Neuro-Explorer: Efficient and Scalable Exploration Planning via Learned Frontier Regions

Kyung Min Han and Young J. Kim

Abstract— **We present an efficient and scalable learning-based autonomous exploration system for mobile robots navigating unknown indoor environments. Our system incorporates three network models trained to identify the frontier region (FR), to evaluate the detected FR regions based on their proximity to the robot (A*-Net), and to measure the coverage reward at the FR regions (Viz-Net). Our method employs an active window of the map that moves along with the robot, offering scalable exploration capabilities while maintaining a high rate of exploration coverage owing to the two exploratory measures utilized by A*-Net (proximity) and Viz-Net (coverage). Consequently, Our system completes over 99% coverage in a large-scale benchmarking world, scaling up to $135m \times 80m$. In contrast, other state-of-the-art approaches completed only less than 40% of the same world with a 30% slower exploration speed than ours.**

I. Introduction

Autonomous indoor exploration is one of the fundamental mobile robotic tasks wherein the agent needs to utilize the incoming map configuration to avoid revisiting the areas already covered while locating and exploring uncharted space. This problem has renewed importance due to the increasing need for contact-free mobility services in daily life.

One of the popular approaches in indoor exploration problems is based on detecting *frontier regions* (FR) in the occupancy grid map [1], followed by a greedy exploration strategy. While we acknowledge the past success of the traditional greedy approaches, our main research question centers on improving the potential of frontier-based exploration by utilizing machine learning (ML)-based approaches.

With this motivation in mind, this paper proposes a new exploration pipeline based on three neural network models. Given a grid map configuration, each network model is designed to play a distinct role in locating the best target point to visit. The first network, *FR-Net*, locates FR cells in the input grid map, which imitates the behavior of the analytical algorithms for frontier region detection [2], [3]. The detected FR cells are a candidate for determining the final target cell to visit. Afterward, we proceed to select the optimal target cell from the FR cells based on two criteria:

1) **Proximity:** How close is the frontier cell w.r.t. the current agent's position?
2) **Coverage:** How much would the new region be explored from the frontier cell when it is to be visited?

The first criterion resembles the typical greedy search strategy most exploration planners utilize. The second criterion quantifies the visibility of the FR cells when they are visited. We propose two additional networks, *A*-Net* and *Viz-Net*, to find the best solution satisfying these criteria.

The authors are with the Department of Computer Science and Engineering at Ewha Womans University in Korea $\{hankm|kimy\}@ewha.ac.kr$

A*-Net is designed to learn and predict the A*-like distance measure [4] for the FR grid cells to minimize the agent's travel distance. The A*-Net is able to estimate the distances to the grid cells belong to numerous frontier regions in the incoming map image, consisting of over a few hundred points.

The closest cell, however, does not necessarily yield a good spot to expose the unexplored region. There might be a better opportunity to reveal vast unexplored regions to increase the exploration performance, costing a little more traveling distance. To address this issue, we define a visibility metric that measures how many unexplored cells would be visible from an FR cell, and propose Viz-Net, which predicts the visibility score of the FR cells. Subsequently, we select the best frontier point by ensembling both predictions from A*-Net and Viz-Net.

Finally, the three networks, FR-Net, A*-Net, and Viz-Net, work only on the locally active map around the robot's position, enabling scalable exploration of environments of varying sizes. Our system can successfully explore a large-scale simulation environment, such as a $135m \times 80m$-sized virtual world, in real time.

In summary, the main contributions of this paper are as follows:

- We propose a novel learning-based pipeline for autonomous mobile robot exploration tasks in unknown indoor environments.
- We propose a novel deep learning-based frontier region detector, FR-Net, that improves existing conventional analytical frontier detectors, such as FFP algorithm [3], with enhanced detection robustness supervised by a novelc FR detector, namely FFP^{+}.
- When the FR-Net is used in conjunction with the two additional networks, A*-Net and Viz-Net, one further can *exploit* the greedy closest distance strategy or *explore* the visibility strategy that selects a balanced goal between the two strategies to guide the robot.
- Our method based on an active window offers a scalable solution capable of robustly exploring environments of various scales, ranging from small ($20m \times 20m$) to very large ($135m \times 80$) indoor environments.
- Comparing against state-of-the-art approaches, in our experiments, our method achieved up to 70% more coverage with 40% faster exploration time when exploring the large-scale indoor environment.

II. Related Work

One of the well-known approaches in mobile robot exploration is locating the *frontier region*, a concept first introduced by Yamauchi [1]. This approach has since been developed to enhance its robustness and efficiency, as several studies show. For example, WFD [2] employs two runs of breadth-first-search (BFS) schemes to ensure a thorough FR search, while [5] focuses on limiting the frontier search space to the local region to expedite the process.

Information-theoretic approaches [6][7] conceptualize exploration as minimizing uncertainty, specifically by reducing entropy. It accomplishes entropy minimization by maximizing mutual information (MI) between the map and future observations, identifying the most meaningful action over the underlying action space. This approach is nonmyopic since the formulation of MI uses the entire map. However, the approach is computationally demanding due to the extensive action spaces of agents. In recent years, several algorithms have been proposed by researchers to overcome this challenge [8][9].

Topological approaches [10][11] represent navigation environments in a simplified graph form consisting of nodes and edges. Each node contains specific location information, such as the spatial coordinates of the environment, and each edge represents traversability among the nodes. Therefore, topological approaches provide an efficient solution. However, this simplification process can result in losing map details (i.e., incomplete reconstructed maps) compared to metric-based approaches.

Recently, learning-based exploration methods have drawn significant attention. For instance, one approach [12] learned to predict unknown areas given a partially reconstructed map to generate exploration plans. Another method [13] utilized a reinforcement learning (RL) network to train an agent that explores an environment to minimize its trajectory for coverage. Another study [14] employed an RL framework to navigate among human crowds while relying on the TSP algorithm to generate coverage plans.

III. Preliminaries

A. Problem Formulation

A 2D grid map $\mathcal{M}$ consists of three types of cell classes: (1) occupied cells $\mathcal{M}_o$ corresponding to obstacles, (2) unoccupied or free cells $\mathcal{M}_f$ corresponding to the space where a robot can freely navigate, and (3) unknown or unexplored cells $\mathcal{M}_u$. Therefore, the occupancy $\mathcal{M}(x)$, a cell x is

$$\mathcal{M}(x) = \begin{cases} \textit{OCCUPIED} & \text{if } x \in \mathcal{M}_o \\ \textit{UNKNOWN} & \text{if } x \in \mathcal{M}_u \\ \textit{FREE} & \text{if } x \in \mathcal{M}_f. \end{cases} \tag{1}$$

A set of unknown cells neighboring free cells is defined as a frontier region $\mathcal{F} := \partial\mathcal{M}_u$; see Figure 1. c_i is the geometric centroid of ith frontier region $\mathcal{F}_i$. Subsequently, ith frontier point f_i is the closest point to c_i among all points belonging to $\mathcal{F}_i$. Formally,

$$f_i = \{x \mid \underset{x}{\operatorname{argmin}} ||\mathrm{c}_i - x|| \text{ and } x \in \mathcal{F}_i\}. \tag{2}$$

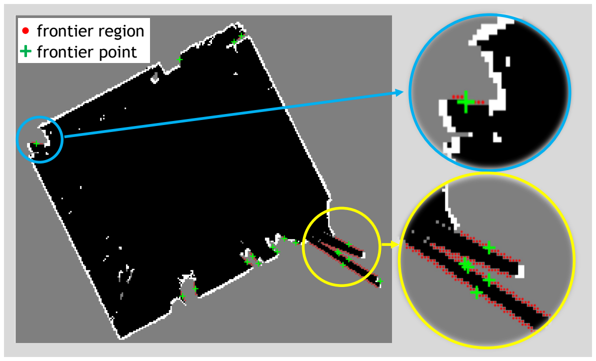


Fig. 1: Black, gray, and white pixels represent free, unknown, and occupied map cells, respectively. Red points and green crosses are superimposed on the map to mark frontier regions and their frontier points.

f_i is considered to be covered iff $||f_i - \mathbf{r}|| \leq \eta$ and $f_i \notin \mathcal{M}_u$ where $\mathbf{r} \in \mathbb{R}^2$ is the robot's position, and η is a sensor-range dependent constant.

f^* refers to the best frontier point selected by a criterion function $\mathcal{C}(\cdot)$ among all frontier candidate points $\{\forall f_i\}$, and f^* is set to a goal point for global path planning. The goal of the exploration task is to visit all reachable unknown regions until the navigating space is fully covered or, equivalently, no more frontier points are left to probe. Thus, locating f^* is essential to efficiently guide the robot to complete the exploration task.

B. System Overview

Algorithm 1 outlines the essential steps explained in this work to locate the best frontier point. Our system begins with setting an active window $\mathcal{S}$ of the map $\mathcal{M}$ placed around the robot's position $\mathbf{r}$, which is subsequently transferred to FR-Net $\mathbb{B}_\psi$ (Line 3). The FR-Net classifies the input grid-map cells into FR $\mathcal{F}$ and non-FR using the FR-Net. Similar to [3], we cluster FR cells based on their connectivity, followed by locating the points closest to the centroids of the FR clusters, which are considered frontier points f_i (Line 5). Then, we evaluate the plausibility of the detected frontier points using A*-Net $\mathbb{A}_\theta$ (Line 6) and Viz-Net $\mathbb{V}_\phi$ (Line 7) in terms of their closeness with respect to the robot's position and the expected coverage (*i.e.,* visibility) amount at the points. Subsequently, the one marked as the maximum of ensembled scores predicted by A*-Net and Viz-Net corresponds to the optimal frontier point with the highest plausibility (Line 8).

This frontier-point detection procedure is integrated with conventional navigation SW components to continuously explore unknown environments until no more frontier points are detected. In the following sections, we elaborate on the individual network models shown in the procedure and how they are composed in our system.

IV. Dataset Generation for Supervising the Network Models

Generating faithful training data sets is important to correctly supervise our three network models, FR-Net, A*-Net, and Viz-Net. This section explains how we improve fast front

Algorithm 1: Neuro-Explorer frontier point detection

```
1  Function
     FindBestFrontierPoint(input map $\mathcal{M}$, robot pos $\mathbf{r}$):
       /* Setting the Active Map Window
           */
2      $\mathcal{S} \leftarrow$ SetActiveMap($\mathcal{M}, \mathbf{r}$)
       /* FR inference                          */
3      $\mathcal{F} \leftarrow \mathbb{B}_\psi(\mathcal{S})$
4      for each $\mathcal{F}_i$ in $\mathcal{F}$ do
           /* Locate frontier points            */
5          Compute $f_i$ using Eq. (2)
           /* Inv A* distance prediction
               */
6          $\tilde{\mathrm{F}}_\mathrm{i} \leftarrow \mathbb{A}_\theta(f_i)$
           /* Visibility prediction             */
7          $\mathrm{V}_i \leftarrow \mathbb{V}_\phi(f_i)$
       /* Best frontier point decision */
8      $f^* = \underset{i}{\mathrm{argmax}}\,(\lambda\tilde{\mathrm{F}}_\mathrm{i} + (1-\lambda)\mathrm{V}_i)$ for $\exists\lambda \in [0,1]$
9      return $f^*$
```

propagation (FFP)[3] to train the FR-Net along with a new dual FFP, namely FFP$^+$. We also elaborate on the essential meta-data set used for training FR-Net, A*-Net, and Viz-Net.

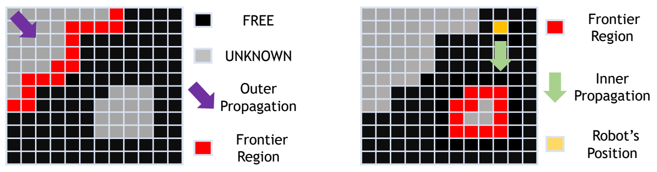
A. *Dual Fast Front Propagation*

Ensuring the reliability of the FR-Net is crucial because it is the most fundamental model in our system as it locates frontier regions to proceed. FFP propagates the front inward from the outside of $\mathcal{M}_u$ to locate the frontier region $\mathcal{F} = \partial\mathcal{M}_u$. When exploring large-scale environments, however, FFP may not make reaching isolated frontier regions possible. To correctly locate these isolated regions, we launch the dual FFP (DFFP) after the FFP ends, which propagates the frontier inside the explored map by reversing the role of free and unknown cells. In other words, our FFP$^+$ employs both FFP and DFFP to correctly locate the frontier region.

Unlike FFP starting from an unknown cell, DFFP needs to start propagation from a free cell, exploiting the robot's current position as a seed point, which is guaranteed to be a free cell. Because DFFP reverses the role of free and unknown cells and propagates over free cells, it results in the inner frontier cells $\tilde{\mathcal{F}}$ that bound the frontier cells $\mathcal{F}$. Since the inner frontier cells $\tilde{\mathcal{F}}$ are not the actual frontier region (see III-A for the definition of the frontier region), DFFP locates the actual frontier regions by searching the neighboring cells to identify the valid frontier region. Figure 2 visually illustrates the difference between FFP and DFFP. Lastly, FFP$^+$ merges the two frontier region sets found by FFP and DFFP.

B. *Generating Meta-Datasets*

Our network models follow a supervised learning regime, which requires a good training dataset. To address this need, we utilized the Explore-Bench [15] simulator and datasets



(a) Outer FR found by FFP (b) Inner FR found by DFFP

Fig. 2: (a) Isolated unknown cells (bottom right) that are unreachable when propagated from the outside using FFP. (b) DFFP propagates from the robot's position (free cell) to identify the inner frontier region that FFP might overlook.

to generate meta-datasets comprising various information. Figure 3 showcases the instances of meta-dataset when the robot is positioned at the bottom right corner of the world (see Figure 3(a) and 3(b)); specifically, Figure 3(c) and 3(f) are utilized for training FR-Net. Figure 3(d), 3(e), and 3(g) are meta-datasets for A*-Net, and Figure 3(c), 3(f) and 3(h) are for VizNet. The methods to generate these datasets will be explained in more detail in Section V-B and V-C.

We repeated 100 times of autonomous exploration runs using Autoexplorer [3] combined with our FFP+ as the FR detector to collect the meta-dataset. The exploration runs were equally executed in the six different simulation worlds available in Explore-Bench. Each run consists of approximately 22 times of exploration plans before completing exploring a world, generating the same number of meta-data sequences. Therefore, the training dataset includes about $13K$ suites of meta-data to train and validate our ANN models. As demonstrated in Sec. VI-B, even though our dataset is generated from small-scale world environments, the trained network model exhibits generalization to much larger-scale environments, effectively capturing important patterns within the dataset.

V. Learning for Frontier Point Detection

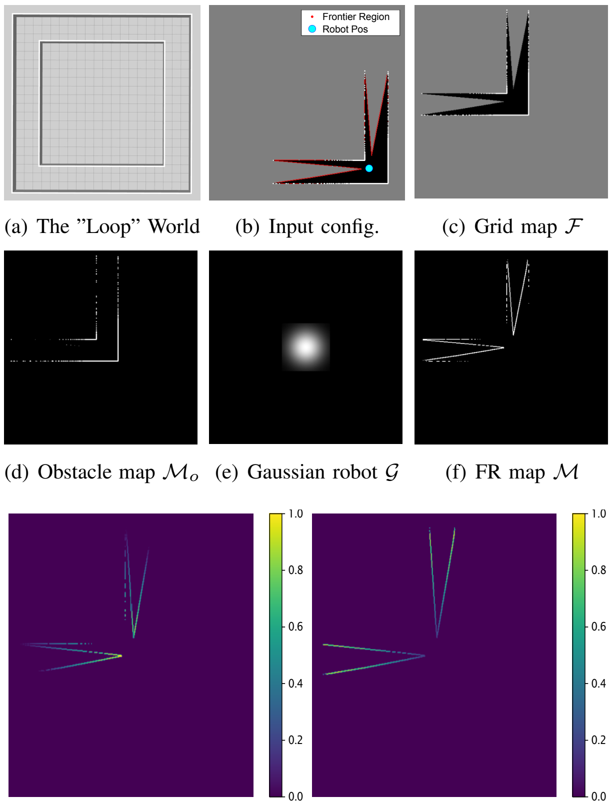
A. *FR-Net*

The main goal of FR-Net is to classify the input grid-map cells into FR or non-FR cells. To model this pixel-wise binary classification, we employed U-Net architecture [16] in which single channel output with sigmoid activation is utilized. In the training mode, the model is supervised by the FR data obtained from the method described in Section IV-A and IV-B, optimizing the binary cross entropy loss.

B. *A*-Net*

A* algorithm [4] is an optimal graph-search method to find a minimal costing path between the start and the goal node. A* traverses the graph by evaluating a cost function $\mathrm{F}(\mathrm{n})$ consisting of the path length measured from the start node to n and a lower-bound distance from n to the goal node based on a heuristic function.

Similarly, using the grid map, our A*-Net learns and predicts the cost value to the goal node from a starting node n. More specifically, A*-Net $\mathbb{A}_\theta$ learns the inverted estimate of a normalized cost $\mathrm{F}(\cdot)$, $\tilde{\mathrm{F}}(\cdot) = 1 - \mathrm{F}(\cdot)$. To train this network, using a 2D grid map $\mathcal{M}$, we generated mass tuple



(a) The "Loop" World (b) Input config. (c) Grid map $\mathcal{F}$

(d) Obstacle map $\mathcal{M}_o$ (e) Gaussian robot $\mathcal{G}$ (f) FR map $\mathcal{M}$

(g) Inverted A* map $\tilde{\mathrm{F}}$ (h) Visibility map V

Fig. 3: (a) Top view of a training world. (b) an input grid map $\mathcal{M}$ with FR in red when the robot in cyan is positioned at the bottom right of this world. The color bar in (g) and (h) indicates the normalized metrics.

sets $D_a = \{\mathcal{F}, \mathcal{M}_o, \mathcal{G}, \tilde{\mathrm{F}}(\mathcal{F})\}$ with the frontier region $\mathcal{F}$, obstacle map $\mathcal{M}_o$, $\mathcal{G}$ of 2D Gaussian distribution centered around the robot's position, and the inverted normalized cost function $\tilde{\mathrm{F}}$ computed for $\mathcal{F}$ using A*, as exemplified by Figure 3. Note that D_a (also D_v in the next section) does not explicitly contain the robot's current position. The position information is provided by ensuring that the robot is always centered in the map $\mathcal{M}$. This *implicit* encoding offers a couple of important benefits:

1) The implicit position encoding reduces the uncertainty of the networks by eliminating the time-varying robot position information from the input data of the network.
2) Constructing the active map, explained in Section V-E, can be simplified and consistent by defining it centered around the robot's position.

A*-Net is also based on U-Net architecture with tanh activation functions at the last output layer to enforce continuous, normalized output values [17].

C. Viz-Net

Many conventional frontier-based exploration methods rely on a greedy exploration strategy where the next best place to visit is the closest point w.r.t the current robot position. However, the closest point does not necessarily maximize exploration coverage when the point is visited. Our exploration system using Viz-Net also strives for maximal coverage when a frontier point is considered to be visited. Specifically, our Viz-Net model $\mathbb{V}_\phi$ is designed to predict the visibility (coverage reward) for each point belonging to $\mathcal{F}$.

To train this model, we generated mass tuple sets $D_v = \{\mathcal{M}, \mathcal{F}, \mathrm{V}(\mathcal{F})\}$, where $\mathrm{V}(\cdot)$ is the number of covered cells (either FREE or OCCUPIED) visible from the cells belonging to $\mathcal{F}$. In our implementation of $\mathrm{V}(\cdot)$, we uniformly sample the angular space of S^1 at 3.6 degrees and check for the visibility along each sampled direction using ray-shooting. Figure 3(h) presents an instance of visibility measured for the grid map corresponding to Figure 3(b). Similar to the A*-Net, Viz-Net employs U-Net architecture with tanh activation output to enforce continuous output.

D. Optimal Frontier Decision

We ensemble the two score measures by weight-averaging the results from Viz-Net and A*-Net to estimate the optimal frontier point. That is :

$$f^* = \underset{i}{\operatorname{argmax}} \, (\lambda \tilde{\mathrm{F}}_\mathrm{i} + (1-\lambda)\mathrm{V}_i), \quad \exists \lambda \in [0,1] \qquad (3)$$

where λ can be interpreted as a relative strength of the exploitation strategy to the exploration strategy, *e.g.,* $\lambda = 1$ makes the system behave as a greedy exploration.

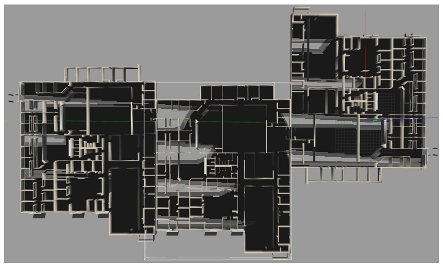
E. Size-Agnostic Exploration

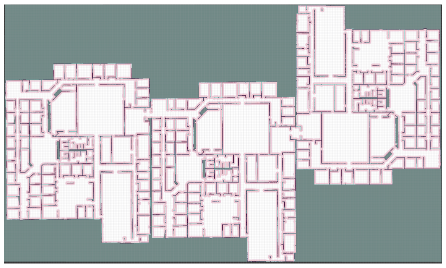
It is often the case that the grid map size increases over time as the robot explores more space during an exploration task. However, since the input and output sizes of our network model are fixed, the size of the input grid map and the output predictions must be fixed accordingly. If the size of the exploring space is known as apriori, we may adjust the network size. However, such information is unavailable when exploring an unknown environment.

To address this issue, we propose a dynamic window-based strategy that employs a fixed-size bounding box, which moves an active map (AM) along with the robot, where the AM continuously feeds itself as the input to the FR-Net. Therefore, our FR-Net applied to the active map only to locate *local* frontier points, and they are sequentially registered in *global* frontier point list. Moreover, our A*-Net and Viz-Net are applied to only the active map to determine the best frontier point out of the *local* frontier points. If *local* frontier point is unavailable because the currently observed AM is fully covered, we choose the next target from the *global* frontier point list. Note that *global* frontier point list accumulated over time as the AM moves around the exploring space. In our implementation, we use the 1024×1024 active map, which is down-sampled once to 512×512 before passing it to FR-Net.

VI. Experimental Results and Analysis

In this work, we utilized a workstation with an AMD Threadripper processor running at 4.2GHz and a Nvidia RTX 3090 GPU. Leveraging this computing setup, we successfully incorporated our learning-based exploration models into the ROS navigation system to build a fully integrated mobile


(a) WGx3 world


(b) Fully explored map with exact localization

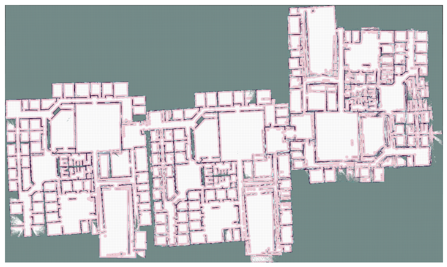

(c) Fully explored map with SLAM

Fig. 4: Benchmarking world and explored map results of Neuro-Explorer with exact and estimated localization.

navigation SW system [18], including global and local planners. We deployed Turtlebot3 of $0.19m \times 0.18m$ size with the OctoMap [19], building an occupancy grid map based on the ground-truth localization. For a fair comparison, we used the TEB local planner [20], the same planner used in the baseline systems.

As testbeds for benchmarking our system, we created two large-scale simulated worlds independent of the training environments by replicating the Willow-Garage (WGx1) world [21] two or three times, denoted as WGx2 and WGx3, respectively, throughout this paper. A top-view image of WGx3 can be found in Fig. 4(a), and its dimension is $135m \times 80m$. Subsequently, we carried out extensive experiments in WGx1∼WGx3 to evaluate the performance of our Neuro-Explorer system. To this end, we posed the following four important questions:

- Q1 **Exploitation vs Exploration:** How should we combine A*-Net and Viz-Net to obtain the best planning performance?
- Q2 **Performance:** How well does Neuro-Explorer perform compared to conventional greedy exploration methods?
- Q3 **Scalability:** How scalable is Neuro-Explorer for a long-term exploration task?
- Q4 **Localization insensitivity:** How insensitive is Neuro-explorer to the planner's localization accuracy?

A. Lambda Analysis

This section explores Q1. We carried out numerous exploration runs to find the optimal λ value between 0 and 1. We conducted five exploration runs for each λ and measured the time spent for each run. All the exploration runs were repeated in WGx1. Table I presents the median results of the five exploration runs. The metric T_p [15] indicates the exploration time spent completing $p\%$ of the map.

$T_p(10^3 s)$	T_{50}	T_{80}	T_{90}	T_{99}
$\lambda = 0.0$	**0.41**	1.04	**1.41**	2.27
$\lambda = 0.2$	0.54	**0.98**	1.6	**2.17**
$\lambda = 0.4$	**0.41**	1.04	1.59	2.20
$\lambda = 0.6$	0.65	1.35	2.06	2.41
$\lambda = 0.8$	0.69	1.19	2.14	2.34
$\lambda = 1.0$	0.84	1.64	2.10	2.34

TABLE I: λ analysis

According to this study, reducing λ is certainly beneficial for achieving up to 90% of the total map. Therefore, λ should be adjusted based on the characteristics of exploration tasks. For example, if the objective of the exploration task is to rapidly cover 90% of the environment, even VizNet alone is adequate. However, if the task strictly requires achieving over 99% coverage, A*-Net should also be considered. Overall, the best exploration performance was obtained when $\lambda = 0.2 \sim 0.4$ by emphasizing the exploration strategy (Viz-Net) over the exploitation strategy (A*-Net). In other words, this experiment signifies that the optimal frontier point is not necessarily the closest relative to the current robot position. Thus, simple imitation learning based on traditional greedy frontier detection is not the best strategy.

B. Large-scale Indoor Exploration

This study investigates Q2 and Q3. We carried out multiple exploration experiments in WGx2 and WGx3, comparing against two state-of-the-art baseline approaches, Autoexplorer [3] and SMMR-Explorer [22], as well as *FR-Net Random*. The FR-Net Random finds frontier points by running FR-Net but disabling both A*-Net and Viz-Net. Testing and comparing this system highlights the importance of locating the optimal frontier point.

Beginning from the same position in the world, each method explores the WGx2 or WGx3 world and attempts to build a map out of it, and we have measured the time spent to build the entire map. Table II reports the exploration time of the four approaches. Only Neuro-Explorer and FR-Net Random achieved 99% coverage of the testing space, while Neuro-Explorer explores much faster than FR-Net Random. Furthermore, the other approaches terminated before completing the exploration - covering only 30% to 40% in WGx3. Moreover, Neuro-Explore outperforms other traditional approaches in terms of exploration time except for one occasion, which corresponds to covering only 30% of the large environment. The result observed with FR-Net Random instantiates why efficient planning matters in exploration tasks, where the random planning strategy in FR-Net Random takes about 86% more time to reach T_p on average than Neuro-Explorer. All results were obtained by taking the median of five exploration runs for each method.

C. Exploration Without Precise Localization

This study investigates Q4. Although the localization problem and planning problems are orthogonal, the significance of this study lies in the practical usability of our system

Methods	Exploration Time for WGx2/WGx3 (in $10^3 s$)			
	T_{30}	T_{50}	T_{90}	T_{99}
SMMR-Explore	0.85/1.38	2.09/ –	– / –	– / –
Autoexplorer	0.64/**1.37**	1.24/ –	2.55/ –	– / –
FR-Net Random	1.73/2.30	3.23/6.03	7.28/13.2	8.89/15.6
Neuro-Explorer	**0.46**/1.82	**0.94**/**2.69**	**2.47**/**6.46**	**4.82**/**8.29**

TABLE II: Comparisons of exploration time spent for different methods in WGx2 and WGx3 worlds: */* in each cell indicates the time taken in WGx2 and WGx3, respectively, and - means no available data due to incomplete exploration.

in more realistic scenarios. Indeed, this study is especially pertinent to indoor exploration problems due to the absence of GPS signals in such environments. To compensate for the unknown ground-truth robot position, we employed the SLAM toolbox [23] to estimate the robot's position and generate the occupancy grid map. Our system completed the exploration of WGx3, resulting in the reconstructed map shown in Fig 4(c). The experiment marked T_{50}, T_{90}, and T_{99} to be 4959s, 8334s, and 9275s, respectively. Compared to the result reported in Table II, the performance degradation may be due to inaccurate localization and mapping, resulting in inefficient planning during the exploration task.

VII. Conclusion

We introduced a learning-based autonomous exploration system in indoor environments with unknown spaces. Our FR-Net is capable of detecting FR cells in the input grid map. The detected FR cells are assessed by our A*-Net and Viz-Net, making a reasonable decision to the best frontier point. In our experiments, our method outperforms other traditional methods in terms of exploration performance and coverage rates. In the future, we aim to conduct an in-depth study of our neural networks, exploring various approaches to improve A*-Net and Viz-Net.

VIII. Acknowledgment

This project was supported in part by ITRC/IITP Program (IITP-2024-2020-0-01460), and in part by the NRF (NRF-2022R1A2B5B03001385 and 2021R1I1A1A01048639) in South Korea.

References

[1] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97.'Towards New Computational Principles for Robotics and Automation'*. IEEE, 1997, pp. 146–151.

[2] M. Keidar and G. A. Kaminka, "Efficient frontier detection for robot exploration," *International Journal of Robotics Research*, vol. 33, no. 2, pp. 215–236, 2014.

[3] K. M. Han and Y. J. Kim, "Autoexplorer: Autonomous exploration of unknown environments using fast frontier-region detection and parallel path planning," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 10 536–10 541.

[4] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[5] P. Senarathne, D. Wang, Z. Wang, and Q. Chen, "Efficient frontier detection and management for robot exploration," in *2013 IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems*, 2013, pp. 114–119.

[6] B. J. Julian, S. Karaman, and D. Rus, "On mutual information-based control of range sensing robots for mapping applications," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 5156–5163.

[7] D. Perea Ström, I. Bogoslavskyi, and C. Stachniss, "Robust exploration and homing for autonomous robots," *Robotics and Autonomous Systems*, vol. 90, pp. 125–135, 2017, special Issue on New Research Frontiers for Intelligent Autonomous Systems. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889016304730

[8] Z. Zhang, T. Henderson, V. Sze, and S. Karaman, "Fsmi: Fast computation of shannon mutual information for information-theoretic mapping," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 6912–6918.

[9] A. Asgharivaskasi, S. Koga, and N. Atanasov, "Active mapping via gradient ascent optimization of shannon mutual information over continuous se(3) trajectories," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 12 994–13 001.

[10] S. McCammon and G. A. Hollinger, "Topological path planning for autonomous information gathering," *Autonomous Robots*, vol. 45, pp. 821–841, 2021. [Online]. Available: https://doi.org/10.1007/s10514-021-10012-x

[11] Z. Zhang, J. Yu, J. Tang, Y. Xu, and Y. Wang, "Mr-topomap: Multi-robot exploration based on topological map in communication restricted environment," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10 794–10 801, 2022.

[12] R. Shrestha, F.-P. Tian, W. Feng, P. Tan, and R. Vaughan, "Learned map prediction for enhanced mobile robot exploration," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 1197–1204.

[13] D. Zhu, T. Li, D. Ho, C. Wang, and M. Q.-H. Meng, "Deep reinforcement learning supervised autonomous exploration in office environments," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 7548–7555.

[14] Z. Zheng, C. Cao, and J. Pan, "A hierarchical approach for mobile robot exploration in pedestrian crowd," *IEEE Robotics and Automation Letters*, vol. 7, no. 1, pp. 175–182, 2022.

[15] Y. Xu, J. Yu, J. Tang, J. Qiu, J. Wang, Y. Shen, Y. Wang, and H. Yang, "Explore-bench: Data sets, metrics and evaluations for frontier-based and deep-reinforcement-learning-based autonomous exploration," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 6225–6231.

[16] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham: Springer International Publishing, 2015, pp. 234–241.

[17] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *IEEE/CVF Computer Vision and Pattern Recognition*, 2017.

[18] E. Marder-Eppstein, E. Berger, T. Foote, B. Gerkey, and K. Konolige, "The office marathon: Robust navigation in an indoor office environment," in *International Conference on Robotics and Automation*, 2010.

[19] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Autonomous Robots*, 2013, software available at http://octomap.github.com. [Online]. Available: http://octomap.github.com

[20] C. Rösmann, F. Hoffmann, and T. Bertram, "Kinodynamic trajectory optimization and control for car-like robots," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 5681–5686.

[21] OpenRobotics, "willowgarage," Open Robotics, 11 2023. [Online]. Available: "https://github.com/osrf/gazebo_models/willowgarage"

[22] J. Yu, J. Tong, Y. Xu, Z. Xu, H. Dong, T. Yang, and Y. Wang, "Smmr-explore: Submap-based multi-robot exploration system with multi-robot multi-target potential field exploration method," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 8779–8785.

[23] S. Macenski and I. Jambrecic, "Slam toolbox: Slam for the dynamic world," *Journal of Open Source Software*, vol. 6, no. 61, p. 2783, 2021. [Online]. Available: https://doi.org/10.21105/joss.02783