|\; \left\| c - \mathbf{p}_f \right\| \le r_s \;\wedge\; \text{visible}(c,\, \mathbf{p}_f) \right\} \right| \tag{5}$$

$$f^* = \arg\max_{f \in \mathcal{F}} \text{IG}(f) \tag{6}$$

  where $\mathcal{C}_{\text{unknown}}$ is the set of grid cells whose occupancy state is unknown, $c \in \mathbb{R}^3$ is the position of a grid cell, $r_s > 0$ is the maximum sensor range, and $\text{visible}(c,\, \mathbf{p}_f) \in \{\text{true}, \text{false}\}$ is a predicate that evaluates to true if and only if cell c lies within the sensor's field of view from $\mathbf{p}_f$ with an unoccluded line of sight [15].
- **Hybrid Approach:** A multi-objective strategy that combines distance cost and information gain through a weighted score [12]:

$$f^* = \arg\max_{f \in \mathcal{F}} \left[\alpha \, \frac{\text{IG}(f)}{\text{IG}_{\max}} - (1 - \alpha) \, \frac{d\left(\mathbf{p}_{\text{robot}},\, \mathbf{p}_f\right)}{d_{\max}} \right] \tag{7}$$

  where $\alpha \in [0, 1]$ is a tunable weight that trades off exploration reward against travel cost, $\text{IG}_{\max} = \max_{f \in \mathcal{F}} \text{IG}(f)$ is the maximum information gain over all current frontiers (used for normalization), $d(\mathbf{p}_{\text{robot}},\, \mathbf{p}_f) = \left\| \mathbf{p}_{\text{robot}} - \mathbf{p}_f \right\|_2$ is the Euclidean distance from the robot to the frontier, and $d_{\max} = \max_{f \in \mathcal{F}} d(\mathbf{p}_{\text{robot}},\, \mathbf{p}_f)$ is the maximum such distance (used for normalization). Three weight configurations are evaluated: $\alpha \in \{0.25,\ 0.50,\ 0.75\}$.
- **TARE Local:** The local planner from the TARE framework [14] scores candidate sensor viewpoints by how much new surface area they would reveal, selects a compact subset via a greedy-randomized procedure, and returns the first waypoint of the resulting route as the next frontier.

$$\text{score}(v) = \left| \left\{ s \in \mathcal{S}^{\text{local}}_{\text{unseen}} \;\middle|\; \text{visible}(s,\, v) \right\} \right| \tag{8}$$

$$V^* = \text{GRASP}\Big(\{v_i\}_{i=1}^{N_v},\, \text{score} \Big) \tag{9}$$

$$\tau^* = \text{TSP}(V^*), \qquad f^* = \tau^*_1 \tag{10}$$

  where $v \in \mathbb{R}^3$ is a candidate sensor viewpoint sampled within the local planning window, N_v is the total number of sampled viewpoints, $\mathcal{S}^{\text{local}}_{\text{unseen}}$ is the set of unseen surface elements within the high-resolution local map centered on the robot, s is an individual surface element in that set, $\text{visible}(s,\, v)$ is a predicate that is true when surface element s is observable from viewpoint v, $\text{GRASP}(\cdot)$ denotes the Greedy Randomized Adaptive Search Procedure that selects a compact

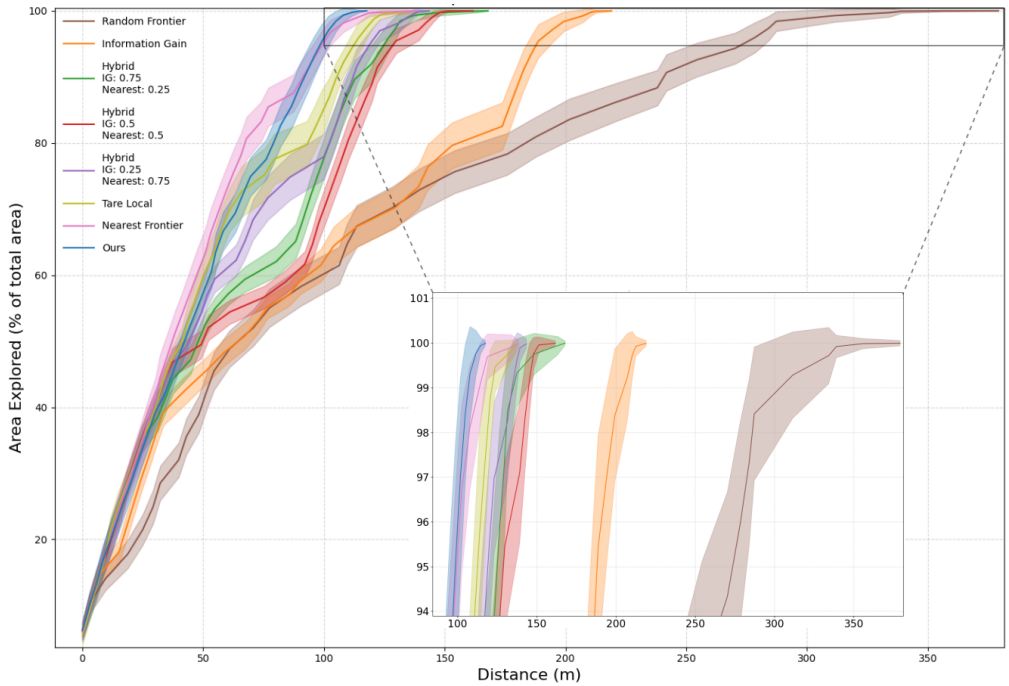
 doi:10.20944/preprints202604.1959.v1

subset $V^* \subseteq \{v_i\}$ of high-scoring viewpoints, TSP(V^*) computes a short visitation tour τ^* through the selected viewpoints that respects robot motion constraints, and τ_1^* is the first stop on that tour, which is returned as the next frontier f^*.

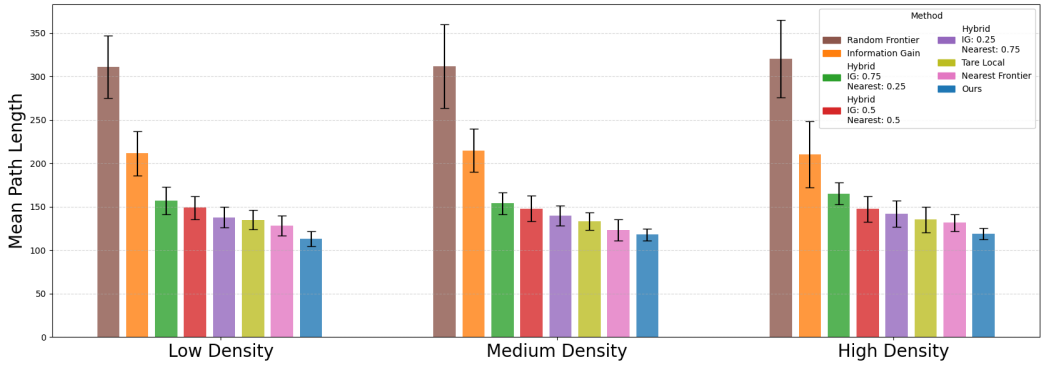
Evaluation metric. Our primary goal is to develop a policy suitable for deployment on an aerial robot, where minimizing the total path length is essential to reduce exploration time and extend mission duration, both critical factors for autonomous operation. To evaluate the efficiency of exploration, we measure the average total path length required to fully explore each scenario, providing a direct measure of the policy's spatial efficiency, along with the deviation to assess consistency.

5.5. Simulation Results and Analysis

We compared our approach to baselines, evaluating each method over `100 episodes` across `100 different maps` for each density scenario. As shown in Figure 10, our trained policy outperforms all heuristics across all different density maps, achieving shorter paths and faster exploration times.



(**a**) Distance to explore the full map for the low density scenario. Standard deviation is shown in shadow.



(**b**) Mean path length with standard deviation for the three scenarios.

Figure 10. Path length and area coverage comparison over 100 episodes (lower is better).

Table 3 summarizes the numerical comparison. Compared to the best-performing heuristic in unstructured environments, we perform **13.1%** better in low-density scenarios, **4.4%** better in mid-

 doi:10.20944/preprints202604.1959.v1

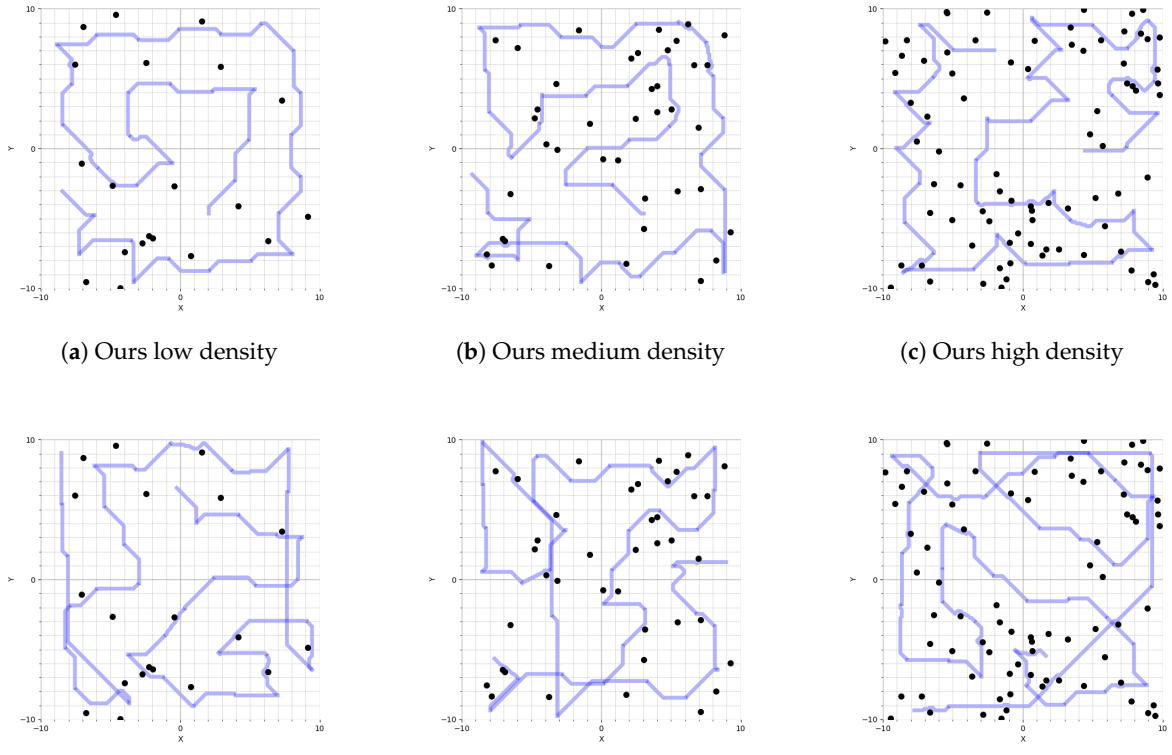
density scenarios, and **10.6%** better in high-density scenarios. Results also show a better standard deviation than all heuristic methods, demonstrating better consistency over the randomly generated scenarios at each obstacle density.

Our method generalizes well across different environments: in all three test scenarios, it consistently outperforms existing baseline strategies in terms of both mean and standard deviation in path length minimization. We also conclude that our method does not fall short when increasing the number of obstacles, obtaining similar results for the three scenarios.

Table 3. Exploration performance results. Mean path length and standard deviation across three scenarios are in meters.

	20 Obstacles		**40 Obstacles**		**80 Obstacles**	
Method	Mean Path Length	Std. Dev.	Mean Path Length	Std. Dev.	Mean Path Length	Std. Dev.
Random	311.32	± 35.92	311.83	± 48.10	320.51	± 44.33
Information Gain	211.58	± 25.47	214.98	± 24.68	210.27	± 38.34
Nearest Frontier	128.33	± 11.32	123.14	± 12.41	131.63	± 9.97
Hybrid 75-25	157.19	± 15.92	154.02	± 12.76	165.12	± 12.62
Hybrid 50-50	148.86	± 13.07	147.83	± 14.80	147.45	± 14.83
Hybrid 25-75	137.85	± 11.90	139.77	± 11.48	142.05	± 14.82
TARE Local	134.99	± 11.13	133.30	± 9.84	135.29	± 14.66
Ours	**113.42**	± **8.74**	**117.98**	± **6.98**	**119.02**	± **6.44**

To provide qualitative insight into the behavior of our policy, we include top-view visualizations of representative exploration trajectories in each test scenario in Figure 11. These plots illustrate the paths followed by the agent during the exploration process in each density map, together with the paths followed by the best heuristic method for these unstructured scenarios in our tests, Nearest Frontier. Our method achieves visually shorter paths with close to no overlap in comparison with the less efficient paths achieved by the Nearest Frontier heuristic, which are longer and overlap in several cases.



(**a**) Ours low density (**b**) Ours medium density (**c**) Ours high density

(**d**) Nearest low density (**e**) Nearest medium density (**f**) Nearest high density

Figure 11. Visual comparison between our method and the best heuristic method for unstructured environments (Nearest) for each type of density map.

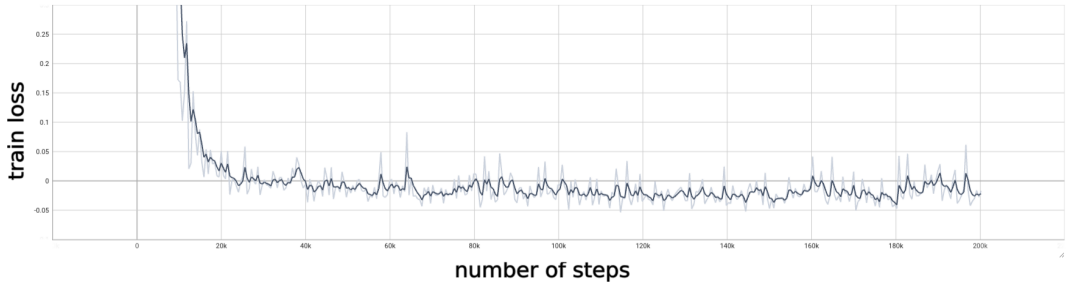
 doi:10.20944/preprints202604.1959.v1

5.6. Ablation Study

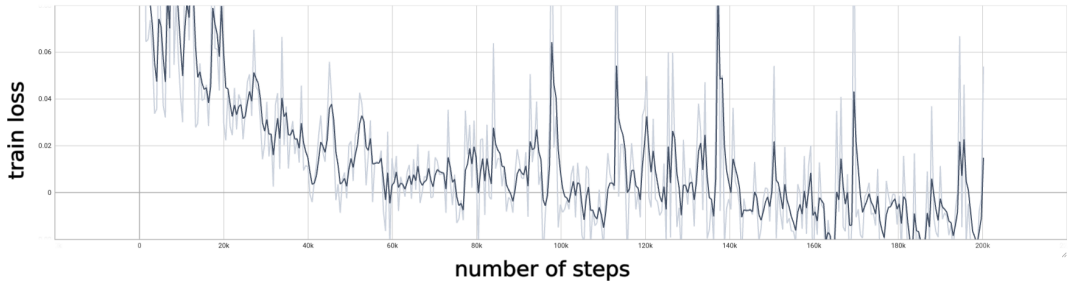
To evaluate the contribution of the cross-attention module within our policy architecture, we conducted an ablation study comparing the full model against a baseline variant without the attention mechanism. In the ablated version, for the non-attentive model, the global map features and local frontier features are concatenated directly before being passed to the scoring network, effectively removing the selective feature interaction enabled by attention. This comparison isolates the impact of the attention module on the model's ability to prioritize informative frontiers and adapt to varying spatial configurations.

Table 4. Key metrics demonstrating Attention vs. no Attention

Metric	No Attention	Attention
Reward Peak (`ep_rew_mean`)	−5.1341	−4.2172
Reward Std (last 25% of training)	0.1180	0.0291
Std Norm Last 25% of Train Loss (normalized)	0.0987	0.00693
Std Norm Last 25% of Value Loss (normalized)	0.00667	0.00089
Residual Std of Normalized Train Loss (w=10)	0.1011	0.0190
Mean Abs Diff of Normalized Train Loss	0.0958	0.0116

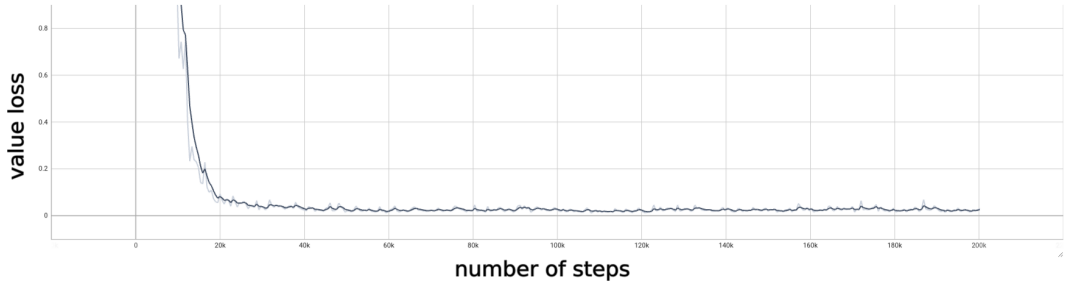


(**a**) Policy loss with attention



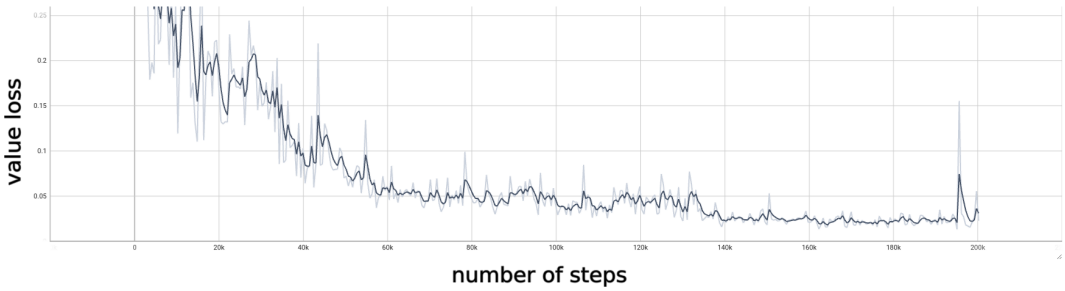
(**b**) Policy loss with no attention

Figure 12. Policy losses: attention vs. no attention.



(**a**) Value network loss with attention

 doi:10.20944/preprints202604.1959.v1



(**b**) Value network loss with no attention

Figure 13. Value network losses: attention vs. no attention.

Given these ablation metrics, we can conclude that our approach helps achieving:

- **Higher peak reward:** The Attention model achieves a peak episode return of -4.2172, compared to -5.1341 without attention, indicating a stronger final policy.
- **More stable rewards:** Attention reduces the standard deviation of episode returns in the final quarter of training from 0.1180 to 0.0291, showing smoother and more reliable performance improvements.
- **Smoother loss convergence:** When normalized to $[0, 1]$, the last-quarter standard deviation of train loss drops from 0.0987 to 0.00693, and of value loss from 0.00667 to 0.00089, confirming that Attention yields much steadier optimization in the later stages.
- **Lower high-frequency noise:** The residual std around a rolling-mean baseline (window = 10) of normalized train loss falls from 0.1011 to 0.0190, and the mean absolute step-to-step change from 0.0958 to 0.0116, further underscoring cleaner gradient updates with Attention.

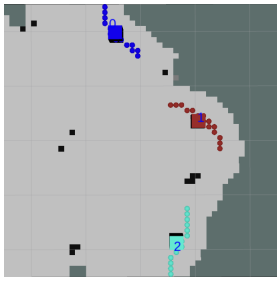
5.7. Real Flight Experiments

Our framework is easily transferable to a real environment as it uses the Arias-Perez [8] ROS 2 setup, replacing only the frontier decider for our policy network. The framework is already validated in real experiments in the previously mentioned work, which concluded that the real sensor data highly correlates with the simulated sensor data used in this study. Besides the ablation study of the attention module provided in Section 5.6, we one-shot transfer the policy into a real scenario and perform real experiments with the entire minimal sensing hardware and framework pipeline.

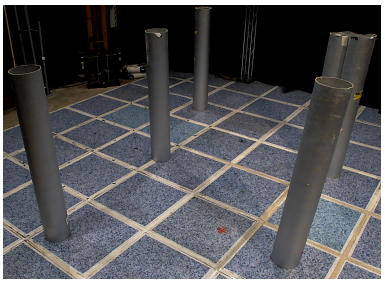
In order to validate our policy in a real world scenario, we performed several runs in a manually crafted scenario. The setup for our real scenario is a 5x5 meter arena, as depicted in Figure 14(**c**), with 5 to 6 randomly placed poles. The drone initial position is randomized between runs. Localization is provided by an Optitrack motion capture system. The drone used is a Bitcraze Crazyflie, depicted in Figure 14(**a**), equipped with the minimal sensing setup as in [8]. It consists of a four point LiDAR sensor pointing front, back, left, and right directions relative to the drone's frame in order to build a map around it (see Figure 14(**b**)). Contrary to simulation training, where we use 360º lidar to train without any control loop, drone control is back enabled in the real setup, and the drone performs a 90 deg rotations when reaching every frontier in order to map around it. Estimation data is obtained through a 16 camera MoCap Optitrack Array.

 doi:10.20944/preprints202604.1959.v1

(**a**) Bitcraze Crazyflie equipped with multi-ranger deck.



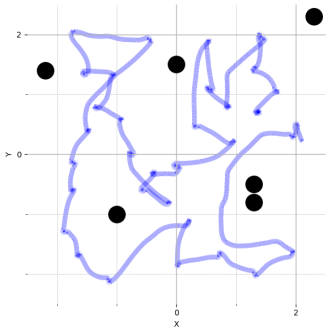
(**b**) Occupancy grid generated with visible frontiers during real experiments.



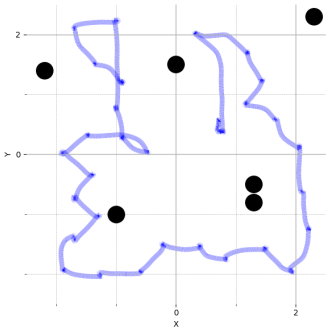
(**c**) One of the real scenario setups used.

Figure 14. Main equipment used for the real experiments.

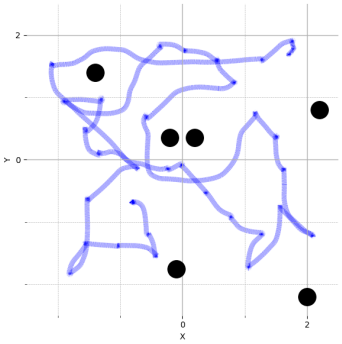
We provide the same top-view style of the performed real experiments in the Figure 15. Figures 15(**a**), 15(**b**), 15(**c**), 15(**d**) are done within the same scenario layout, differing from each other in the starting point. Figures 15(**e**) and 15(**f**) are done in different scenarios. Results obtained in layout B are inconclusive due to the lack of possible choices (frontiers) in certain points of the exploration as the frontier generator was only giving one single choice, so the policy was forced to select that one single frontier.



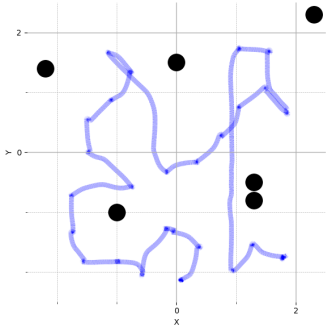
(**a**) Scenario A1



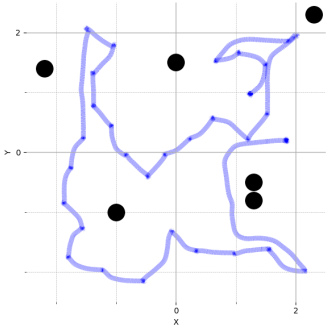
(**b**) Scenario A2



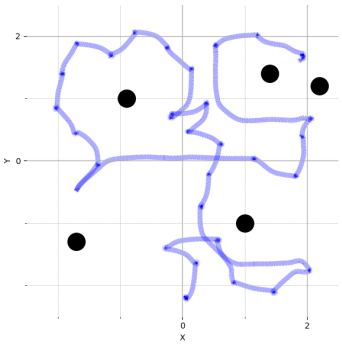
(**e**) Scenario B



(**c**) Scenario A3



(**d**) Scenario A4



(**f**) Scenario C

Figure 15. Top-view of the followed paths during the real experiments (blue lines), with the real obstacle distribution (black dots).

These experiments demonstrate three key contributions.

1. Our policy achieves zero-shot sim to real transfer by using the same laser generated occupancy grid map, together with the frontier scalar features.
2. We demonstrate that even though the policy has been trained with a 200x200 map generated from a 400 squared meter simulated scenario, we are able to upscale the 50x50 map generated

 doi:10.20944/preprints202604.1959.v1

from a 25 squared meter real scenario, giving the policy the flexibility to use state matrices of different sizes.
3. Our policy performs with consistent behavior in minimizing the distance per percentage of area explored for which it was trained.

6. Conclusion

In this work, we present a novel policy architecture for autonomous frontier-based exploration that introduces a cross-attention module to fuse global map features extracted by a CNN with local frontier features represented as scalar values. Unlike prior RL-based exploration methods that rely on fixed-size action spaces, imitation learning priors, or padded visual feature maps, our approach operates as a true variable frontier selector through a scoring network combined with cross-attention, trained end-to-end with PPO. The residual connection preserves original local frontier information while enriching each candidate with global contextual cues. Simulation experiments across three obstacle density levels demonstrate that our method consistently outperforms all evaluated heuristic baselines, achieving reductions in mean path length of 13.1%, 4.4%, and 10.6% relative to the best-performing heuristic in low, medium, and high density scenarios respectively, with lower standard deviation across all cases. The ablation study confirms the attention module's contribution, yielding a stronger final policy with substantially smoother convergence. Furthermore, zero-shot sim-to-real transfer on a Bitcraze Crazyflie with minimal sensing validates the approach's practical applicability, demonstrating flexibility across different input resolutions without retraining.

Limitations and future work. Despite the observed performance gains, our approach presents several limitations. First, the reward function can be further improved to address suboptimal behaviors. Future work will investigate reward shaping strategies that encourage early coverage and promote a more uniform exploration rate across available frontiers. Second, the method currently requires explicit frontier extraction and hand-crafted feature computation, which introduces an additional processing step and limits end-to-end learning. Integrating frontier detection and representation learning directly into the policy architecture is a promising direction to enhance scalability and autonomy. Finally, although we use PPO for training, recent advances such as Generalized Reinforcement Policy Optimization (GRPO) [30] may yield better sample efficiency and training stability and are worth exploring in future iterations of this work.

Author Contributions: Conceptualization, J.M.D., P.A.P. and M.M.; Methodology, J.M.D. and M.M.; Software, J.M.D.; Validation, J.M.D., P.A.P. and M.M.; Formal analysis, J.M.D., G.GP.L. and M.M.; Investigation, J.M.D. and M.M.; Resources, J.M.D.; Data curation, J.M.D.; Writing—original draft, J.M.D.; Writing—review & editing, J.M.D., P.A.P, G.GP.L., M.M. and P.C.; Visualization, J.M.D.; Supervision, P.A.P., M.M. and P.C.; Project administration, P.C.; Funding acquisition, P.C. All authors have read and agreed to the published version of the manuscript.

Funding: The work of the first author is supported by the Community of Madrid under its Program for Predoctoral Researcher with Ref. PIPF-2024/TEC-34898 (BOCM - 5021/2025)

Data Availability Statement: Available public open-source code and docker can be found in Github Repository. (public ros2 repositories sometimes have their keys outdated so docker build may fail, the authors of this work commit to keep the docker and test code working). Video of the paper can be found in Vimeo.

DURC Statement: Current research is limited to autonomous exploration of unstructured civilian environments for search and rescue applications, which is beneficial for disaster response and does not pose a threat to public health or national security. The authors acknowledge the dual-use potential of UAV autonomy research and confirm that all necessary precautions have been taken to prevent potential misuse. As an ethical responsibility, the authors strictly adhere to relevant national and international laws about DURC.

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

 doi:10.20944/preprints202604.1959.v1

Abbreviations

The following abbreviations are used in this manuscript:

RL: Reinforcement Learning
PID: Proportional Integral Derivative
MLP: Multilayer Perceptron
CNN: Convolutional Neural Network
PPO: Proximal Policy Optimization

References

1. Yamauchi, B. A frontier-based approach for autonomous exploration. In *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*; 1997; pp. 146–151.
2. Amigoni, F. Experimental evaluation of some exploration strategies for mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*; 2008; pp. 2818–2823.
3. Senarathne, P.N.; Wang, D. Incremental algorithms for safe and reachable frontier detection for robot exploration. *Robotics and Autonomous Systems* **2015**, *72*, 189–206.
4. Lu, L.; Redondo, C.; Campoy, P. Optimal frontier-based autonomous exploration in unconstructed environment using RGB-D sensor. *Sensors* **2020**, *20*, 6507.
5. K. Leong. Reinforcement Learning with Frontier-Based Exploration via Autonomous Environment. *arXiv* **2023**, arXiv:2307.07296. https://arxiv.org/abs/2307.07296.
6. Garaffa, L.C.; Basso, M.; Konzen, A.A.; Freitas, E.P. Reinforcement learning for mobile robotics exploration: A survey. *IEEE Transactions on Neural Networks and Learning Systems* **2023**, *34*, 3796–3810.
7. Singh, B.; Kumar, R.; Singh, V.P. Reinforcement learning in robotic applications: a comprehensive survey. *Springer Artificial intelligence review* **2022**, *55*, pp. 945–990. https://doi.org/10.1007/s10462-021-09997-9
8. Arias-Perez, P.; Gautam, A.; Fernandez-Cortizas, M.; Perez-Saura, D.; Saripalli, S.; Campoy, P. Exploring unstructured environments using minimal sensing on cooperative nano-drones. *IEEE Robotics and Automation Letters* **2024**, *9*, 11202–11209. https://doi.org/10.1109/LRA.2024.3486212.
9. Fernandez-Cortizas, M.; Molina, M.; Arias-Perez, P.; Perez-Segui, R.; Perez-Saura, D.; Campoy, P. Aerostack2: A software framework for developing multi-robot aerial systems. *arXiv* **2024**, arXiv:2303.18237.
10. Zhao, H.; Gao, J.; Lan, T.; Sun, C.; Sapp, B.; Varadarajan, B.; Shen, Y.; Shen, Y.; Chai, Y.; Schmid, C.; Li, C.; Anguelov, D. TNT: Target-driven trajectory prediction. *arXiv* **2020**, arXiv:2008.08294.
11. Vaswani, A.; et al. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*; 2017; pp. 5998–6008.
12. Basilico, N.; Amigoni, F. Exploration strategies based on multi-criteria decision making for searching environments in rescue operations. *Autonomous Robots* **2011**, *31*, 401–417. https://doi.org/10.1007/s10514-011-9249-9.
13. Macenski, S.; Foote, T.; Gerkey, B.; Lalancette, C.; Woodall, W. Robot Operating System 2: Design, architecture, and uses in the wild. *Science Robotics* **2022**, *7*, eabm6074.
14. Cao, C.; Zhu, H.; Choset, H.; Zhang, J. TARE: A hierarchical framework for efficiently exploring complex 3D environments. In *Robotics: Science and Systems (RSS)*; 2021. https://doi.org/10.15607/RSS.2021.XVII.018.
15. Burgard, W.; Moors, M.; Stachniss, C.; Schneider, F. Coordinated multi-robot exploration. *IEEE Transactions on Robotics* **2005**, *21*, 376–386. https://doi.org/10.1109/TRO.2004.839232.
16. González-Baños, H.H.; Latombe, J.-C. Navigation strategies for exploring indoor environments. *International Journal of Robotics Research* **2002**, *21*, 829–848. https://doi.org/10.1177/027836490202101083 4.
17. Umari, H.; Mukhopadhyay, S. Autonomous robotic exploration based on multiple rapidly-exploring randomized trees. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*; 2017; pp. 1396–1402.
18. Deng, D.; Duan, R.; Liu, J.; Sheng, K.; Shimada, K. Robotic exploration of unknown 2D environment using a frontier-based automatic-differentiable information gain measure. In *Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*; 2020. https://doi.org/10.1109/AIM43001.2020.9158881.
19. Sun, Z.; Wu, B.; Xu, C.-Z.; Sarma, S.E.; Yang, J.; Kong, H. Frontier detection and reachability analysis for efficient 2D graph-SLAM based active exploration. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*; 2020. https://doi.org/10.1109/IROS45743.2020.9341735.

 doi:10.20944/preprints202604.1959.v1

20. Zhang, T.; Yu, J.; Li, J.; Pang, M. A heuristic autonomous exploration method based on environmental information gain during quadrotor flight. *International Journal of Advanced Robotic Systems* **2024**, *21*. https://doi.org/10.1177/17298806231151187.
21. Fan, J.; Zhang, X.; Zou, Y. Hierarchical path planner for unknown space exploration using reinforcement learning-based intelligent frontier selection. *Expert Systems with Applications* **2023**, *230*, 120630. https://doi.org/10.1016/j.eswa.2023.120630.
22. Yu, B.; Kasaei, H.; Cao, M. Frontier semantic exploration for visual target navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*; 2023; pp. 4099–4105.
23. Iyer, G.; Jain, V.; Pratapa, A.; Setlur, A. Learning frontier selection for navigation in unseen structured environment. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*; 2020; pp. 2130–2137.
24. Niroui, F.; Zhang, K.; Kashino, Z.; Nejat, G. Deep reinforcement learning robot for search and rescue applications: Exploration in unknown cluttered environments. *IEEE Robotics and Automation Letters* **2019**, *4*, 610–617.
25. Cao, Y.; Hou, T.; Wang, Y.; Yi, X.; Sartoretti, G. Ariadne: A reinforcement learning approach using attention-based deep networks for exploration. *arXiv* **2023**, arXiv:2301.11575.
26. Liu, Z.; Deshpande, M.; Qi, X.; Zhao, D.; Madhivanan, R.; Sen, A. Learning to Explore (L2E): Deep reinforcement learning-based autonomous exploration for household robot. In *Robotics: Science and Systems (RSS) Workshop*; 2023.
27. Wang, R.; Zhang, J.; Lyu, M.; Yan, C.; Chen, Y. An improved frontier-based robot exploration strategy combined with deep reinforcement learning. *Robotics and Autonomous Systems* **2024**, *181*, 104783.
28. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal policy optimization algorithms. *arXiv* **2017**, arXiv:1707.06347.
29. Raffin, A.; Hill, A.; Gleave, A.; Kanervisto, A.; Ernestus, M.; Dormann, N. Stable-Baselines3: Reliable Reinforcement Learning Implementations. *Journal of Machine Learning Research* **2021**, *22*, 1–8.
30. Queeney, J.; Paschalidis, Y.; Cassandras, C.G. Generalized proximal policy optimization with sample reuse. In *Advances in Neural Information Processing Systems (NeurIPS)*; 2021; Vol. 34, pp. 11909–11919.
31. Schulman, J.; et al. Proximal policy optimization algorithms. *arXiv* **2017**, arXiv:1707.06347.