2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Platform-Agnostic Reinforcement Learning Framework for Safe Exploration of Cluttered Environments with Graph Attention

Gabriele Calzolari, Vidya Sumathy, Christoforos Kanellakis, George Nikolakopoulos

Abstract— **Autonomous exploration of obstacle-rich spaces requires strategies that ensure efficiency while guaranteeing safety against collisions with obstacles. This paper investigates a novel platform-agnostic reinforcement learning framework that integrates a graph neural network-based policy for next-waypoint selection, with a safety filter ensuring safe mobility. Specifically, the neural network is trained using reinforcement learning through the Proximal Policy Optimization (PPO) algorithm to maximize exploration efficiency while minimizing safety filter interventions. Henceforth, when the policy proposes an infeasible action, the safety filter overrides it with the closest feasible alternative, ensuring consistent system behavior. In addition, this paper introduces a reward function shaped by a potential field that accounts for both the agent's proximity to unexplored regions and the expected information gain from reaching them. The proposed framework combines the adaptability of reinforcement learning-based exploration policies with the reliability provided by explicit safety mechanisms. This feature plays a key role in enabling the deployment of learning-based policies on robotic platforms operating in real-world environments. Extensive evaluations in both simulations and experiments performed in a lab environment demonstrate that the approach achieves efficient and safe exploration in cluttered spaces.**

I. Introduction

Efficient and safe autonomous exploration in obstacle-rich environments, such as forests and mines, remains a critical challenge for robotic systems. These issues encompass optimal path planning, uniform coverage, and ensuring safety during decision making. Traditional exploration strategies, such as random exploration and frontier-based approaches, are often inadequate in real-world environments because they lack adaptability to evolving scenarios, suboptimal path planning, and take myopic decisions. On the other hand, Reinforcement Learning (RL)-based policies enable adaptive decision-making by optimizing long-term rewards and dynamically responding to environmental changes. Among these learning-based methods, Graph Neural Network (GNN)-based architectures have gained increasing attention due to their ability to represent spatial relationships within an environment using graph structures, allowing policies to extract and leverage meaningful features from critical locations in the scene when making navigation decisions. Unlike classical strategies, learned policies provide limited insight into the decision-making process and offer no explicit guarantees that they have acquired the correct behavior beyond statistical evaluation in testing environments. This issue is especially critical in safety-sensitive applications, such as flying a drone in a thick forest, where an agent must demonstrably adhere to strict constraints to prevent hazardous behavior. To address this, recent research is focusing on integrating safety mechanisms into RL-based frameworks, ensuring compliance with safety constraints while maintaining adaptability. One promising direction involves combining deep learning methods, capable of encoding complex behaviors, with safety filters that impose hard constraints on action selection, thereby preventing unsafe decisions. This hybrid approach improves the robustness and reliability of learned policies, facilitating their deployment in real-world environments.

A. Related works

Recently, reinforcement learning has had a profound impact on the development of autonomous robotic exploration approaches, enabling agents to safely navigate and explore cluttered environments achieving great efficiency [1]–[6]. For instance, [7] investigates a Deep Reinforcement Learning-based navigation approach for unmanned aerial vehicles that integrates a stochastic value function. By formulating navigation as a partially observable Markov decision process (POMDP), the method leverages the fast recurrent stochastic value-gradient algorithm to achieve safe and reliable exploration in large, cluttered three-dimensional environments. On the other hand, [8] proposes a strategy for target-oriented exploration in which waypoints are selected using a TD3-based DRL policy. This framework mitigates convergence to local optima while operating without prior maps or human intervention. Notably, [9] introduce ARiADNE which is an attention-based framework that allows real-time, non-myopic path planning by capturing spatial dependencies and predicting exploration gains. [10] proposes ReLMM that allows robots to learn how to navigate via reinforcement learning, using modular policies and uncertainty-driven exploration. Notably, literature shows that a recent line of research explores the integration of Graph Neural Networks with reinforcement learning for the implementation of exploration policy [11]–[13]. Such approaches effectively exploit the underlying spatial relationships between relevant locations in the environment, thereby enhancing decision-making dur-

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation, and by the European Union's Horizon Europe Research and Innovation Program, under the Grant Agreement No. 101119774 SPEAR. This research was conducted using the resources of High Performance Computing Center North (HPC2N). Additionaly, the RL-training were enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS), partially funded by the Swedish Research Council through grant agreement no. 2022-06725. The authors are within the Robotics and AI Group, Department of Computer Science, Electrical and Space Engineering, Luleå University of Technology, Sweden. Corresponding author's e-mail: gabcal@ltu.se

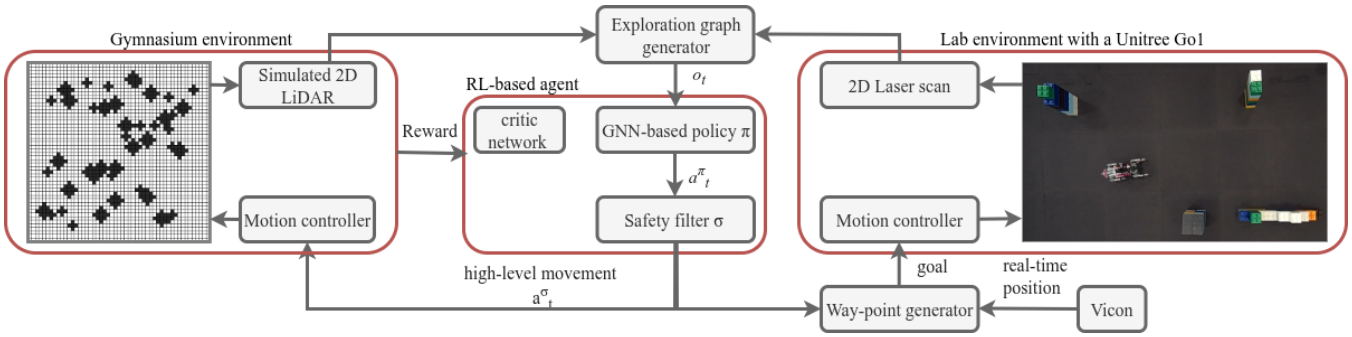


Fig. 1. Overview of the proposed safe reinforcement learning framework for exploration in cluttered environments. The hierarchical architecture integrates a GNN-based policy with a safety filter to generate the next-step waypoint for exploration. The resulting high-level movement can be applied either to a simulated environment in Gymnasium for policy training and ablation studies (left branch), or to the laboratory setting with the Unitree Go1 quadruped robot for physical experiments (right branch).

ing exploration. Indeed, [14] investigates a Spatiotemporal Neural Network on graph for efficient exploration relying on past trajectories, obstacles, and waypoints to estimate optimal targets. [15] presents a GNN-based method for autonomous exploration which generalizes across graph topologies and ensures high performances. The work proposed by [16] presents SGRL, which is a GNN-based deep Q-learning algorithm for autonomous driving decisions that leverages agent interactions. [17] proposes MGRL, a reinforcement learning-based visual navigation framework that models uncertainty through a Markov network, employs Graph Neural Networks for inference, and incorporates knowledge graphs to enhance generalizability and adaptability. To enhance the safety guarantees of reinforcement learning frameworks, Safe RL incorporates safety constraints via constrained optimization, risk-sensitive policy design, and robustness techniques, thereby promoting optimal decision-making while mitigating unsafe actions [18]–[22]. For instance, [23] proposes Constrained Policy Optimization (CPO), an algorithm that enforces constraint satisfaction during policy learning while simultaneously optimizing performance, thereby enabling neural networks to address high-dimensional tasks with formal safety guarantees. [24] presents the Conservative Update Policy (CUP) algorithm, a safe reinforcement learning approach that provides theoretical safety guarantees by leveraging surrogate functions and establishing tighter performance bounds. [25] investigates SafeDreamer, an approach that integrates Lagrangian methods into the Dreamer framework for safe decision-making, achieving near-zero-cost performance in vision-only and low-dimensional tasks in the Safety-Gymnasium benchmark. [26] introduces D2RL, a learning-based approach that accelerates autonomous vehicle safety testing by densifying training data around critical states, thereby expediting evaluation while preserving statistical unbiasedness. Augmented Proximal Policy Optimization (APPO) is proposed by [27] and is a safety-constrained RL approach that improves convergence and cost control by adding a quadratic penalty to the Lagrangian function. [28] proposes Unrolling Safety Layer (USL), a method combining safety optimization and projection to enforce state-wise safety constraints in model-free RL. Finally, [29] presents the Conservative and Adaptive Penalty (CAP) framework, a model-based Safe Reinforcement Learning method that dynamically adjusts an uncertainty-aware penalty to balance reward maximization and cost minimization, thereby ensuring safety in both tabular and high-dimensional environments.

Inspired by prior studies, we investigate a novel platform-agnostic framework for safe autonomous exploration in cluttered environments. The exploration policy follows a hierarchical two-stage process: first, a Graph Neural Network, trained with reinforcement learning, selects a candidate waypoint; then, a safety filter evaluates its feasibility and, if necessary, adjusts it to ensure collision-free navigation. The most significant contributions of this work are as follows.

- A platform-agnostic hierarchical framework based on reinforcement learning that combines a GNN-driven exploration policy with a novel safety filter, which intervenes by replacing motions violating safety constraints with feasible ones.
- A novel attention-enhanced graph neural network that implements the exploration policy by extracting exploration-relevant features from a custom graph-based representation of the environment, which encodes waypoints, frontiers, and occluded regions.
- A proposed reward function that promotes consistent map expansion using potential field variations tied to frontier information gain and distance, validated through ablation studies.
- A validation of the proposed framework through simulations and experiments in a lab environment with a Unitree Go1 robotic platform.

II. Methodology

This section presents the exploration framework formulation in II-A, the design of the agent's observation space in II-B, the GNN architecture implementing the policy and critic in II-C, and the reward function formulation in II-D.

A. Platform-agnostic exploration framework

The overall structure of the proposed hierarchical framework operating in obstacle-rich environments is depicted in Fig. 1. Specifically, the agent is tasked to explore an arena

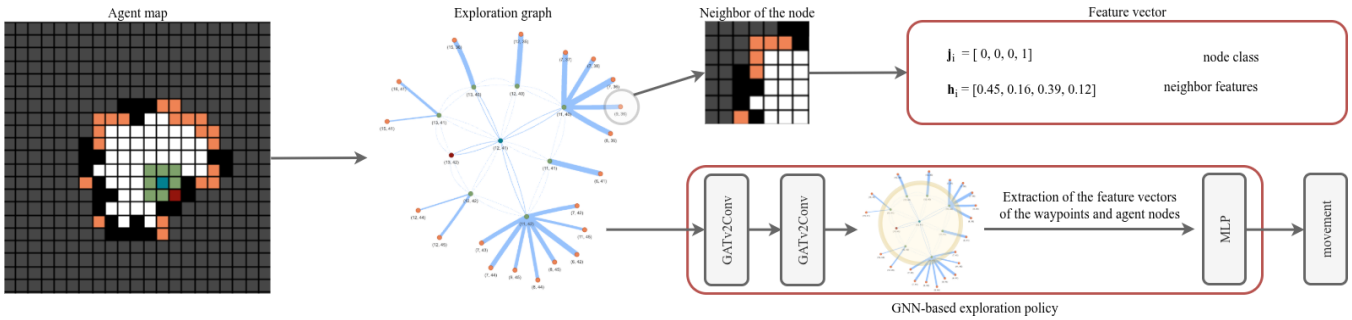


Fig. 2. Illustration of the exploration graph used as the observation o_t and the GNN-based policy. The image shows a section of the agent's map, where occupied, unknown, and free cells are represented in black, gray, and white, respectively. The agent's position, feasible and infeasible next-step navigation goals, and frontiers are indicated in blue, green, red, and orange, respectively. The graph structure encodes node relationships, with edge thickness proportional to the distance between connected nodes. On the right, the local neighborhood of a frontier and its extracted feature vector are highlighted, together with the internal architecture of the exploration policy, where the main layers are emphasized.

that is represented as an occupancy grid map $\mathcal{M}_{h\times w}$, with grid resolution r_g, containing n_o circular non-traversable regions $\mathcal{O} = \{o(c_i, r_i)\}_{i=1}^{n_o}$, with radii $r_i \sim \mathcal{U}[r_{\min}, r_{\max}]$ and centers c_i randomly distributed in the environment. At each time step t, the exploring agent occupies a grid cell v_t^a and has a partial map $\mathcal{M}_t^a$ of the reconstructed environment that contains only explored regions. Upon reaching a goal, the agent updates $\mathcal{M}_t^a$ using simulated 360-degree LiDAR measurements with range l. From this map, a graph-based observation o_t is constructed for the policy, encoding the agent's position, navigation goals, and the frontiers between traversable and unknown regions, as described in II-B.

As illustrated in Fig. 1, the hierarchical exploration strategy combines an RL-trained policy $\pi_\theta(o_t)$, which selects a movement $a_t^\pi \in \mathcal{A}$ towards one of the eight neighboring cells, with a safety filter $\sigma(a_t^\pi)$ that ensures that the agent's movement is feasible. The agent's action space comprises the movements to its adjacent eight cells in the grid and a null action corresponding to no displacement. Formally, if the agent is at position $(i, j) \in \mathcal{M}_{h\times w}$, then an action $a = (\Delta i, \Delta j) \in \mathcal{A}$ produces the transition

$$(i, j) \;\mapsto\; (i + \Delta i,\; j + \Delta j),$$

where the action space is defined as

$$\mathcal{A} = \{(\Delta i, \Delta j) \mid \Delta i, \Delta j \in \{-1, 0, 1\}\}.$$

This yields $|\mathcal{A}| = 9$, accounting for radial (axial), diagonal, and null motions of the agent. In particular, the safety filter validates the policy-selected action a_t^π considering the agent's reconstructed map. If the action is found feasible, then it is executed unchanged. Otherwise, the filter substitutes a_t^π with the closest feasible action $a_t^\sigma \in \mathcal{A}_t^f$, determined by minimizing the angular deviation from a_t^π, i.e., by selecting the feasible action that maximizes the cosine similarity with the original policy action, as in Eq. (1).

$$a_t^\sigma = \begin{cases} a_t^\pi, & \text{if } a_t^\pi \in \mathcal{A}_t^f \\ \arg\max\limits_{a\in\mathcal{A}_t^f} \dfrac{\langle a_t^\pi, a\rangle}{\|a_t^\pi\|\,\|a\|}, & \text{otherwise} \end{cases} \tag{1}$$

where $\langle\cdot,\cdot\rangle$ denotes the scalar product, $\mathcal{A}_t^f \subseteq \mathcal{A}$ indicates the feasible action set containing all actions that keep the agent within the map boundaries, avoid collisions with non-traversable cells, and, for diagonal moves, require that at least one of the adjacent orthogonal moves is traversable to prevent corner-cutting collisions. This safety check guarantees that the action selected will not cause a collision. The high-level movement a_t^σ is then applied to move the agent to a new location, and this process is repeated until the exploration ratio satisfies $\rho_t \geq \rho^*$ or the number of iterations reaches $n_s \geq n_s^*$, where ρ^* denotes the exploration threshold and n_s^* the maximum allowed number of agent–environment interactions. At each timestep, the environment provides a reward $r_t(\mathcal{M}_t^a, v_t^a, a_t^\pi, a_t^\sigma)$ that is used during training to update the policy π_θ and the critic network V_ϕ through the PPO algorithm [30]. Notably, the proposed formulation is platform-agnostic since the exploration policy operates on a graph-based representation of the environment and outputs high-level moves that do not depend on the platform-specific dynamics.

B. Agent's observation space

At time t, the agent's observation is represented as an exploration graph $o_t = (\mathcal{V}_t, \mathcal{E}_t)$ as illustrated in Fig. 2 where the node set $\mathcal{V}_t$ consists of the next-step waypoints $\mathcal{V}_t^n$ as per Eq. (2), and exploration frontiers $\mathcal{V}_t^f$. Specifically, $\mathcal{V}_t^f$ is the subset of free cells in $\mathcal{M}_t^a$ that are adjacent, under 8-connectivity, to at least one unknown cell, and represent the interface between explored and unexplored regions of the environment.

$$\mathcal{V}_t^n = \{v_t^a + a \;\mid\; a \in \mathcal{A}\} \tag{2}$$

The edge set $\mathcal{E}_t$ encodes the spatial relations among nodes, with each edge weighted by the Euclidean distance between the centroids of the corresponding cells. Specifically, it contains bidirectional edges among all the waypoint nodes $\mathcal{V}_t^n$ and unidirectional edges connecting each frontier node $f \in \mathcal{V}_t^f$ to its nearest next-step waypoint $w^* \in \mathcal{V}_t^n$, defined by Eq. (3).

$$w^* = \arg\min_{w\in\mathcal{V}_t^n} \|f - w\|_2, \tag{3}$$

TABLE I

HYPERPARAMETERS OF THE FIRST-STAGE GRAPH NEURAL NETWORK LAYERS USED IN BOTH THE POLICY AND CRITIC MODELS

GATv2Conv	Input dim	Output dim	Heads	Activation
Layer 1	8	16	4	ReLU
Layer 2	64	1	1	-

where $\|\cdot\|_2$ denotes the Euclidean distance. Each node $v_i \in \mathcal{V}_t$ is associated to a feature vector $\mathbf{x}_i = [\,\mathbf{j}_i \;\; \mathbf{h}_i\,]$, encoding both its class and local environment descriptors derived from the current agent's map. As illustrated in Fig. 2, the first four dimensions of the node feature vector $\mathbf{j}_i \in \{0,1\}^4$ employ one-hot encoding to indicate whether a node corresponds to the agent's position, a traversable or non-traversable next-step waypoint, or an exploration frontier. The remaining of the feature vector $\mathbf{h}_i \in [0,1]^4$ encodes local occupancy statistics from a $(2k+1)\times(2k+1)$ neighborhood in the agent's map $\mathcal{M}_t^a$ of the node v_i. Specifically, the normalized counts of free, unknown, occupied, and frontier cells surrounding v_i are computed according to Eq. (4).

$$\mathbf{h}_i(v_i) = \frac{1}{(2k+1)^2}\begin{bmatrix} \Gamma_{traversable}(v_i) \\ \Gamma_{unkown}(v_i) \\ \Gamma_{non-traversable}(v_i) \\ \Gamma_{frontiers}(v_i) \end{bmatrix}^\top \tag{4}$$

where Γ_{type} denotes the number of cells of a given type within a $(2k+1)\times(2k+1)$ neighborhood centered at cell v_i, defined as

$$\Gamma_{\text{type}}(v_i) \;=\; \sum_{x=v_{i,x}-k}^{v_{i,x}+k} \sum_{y=v_{i,y}-k}^{v_{i,y}+k} \mathbf{1}_{type}(x,y), \tag{5}$$

with $v_i = (v_{i,x}, v_{i,y})$ and $\mathbf{1}_{type}(x,y)$ is the indicator function.

C. Architecture of the exploration policy and critic network

Both the exploration policy π_θ, shown in Fig. 2, and the critic network V_ϕ are modeled as graph neural networks that leverage attention-based message passing. They operate on the same exploration graph o_t and adopt a two-stage architecture. The first stage, identical in structure but instantiated as independent networks, encodes local relational dependencies within the graph through attention mechanisms, with the policy and critic trained separately under disjoint parameterizations. The second stage differs between the two networks: for π_θ, it maps node embeddings to action probabilities, whereas for V_ϕ, it maps the graph-level representation to a scalar value estimate, both realized through fully connected layers.

- **First stage.** This network consists of two stacked GATv2Conv layers [31] with hyperparameters shown in Table I. The first layer projects the 8-dimensional node features of the exploration graph o_t into a 16-dimensional hidden representation $\mathbf{x}_i'$ by using four attention heads, while integrating edge attributes into the attention mechanism as per Eq. 6. Subsequently, the second layer further refines these representations using a single attention head, yielding the embedding $\mathbf{x}_i''$. Formally, the embedding of a graph node after the application of one attention head of a GATv2Conv layer is computed according to Eq. (6)

$$\mathbf{x}_i' = \sum_{j\in\mathcal{N}(i)\cup\{i\}} \alpha_{i,j}\,\mathbf{\Theta}_t\,\mathbf{x}_j, \tag{6}$$

where $\mathcal{N}(i)$ denotes the neighbors of i, $\mathbf{\Theta}_t$ is a learned projection matrix, and the attention coefficients $\alpha_{i,j}$ are obtained by applying Eq. (7).

$$\alpha_{i,j} = \frac{e^{\mathbf{a}^\top \epsilon_{i,j}}}{\sum_{k\in\mathcal{N}(i)\cup\{i\}} e^{\mathbf{a}^\top \epsilon_{i,k}}}, \tag{7}$$

with

$$\epsilon_{p,q} = \xi(\mathbf{\Theta}_s\mathbf{x}_p + \mathbf{\Theta}_t\mathbf{x}_q + \mathbf{\Theta}_e\mathbf{e}_{p,q}) \tag{8}$$

where $\mathbf{a}$ is the attention vector, $\mathbf{e}_{p,q}$ is the weight of the edge connecting nodes p and q, $\xi(\cdot)$ denotes the LeakyReLU activation function, and $\mathbf{\Theta}_s$, $\mathbf{\Theta}_t$, $\mathbf{\Theta}_e$ are learnable transformation matrices for source, target nodes, and the projection matrix mapping edge features into the attention space, respectively.

- **Second stage**. In the policy network, the output is reduced to a scalar per node, and the logits of the nodes in $\mathcal{V}_n$ define the action distribution. In the critic network, the output is 16-dimensional, averaged across the nodes in $\mathcal{V}_n$, and passed through a fully connected layer to predict the critic value.

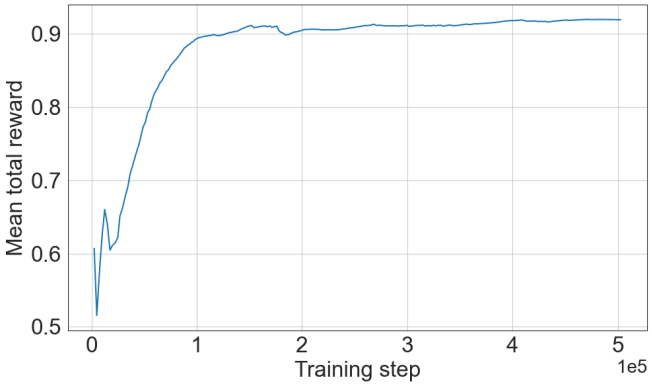


Fig. 3. Training performance curve showing the mean total reward versus training steps, with rewards normalized to $[0,1]$ and smoothed using a 1000-step simple moving average.

D. Reward formulation and ablation study

The reward function used to train the exploration policy π_θ is designed to encourage efficient exploration while penalizing interventions by the safety filter. We consider the base reward structure in Eq. (9) to formulate the reward function.

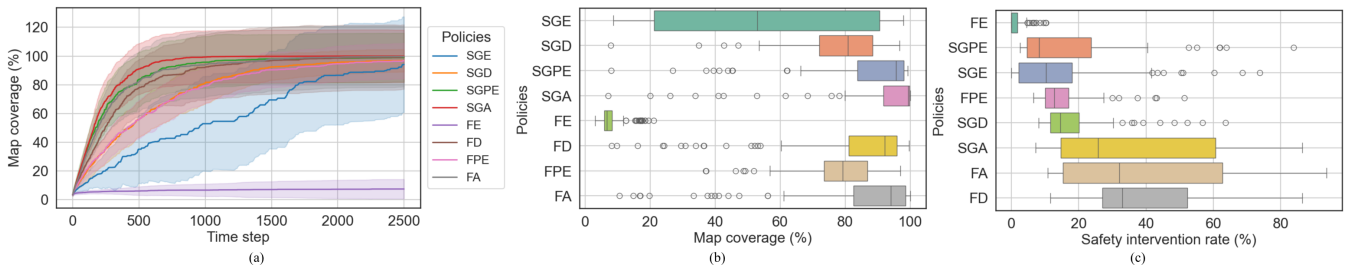


Fig. 4. Data collected from the simulation of the trained policies on 100 randomly generated Gymnasium environments. From left to right: (a) Median map coverage per time step (colored lines) with variability shown as ±1 standard deviation (shaded regions) across testing environments. (b) Distribution of agents' map coverage after 1000 steps. (c) Distribution of the proportion of safety-filter interventions over agent actions across 100 test simulations per policy.

$$r_t(\mathcal{M}_t^a, v_t^a, a_t^\pi, a_t^\sigma) = \begin{cases} r^*_{unsafe}, & \text{if } a_t^\pi \neq a_t^\sigma \\ r^*_{exp}, & \text{if } a_t^\pi = a_t^\sigma \ \wedge \ \rho \geq \rho^* \\ r_t^{step}, & \text{otherwise} \end{cases} \tag{9}$$

where r^*_{unsafe} is applied to discourage unsafe behaviors, r^*_{exp} is a reward given when the agent has explored a sufficiently high portion of the exploration arena, and r_t^{step} is a step-wise reward that measures the benefit of executing action a_t^ϕ for exploration. Based on Eq. 9, we conduct an ablation study by analyzing eight different reward functions to evaluate the contribution and effectiveness of each term in the proposed reward SGA:

- **Safety-Gated Exploration (SGE)**. This formulation penalizes unsafe actions while providing only a sparse terminal reward. Consequently, the step-wise term is set to $r_t^{step} = 0$ in Eq. 9, such that exploration is positively rewarded exclusively upon reaching the threshold ρ^*.
- **Safety-Gated Discovery (SGD)**. To explicitly incentivize discovery, this formulation assigns $r_t^{step} = n_d$, where n_d denotes the number of cells newly discovered at step t, thereby capturing the progressive expansion of the known map.
- **Safety-Gated Pure Exploration (SGPE).** This approach differs from the previous ones by positively rewarding discoveries while penalizing a lack of progress in the exploration process, with the step reward r_t^{step} defined in Eq. (10).

$$r_t^{\text{step}} = \begin{cases} r_0, & \text{if } n_d \leq 0 \\ n_d, & \text{if } n_d > 0 \end{cases} \tag{10}$$

- **Safety-Gated Adaptive (SGA)**. This reward function is the one proposed in this paper and integrates penalization of unsafe actions, discovery-driven rewards, and a potential-field-inspired term defined on frontier points. Therefore, the step-wise reward is defined according to Eq. (11).

TABLE II

PARAMETERS USED FOR ENVIRONMENT, TRAINING SETUP, AND REWARD TERMS

Environment		Training		Reward	
Name	Value	Name	Value	Name	Value
h, w	50	Rollouts	1024	r^*_{unsafe}	-1
n_o	45	Mini-Batches	64	r_0	-0.5
$r_{\min}, r_{\max}$	$[1, 3]m$	Learning Rate	3×10^{-4}	r^*_{exp}	100
l	5.0	Learning Epochs	8		
ρ^*	0.98	Discount factor	0.99		
n_s^*	2500	Timesteps	$4 \cdot 10^5$		
k	3	ϵ	0.2		
r_g	1.0 m				

$$r_t^{\text{step}} = \begin{cases} n_d, & \text{if } n_d > 0 \\ \Phi_t - \Phi_{t-1}, & \text{if } n_d \leq 0 \ \wedge \ |\mathcal{V}_t^f|, |\mathcal{V}_{t-1}^f| > 1 \\ r_0, & \text{otherwise} \end{cases} \tag{11}$$

where $\Phi_t(\mathcal{M}_t^a, v_t^a)$ is the potential field:

$$\Phi_t = \max_{f \in \mathcal{V}_t^f} \frac{\Gamma_{unknown}(f)}{\|f - v_t^a\|_2}, \tag{12}$$

with v_t^a the agent's position, $\mathcal{V}_t^f$ the set of frontiers and $\|\cdot\|_2$ is the Euclidean distance.

For each formulation, we additionally consider variants in which the unsafe penalty r^*_{unsafe} is removed. These are denoted respectively as Free Exploration (FE), Free Discovery (FD), Free Pure Exploration (FPE), and Free Adaptive (FA). As ablation study, the eight policies based on these reward formulations have been trained and evaluated over 100 simulations in randomly generated environments, comparing map coverage and safety shield interventions to assess the impact of each reward terms.

III. SIMULATION AND EXPERIMENT

This section outlines the training setup in III-A, and presents the evaluation of the trained policies both in Gymnasium-based simulations in III-B and experiments in a lab environment III-C.

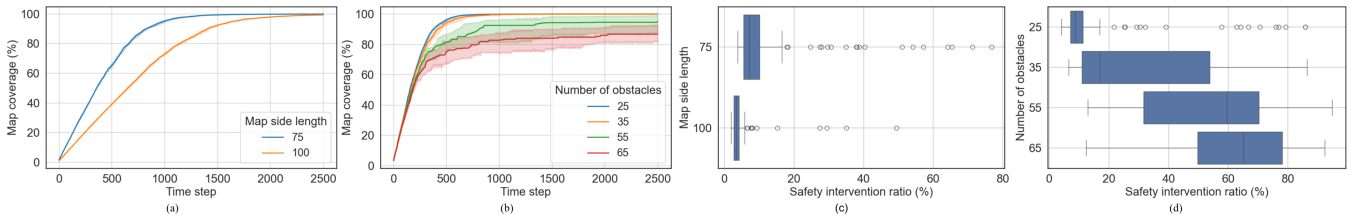


Fig. 5. From left to right: median map coverage trajectories across 100 randomly generated exploration environments with different environment sizes (a) and number of trees (b) for the proposed policy SGA. Distribution of the proportion of safety filter interventions over agent actions across 100 test simulations for the policy SGA in environments with different sizes (c) and number of trees (d).

A. *Training setup*

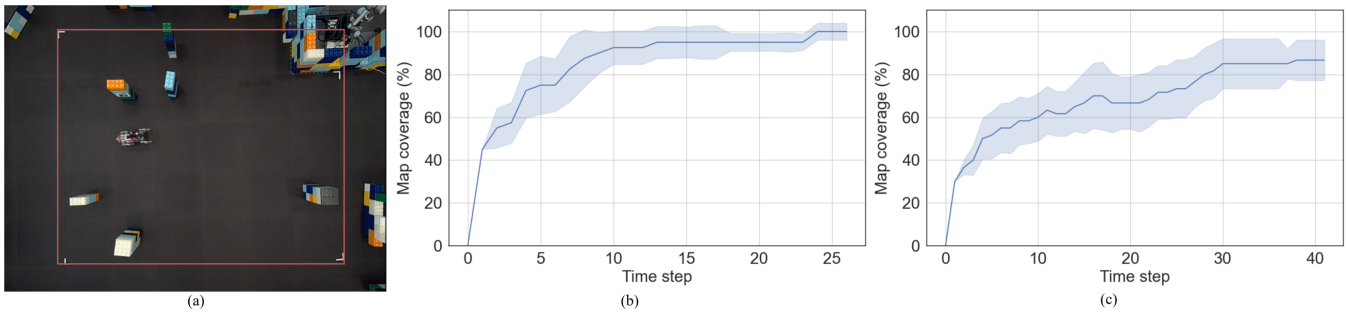
The training is conducted on HPC2N's Kebnekaise cluster running Ubuntu 20.04.6 LTS, utilizing 5 Skylake CPU cores and an NVIDIA H100 GPU, while policy evaluations are performed on the same system using 10 Skylake CPU cores. The environment is implemented using Gymnasium [32], with the main design parameters summarized in Table II. Both the exploration policy and the critic network are modeled as graph-based neural networks using PyTorch Geometric [33] and trained in environments with randomized initial agent placements using the PPO algorithm implemented by skrl [34], with hyperparameters listed in Table II. Fig. 3 depicts the evolution of the mean total reward during training of the policy trained using SGA, characterized by an overall upward trend with an initial phase of instability up to approximately 2×10^4 steps, after which the curve stabilizes and converges. The policy was trained for slightly more than 5×10^5 steps, wherein the policy converged after about 1×10^5 steps.

B. *Simulation results*

The hierarchical exploration framework with the exploration policies trained under the devised eight reward formulations discussed in II-D have been evaluated across 100 randomly-generated testing environments in Gymnasium, each executed for 2,500 time steps, in order to assess the robustness to variations in non-traversable region placement and agent initialization. Fig. 4 shows the simulation data, including the coverage of the map as a function of the exploration time and the frequency of safety-filter interventions to prevent unsafe actions. These results provide the basis for the ablation analysis of the proposed SGA reward function.

Fig. 4-(a) illustrates the median and standard deviation of map coverage as a function of time steps during exploration for each evaluated policy. The use of the median trajectory provides a robust indicator of typical performance across environments, mitigating the influence of outliers. The policy trained with the proposed SGA reward exhibits the steepest and most consistent increase in coverage, achieving 95% map exploration within approximately 605 steps in median performance. These results underscore the efficiency and robustness of the proposed framework, even under varying environment configurations. In contrast, policies trained with reward functions comprising fewer components than SGA yield exploration trajectories with a smaller slope, such as FPE, and, in some cases, substantially reduced map coverage considering corresponding steps. For example, after 1,000 steps, the SGA policy achieves over 99% coverage, whereas FE remains at only 6.7%. In particular, SGE and FE, the simplest reward formulations, yield the lowest performance, failing to exceed 60% coverage within the first 1,000 steps. Furthermore, SGE exhibits substantial variability across simulations, as indicated by the large standard deviation reported in Fig. 4-(a). Incorporating additional reward components, as in SGD and FD, yields more favorable trajectories by explicitly encouraging incremental exploration. FD attains high levels of coverage; however, its trajectory is flatter than those of other policies and converges to lower final coverage. SGPE further improves performance by penalizing non-expansive actions. Nevertheless, SGA consistently outperforms all alternatives, achieving both faster convergence and higher overall coverage. Fig. 4-(b) presents the distribution of map coverage after 1,000 steps. The SGA policy achieves the highest coverage and exhibits the most reliable performance, as indicated by its narrow interquartile range (IQR). Although FE displays an even narrower IQR, around 2.5, its overall coverage remains extremely limited. All other policies yield inferior results, characterized by both lower median coverage and greater variability across simulations. In particular, SGE shows highly inconsistent behavior, with an IQR of 70. Fig. 4(c) shows the proportion of agent steps in which the safety filter intervenes to prevent collisions. Policies trained with reward functions that explicitly penalize unsafe actions (*e.g.*, SGPE, SGE, SGD, and SGA) generally exhibit fewer interventions, as reflected by lower median filter activations, compared to formulations without such penalization. This indicates that these policies not only achieve efficient exploration but also learn to operate with reduced reliance on the safety filter. Nonetheless, the occurrence of occasional activations highlights the filter's continued relevance, providing robust safety guarantees in critical scenarios.

To further assess robustness, the proposed SGA policy was evaluated in environments with varying numbers of obstacles and different map sizes. For each configuration, 100 simulations of 2,500 steps were conducted. Fig. 5 illustrates the policy's adaptability to both conditions. In particular, Fig. 5-(a) shows that the agent successfully explores more than 90% of the traversable area within 831 steps in maps 125% larger than the nominal size, and within 1,405 steps in maps 300% larger. As shown in Fig. 5-(b), increasing the proportion



(a) (b) (c)

Fig. 6. From left to right: (a) Custom-designed exploration arena, delineated in red, constructed in a laboratory environment, with blocks defining non-traversable regions and the Unitree Go1 quadruped robot acting as the exploring agent. Median map coverage obtained by the Unitree Go1 across 10 environments with randomly placed obstacles in arenas of size $4\,\mathrm{m} \times 5\,\mathrm{m}$ (b) and 10 arenas of size $6\,\mathrm{m} \times 5\,\mathrm{m}$ (c).

of non-traversable regions slightly reduces coverage at a fixed number of steps; nevertheless, the policy maintains over 80% coverage even with a 45% increase in obstacles. The impact of obstacle density is further reflected in Fig. 5-(d), where safety-filter activations grow with increasing obstacle counts, underscoring the importance of the filter in ensuring safe exploration. Conversely, Fig. 5-(c) shows that the frequency of safety interventions decreases with larger environment sizes. Eventually, all results confirm the scalability and robustness of the proposed framework across diverse exploration scenarios.

C. Experimental results

To evaluate the trained policy in real environment, the proposed framework was implemented in ROS 2 (Humble Hawksbill) and deployed on a Unitree Go1 quadruped robot. The experiments have been carried out in multiple custom-designed exploration arenas with varying sizes, obstacle configurations, and obstacle densities, all constructed within a laboratory environment, as illustrated in Fig. 6-(a). The figure shows one configuration of the arena, outlined in red, with dimensions of $6\,\mathrm{m} \times 5\,\mathrm{m}$. The Unitree Go1 quadruped robot is shown alongside several randomly positioned LEGO blocks, which define non-traversable regions within the arena. The Unitree Go1 robot is equipped with an onboard Intel NUC computer and a SLAMTEC RPLIDAR S1 360° Laser Scanner. Ground-truth position and orientation are obtained using a Vicon motion capture system. The implemented framework assigns to the agent a ROS 2 node that is responsible for: (i) collecting laser scans from the LiDAR and odometry measurements from the Vicon system when the robot reaches an exploration waypoint; (ii) computing the exploration graph o_t; (iii) inferring the action a_t^π by applying the trained policy π_θ on the exploration graph o_t; (iv) applying the safety filter that accounts for the reconstructed map and nearby non-traversable regions, thus constraining the policy action a_t^π to yield a feasible action a_t^σ; and (v) generating control commands to drive the quadruped toward the next waypoint. For these experiments, the LiDAR range was restricted to $2.12\,\mathrm{m}$, so that at each time step the robot could only assess the traversability of its neighboring cells. Moreover, each action selected by the proposed hierarchical exploration framework is executed by a navigation controller, which guides the robot to the center of the neighboring cell corresponding to the chosen action. In the case of diagonal movements, the robot sequentially traverses two adjacent cells to reach the final goal, with the constraint that such movement is permitted only if at least one of the two intermediate cells is traversable. Fig. 6-(b) reports the map coverage across 10 experiments conducted in arenas of size $4\,\mathrm{m} \times 5\,\mathrm{m}$ with randomly placed obstacles. The results indicate that after 7 time steps the agent achieved more than 80% coverage, and after 13 time steps it exceeded 95%. The framework is further evaluated in larger arenas, as illustrated in Fig. 6-(c). Specifically, 10 experiments are conducted in the larger arena of size $6\,\mathrm{m} \times 5\,\mathrm{m}$. In this setting, the map coverage reached 60% after 10 steps and 80% after 28 steps. Overall, the experimental results demonstrate that the proposed framework successfully generalizes to environments that differ from those encountered during training, thereby confirming the robustness of the SGA policy.

IV. CONCLUSIONS

This paper investigates a novel platform-agnostic hierarchical framework for safe reinforcement learning-based exploration in obstacle-rich cluttered environments. The method combines an exploration-oriented policy implemented as a graph neural network policy, with attention-based message passing, and a safety filter that guarantees the feasibility of the chosen exploration actions. Furthermore, this article proposes a novel graph representation of the environment with exploration-relevant locations and a frontier-inspired reward design that promote movement towards high-gain frontiers. Eventually, simulation studies across 100 randomly-generated testing environments in Gymnasium and experiments in a lab environment demonstrate the devised agent's efficacy to achieve robust map coverage across diverse environments, with consistent performance under varying obstacle densities and scales. Future work will focus on deploying the framework as a high-level exploration planner in a real-world experiment.

REFERENCES

[1] S. Amin, M. Gomrokchi, H. Satija, H. Van Hoof, and D. Precup, "A survey of exploration methods in reinforcement learning," *arXiv preprint arXiv:2109.00157*, 2021.

[2] L. C. Garaffa, M. Basso, A. A. Konzen, and E. P. de Freitas, "Reinforcement learning for mobile robotics exploration: A survey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 8, pp. 3796–3810, 2021.

[3] M. Hutsebaut-Buysse, K. Mets, and S. Latré, "Hierarchical reinforcement learning: A survey and open research challenges," *Machine Learning and Knowledge Extraction*, vol. 4, no. 1, pp. 172–221, 2022.

[4] B. Singh, R. Kumar, and V. P. Singh, "Reinforcement learning in robotic applications: a comprehensive survey," *Artificial Intelligence Review*, vol. 55, no. 2, pp. 945–990, 2022.

[5] S. Pateria, B. Subagdja, A.-h. Tan, and C. Quek, "Hierarchical reinforcement learning: A comprehensive survey," *ACM Computing Surveys (CSUR)*, vol. 54, no. 5, pp. 1–35, 2021.

[6] P. Ladosz, L. Weng, M. Kim, and H. Oh, "Exploration in deep reinforcement learning: A survey," *Information Fusion*, vol. 85, pp. 1–22, 2022.

[7] Y. Xue and W. Chen, "A uav navigation approach based on deep reinforcement learning in large cluttered 3d environments," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 3, pp. 3001–3014, 2022.

[8] R. Cimurs, I. H. Suh, and J. H. Lee, "Goal-driven autonomous exploration through deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 730–737, 2021.

[9] Y. Cao, T. Hou, Y. Wang, X. Yi, and G. Sartoretti, "Ariadne: A reinforcement learning approach using attention-based deep networks for exploration," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 10 219–10 225.

[10] C. Sun, J. Orbik, C. M. Devin, B. H. Yang, A. Gupta, G. Berseth, and S. Levine, "Fully autonomous real-world reinforcement learning with applications to mobile manipulation," in *Conference on Robot Learning*. PMLR, 2022, pp. 308–319.

[11] Q. Liu, X. Li, Y. Tang, X. Gao, F. Yang, and Z. Li, "Graph reinforcement learning-based decision-making technology for connected and autonomous vehicles: Framework, review, and future trends," *Sensors*, vol. 23, no. 19, p. 8229, 2023.

[12] S. Munikoti, D. Agarwal, L. Das, M. Halappanavar, and B. Natarajan, "Challenges and opportunities in deep reinforcement learning with graph neural networks: A comprehensive review of algorithms and applications," *IEEE transactions on neural networks and learning systems*, 2023.

[13] M. Nie, D. Chen, and D. Wang, "Reinforcement learning on graphs: A survey," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 7, no. 4, pp. 1065–1082, 2023.

[14] Z. Zhang, C. Shi, P. Zhu, Z. Zeng, and H. Zhang, "Autonomous exploration of mobile robots via deep reinforcement learning based on spatiotemporal information on graph," *Applied Sciences*, vol. 11, no. 18, p. 8299, 2021.

[15] E. P. Herrera-Alarcón, G. Baris, M. Satler, C. A. Avizzano, and G. Loianno, "Learning heuristics for efficient environment exploration using graph neural networks," in *2023 21st International Conference on Advanced Robotics (ICAR)*. IEEE, 2023, pp. 86–93.

[16] F. Yang, X. Li, Q. Liu, Z. Li, and X. Gao, "Generalized single-vehicle-based graph reinforcement learning for decision-making in autonomous driving," *Sensors*, vol. 22, no. 13, p. 4935, 2022.

[17] Y. Lu, Y. Chen, D. Zhao, and D. Li, "Mgrl: Graph neural network based inference in a markov network with reinforcement learning for visual navigation," *Neurocomputing*, vol. 421, pp. 140–150, 2021.

[18] S. Gu, L. Yang, Y. Du, G. Chen, F. Walter, J. Wang, and A. Knoll, "A review of safe reinforcement learning: Methods, theories and applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[19] L. Brunke, M. Greeff, A. W. Hall, Z. Yuan, S. Zhou, J. Panerati, and A. P. Schoellig, "Safe learning in robotics: From learning-based control to safe reinforcement learning," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 5, no. 1, pp. 411–444, 2022.

[20] W. Zhao, T. He, R. Chen, T. Wei, and C. Liu, "State-wise safe reinforcement learning: A survey," *arXiv preprint arXiv:2302.03122*, 2023.

[21] M. Xu, Z. Liu, P. Huang, W. Ding, Z. Cen, B. Li, and D. Zhao, "Trustworthy reinforcement learning against intrinsic vulnerabilities: Robustness, safety, and generalizability," *arXiv preprint arXiv:2209.08025*, 2022.

[22] M. Guerrier, H. Fouad, and G. Beltrame, "Learning control barrier functions and their application in reinforcement learning: A survey," *arXiv preprint arXiv:2404.16879*, 2024.

[23] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *International conference on machine learning*. PMLR, 2017, pp. 22–31.

[24] L. Yang, J. Ji, J. Dai, Y. Zhang, P. Li, and G. Pan, "Cup: A conservative update policy algorithm for safe reinforcement learning," *arXiv preprint arXiv:2202.07565*, 2022.

[25] W. Huang, J. Ji, C. Xia, B. Zhang, and Y. Yang, "Safedreamer: Safe reinforcement learning with world models," *arXiv preprint arXiv:2307.07176*, 2023.

[26] S. Feng, H. Sun, X. Yan, H. Zhu, Z. Zou, S. Shen, and H. X. Liu, "Dense reinforcement learning for safety validation of autonomous vehicles," *Nature*, vol. 615, no. 7953, pp. 620–627, 2023.

[27] J. Dai, J. Ji, L. Yang, Q. Zheng, and G. Pan, "Augmented proximal policy optimization for safe reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 6, 2023, pp. 7288–7295.

[28] L. Zhang, Q. Zhang, L. Shen, B. Yuan, X. Wang, and D. Tao, "Evaluating model-free reinforcement learning toward safety-critical tasks," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 37, no. 12, 2023, pp. 15 313–15 321.

[29] Y. J. Ma, A. Shen, O. Bastani, and J. Dinesh, "Conservative and adaptive penalty for model-based safe reinforcement learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 36, no. 5, 2022, pp. 5404–5412.

[30] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017. [Online]. Available: https://arxiv.org/abs/1707.06347

[31] S. Brody, U. Alon, and E. Yahav, "How attentive are graph attention networks?" 2022. [Online]. Available: https://arxiv.org/abs/2105.14491

[32] M. Towers, A. Kwiatkowski, J. Terry, J. U. Balis, G. De Cola, T. Deleu, M. Goulão, A. Kallinteris, M. Krimmel, A. KG *et al.*, "Gymnasium: A standard interface for reinforcement learning environments," *arXiv preprint arXiv:2407.17032*, 2024.

[33] M. Fey and J. E. Lenssen, "Fast graph representation learning with PyTorch Geometric," in *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[34] A. Serrano-Muñoz, D. Chrysostomou, S. Bøgh, and N. Arana-Arexolaleiba, "skrl: Modular and flexible library for reinforcement learning," *Journal of Machine Learning Research*, vol. 24, no. 254, pp. 1–9, 2023.