2021 IEEE International Conference on Robotics and Automation (ICRA 2021)
May 31 - June 4, 2021, Xi'an, China

2021 IEEE International Conference on Robotics and Automation (ICRA) | 978-1-7281-9077-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICRA48506.2021.9561040

Extendable Navigation Network based Reinforcement Learning for Indoor Robot Exploration

Woo-Cheol Lee[1] Ming Chong Lim[1] and Han-Lim Choi[1]

***Abstract*— This paper presents a navigation network based deep reinforcement learning framework for autonomous indoor robot exploration. The presented method features a pattern cognitive non-myopic exploration strategy that can better reflect universal preferences for structure. We propose the Extendable Navigation Network (ENN) to encode the partially observed high-dimensional indoor Euclidean space to a sparse graph representation. The robot's motion is generated by a learned Q-network whose input is the ENN. The proposed framework is applied to a robot equipped with a 2D LIDAR sensor in the GAZEBO simulation where floor plans of real buildings are implemented. The experiments demonstrate the efficiency of the framework in terms of exploration time.**

I. INTRODUCTION

In robot application, autonomous planning of trajectories to reduce the unknown area in a given environment and build a map incrementally at the same time is known as robot exploration. Robot exploration plays an important role in a variety of tasks, such as search and rescue, planetary exploration and structure investigation, and the efficiency of the exploration process is crucial in maximizing the performance of these tasks.

Robot exploration is known to be close to an NP-complete problem such as the traveling salesman problem (TSP) or the art gallery problem [1]–[4]. Frontier exploration methods [1], [4]–[10] have traditionally been used, from a practical point of view, to overcome the complexity and uncertainty of the exploration problems resulting from the gradually expanding environment. In these approaches, the one-step information gain is inferred through either geometric heuristics [6], [7], [10] or prediction of map features in a probabilistic manner [1], [4], [8], [9] and greedy optimization is performed to maximize the information gain with respect to distance traveled.

Despite the difficulty of predicting future observations, non-myopic approaches have also been proposed [2], [3], [11], [12]. Oßwald et al. [3] proposed the use of TSP solver assuming that a prior map information in the form of topometric graph is available. Recently, deep reinforcement learning (DRL) techniques were also implemented to solve the autonomous exploration problems [2], [12]. The state-action value is inferred directly rather than making predictions on the features of the map. These approaches, however, suffers from the curse of dimensionality as the number of state-action pairs can be very significant as exploration progresses. Niroui et al. [2] used a frontier detection module to build the action space and resize the occupancy grid map to reduce the state dimension. Bai et al. [13] suggested setting discrete actions with a fixed distance from the robot at regular angular intervals. These reduction processes generally do not consider navigation-related features, resulting in possible exploration performance degradation.

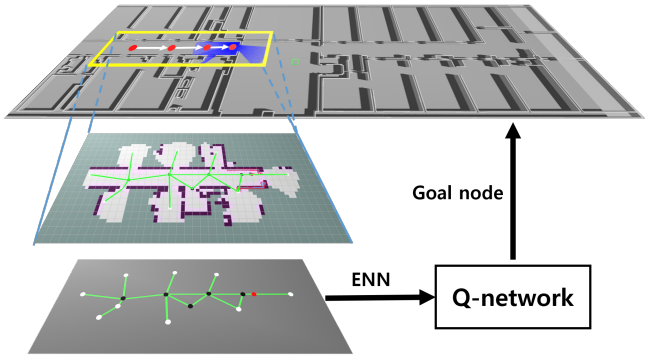

Fig. 1. A representative overview of the proposed ENN-RL. The robot generates an extendable navigation network (ENN) based on the observations up to the current time-step. The ENN maintains information regarding the current exploration such as the position of the robot (red node), visited positions (black nodes) and possible targets for exploration (white nodes). The Q-network takes the ENN as the input and outputs the destination for the next time-step.

Leveraging on these existing works, the DRL approach with a domain conversion of the map from 2D Euclidean space to a graph representation, as seen in Figure 1, is considered in this paper. We inherit the philosophy of existing studies that try to represent the indoor environment using the navigation network [14]–[17]. Among them, we focus on the visibility-based navigation network generation method [14], [15], which is based on the inter-visibility of locations. Generally, visibility-based approaches are good at capturing characteristic locations of significance (i.e., intersections, near the frontiers, behind walls, locations where the direction changes) which make the nodes in these navigation networks good action candidates. Previous works were generally focused on generating the network using a complete map. Hence, we propose a modified network called the Extended Navigation Network (ENN) which expands on a network recursively as new observations are obtained from the exploration.

Deep Q-learning is used to learn the optimal policy. While Convolution Neural Network (CNN) is popular in resolving tasks with two dimensional image representation inputs, we consider inputs in the form of a graph, and therefore, a Q-network consisting of graph convolution network (GCN)s is

[1]W.-C. Lee, M.C. Lim and H.-L. Choi are with the Department of Aerospace Engineering, KAIST, Daejeon, Korea. {wclee, mclim}@lics.kaist.ac.kr, hanlimc@kaist.ac.kr

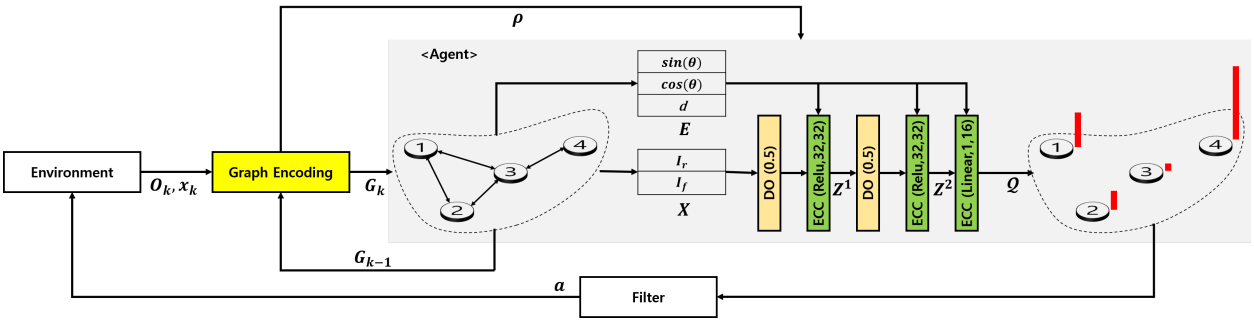

Fig. 2. An overview of the ENN-RL framework.

required. GCN can be broadly classified into two categories according to the local filtering method — the spectral method [18], [19] and spatial method [20]–[23]. In the spectral method, the filtering is performed using graph Laplacian, and the graph is limited to have a fixed structure with only changes in the node features. In our application, this is not a viable option since the number of nodes adjacent to a particular node, and also the ordering of the nodes, can change as the graph structure expands. On the other hand, spatial method performs local filtering in a way where weights are shared between neighbors, and works with graphs of varying sizes and connectivities. Among them, Edge-Conditioned Convolution (ECC) [23] consider not only node features but also the edge features, which is suitable for our application where geometric relationship of the indoor structure is important.

In this paper, Extendable Navigation Network based reinforcement learning (ENN-RL) framework is proposed to learn an efficient exploration policy. For this, ENN for 2D partial map encoding and ECC based Q-network are introduced. Our contributions are follows:

1) Firstly, the suggestion of domain conversion from 2D Euclidean space to a navigational graph network which can hold representative information in a concise yet sufficient manner for robot exploration,
2) secondly, the introduction of ENN to encode the indoor structures in an iterative and computationally efficient way,
3) and, lastly, an implementation of ECC based Q-network to learn the policy.

The structure of this paper is as follow: Section II presents the problem statement, which includes the concept of domain conversion. Section III describes the proposed framework. In section IV and V, the proposed ENN and ECC embedded Q-network are described, respectively. Section VI provides the performance evaluation. Finally, section VII is devoted to the conclusion.

II. Problem Statement

We consider a nonholonomic mobile robot equipped with a 2D LIDAR with limited sensing range. The robot operates in a closed 2D indoor environment $M \in \mathcal{F}^{w\times h}$, incrementally building an estimated map $\hat{M} \in \mathcal{F}^{w\times h}$, in which $\mathcal{F} \in \{F_o : \text{occupied}, F_f : \text{free}, F_u : \text{unknown}\}$ is the possible map configuration, where w and h are the maximum width and height of map, respectively. Following typical discounted return, the expected return at time k is given as:

$$\begin{aligned} R_k &= \mathbb{E}[\Sigma_{k'=k}^{K} \gamma^{k'-k} \rho(s_{k'}, h(s_{k'}), s_{k'+1})] \\ s_{k+1} &\sim f(s_k, h(s_k)) \end{aligned} \tag{1}$$

where, $s_k = (\hat{M}_k, x_k) \in \mathcal{S}$ is the state at time k, $x \in \mathbb{R}^2$ is the robot position, $f : \mathcal{S} \times \mathcal{U} \to \mathcal{S}$ is the transition function, $\rho : \mathcal{S} \times \mathcal{U} \times \mathcal{S} \to \mathbb{R}^1$ is the reward function, $h : \mathcal{S} \to \mathcal{U} \in \mathbb{R}^2$ is the policy (goal point in the 2D map), and $\gamma \in [0, 1)$ is the discount factor. The objective of this paper is to define a proper reward structure $\rho(\cdot)$, and to find an optimal policy h^* that maximizes the expected return R_k while reducing the total distance travelled in the exploration process.

Instead of dealing with the above problem directly, we propose the domain conversion from the Euclidean space to a graph which is represented as follows:

$$G = T(\hat{M}, x) \tag{2}$$

where, $T : \mathcal{F}^{w\times h} \times \mathbb{R}^2 \to \mathcal{G}$ is the conversion function from a Euclidean space map to a directed graph, $G = (\mathcal{V}, X, \mathcal{E}, E)$ with $|\mathcal{V}|$ nodes and $|\mathcal{E}|$ edges, where $\mathcal{V} = \{v_1, ..., v_{|\mathcal{V}|}\}$ is the set of nodes, $X \in \mathbb{R}^{|\mathcal{V}|\times d_N}$ is the collection of node features, $\mathcal{E} \in \{e_1, ..., e_{|\mathcal{E}|}\}$ is the set of edges, $E \in \mathbb{R}^{|\mathcal{E}|\times d_E}$ is the collection of edge features, d_N is dimension of the node features, and d_E is the dimension of the edge features. To apply the conversion from (1) to (2), the following assumptions are made:

Assumption 1: G is the sufficient statistics for $\hat{M}$ in terms of the navigation with the following components:

- Spatially representative nodes,
- proper connectivity between nodes,
- descriptive attributes for the nodes and edges (i.e., the robot location, the closeness with frontier, the distance and angle between nodes).

Assumption 2: The node set $\mathcal{V}$ provides sufficient actions for exploration.

With above assumptions, the problem in equation (1) is rewritten as follows:

$$R_k = \mathbb{E}[\Sigma_{k'=k}^{K} \gamma^{k'-k} \rho_g(G_{k'}, h_g(s_{k'}), G_{k'+1})] \tag{3}$$

where, $\rho_g : \mathcal{G} \times \mathcal{U} \times \mathcal{G} \rightarrow \Re^1$ is the reward function, and $h_g : \mathcal{G} \rightarrow \mathcal{U} \in \mathcal{V}$ is the policy whose output is a node in the graph network G. The detailed domain conversion mechanism, reward design, and policy learning method are described in the following sections.

III. Framework Overview

Figure 2 shows the overall framework of our method. A "Graph Encoding" block generates G_k and instead of a direct conversion from $\hat{M}_k$ to G_k, we opt to reduce computational burden by incrementally deriving G_k from existing map $\hat{M}_{k-1}$ and local observation O_k. We first approximate $\hat{M}_k$ as follows:

$$\begin{aligned} \hat{M}_k &= \bigcup_{k'=0}^{k} O_k \\ &= \hat{M}_{k-1} \cup O_k \end{aligned} \tag{4}$$

where O_k is one frame observation at time k. We assume that the conversion function T in equation (2) allows the following operation:

$$T(A \cup B, x) = T(A, x) \oplus T(B, x) \tag{5}$$

where $\oplus$ is the merging operation that combines two graphs. Then, following relationship can be established:

$$G_k = T(\hat{M}_{k-1}, x_k) \oplus T(O_k, x_k) \tag{6}$$

Based on equation (6), the computational load to obtain G_k is significantly reduced due to the following reasons: 1) $T(\hat{M}_{k-1}, x_k)$ can be obtained from G_{k-1}, with only changes to the attributes describing the robot location and 2) $T(O_k, x_k)$ will be significantly smaller than $T(\hat{M}_k, x_k)$ as the exploration progresses.

With the G_k input, a Q-network composed of Dropout (DO) and ECC layers is used to predict the Q-values of every nodes in G_k (shown as red bars in Figure 2). The Q-values pass through the user-designed filter and the Q-network outputs the final action. In the training phase, the weights in the network are learned through trial and error in the indoor environment. In the evaluation phase, action a_k is chosen based on the Q-values.

IV. Extendable Navigation Network

In this section, detailed description of equation (6) is provided. We design the conversion function $T(\cdot)$ and merging operation $\oplus$ with consideration of assumptions 1 and 2. We propose the ENN, taking visibility-based navigation network modeling approach [14] as a benchmark. The overall procedures are shown in Figure 3.

A. Visibility Map

The visibility map B_k (Figure 3-(a)) is obtained based on the inter-visibility perspective in O_k. Each cell i of O_k is given a value $b_i \in \{0, 0.5, 1\}$ representing obstacles, shadows and free space, respectively. The visibility of every other cells from i are determined using Bresenham's line algorithm [24]. Then, the visibility score $B_{k,i}$ is calculated as follows:

$$B_{k,i} = \frac{\sum_{j \in c_v} b_j}{\sum_{j \in c_f} b_j} \tag{7}$$

where c_v and c_f are the set of visible and free cells in O_k, respectively.

B. Local Navigation Network

To build local navigation network $T(O_k, x_k)$, we need to generate nodes and make connections between them. The candidate nodes are generated using the O_k and B_k. Jenk natural breaks optimization method [25] is applied to divide the cells into discrete number of levels (5 is used in this paper) according to their visibility scores (Algorithm 1, line 3). Cells within the same level are spatially clustered using connected components labeling (CCL) method. Unique labels and centroids C_k are given to every clustered regions excluding small or non-visible regions (Algorithm 1, line 5$\sim$ 10). Also, depending on the adjacency with unknown regions and relative level with neighboring regions, attributes $X_{i,u}$ (adjacency with unknown region) and $X_{i,t}$ (relative level) are assigned to the nodes (Algorithm 1, line 11$\sim$ 14).

Edges are added in a similar way to [14] (Algorithm 2). Firstly, because 'peak' and 'pit' nodes captures characteristic locations, they are registered by default (Algorithm 2, line 3). Then starting from 'pit' nodes, edges are made to neighboring nodes based on visibility and distance until it is connected to a 'peak' node (Algorithm 2, line 4$\sim$23). Multiple subnetworks may be generated in one map due to the (possible) presence of multiple 'peak' nodes. The subnetworks are also combined based on the distance and visibility (Algorithm 2, line 24$\sim$35).

C. Merge

This section describes the merge operation $\oplus$ in the equation (6). A distance-based cost matrix is built between the two node sets of $T(\hat{M}_{k-1}, x_k)$ and $T(O_k, x_k)$ as follows:

$$c_{ij} = \begin{cases} d_{ij}^2 & \text{(if } d_{ij} < \lambda \text{ and visible}(i,j)\text{=}True\text{)} \\ D & \text{(otherwise)} \end{cases} \tag{8}$$

where d_{ij} is the distance between node i in $T(\hat{M}_{k-1}, x_k)$ and node j in $T(O_k, x_k)$, λ is the gating threshold, and D is a sufficiently large number. The desired merge is one that minimizes the sum of costs and the Kuhn-Munkres algorithm [26] is used to obtained the optimal solution. When merging node j to node i, all neighbors of node j should also be visible of node i, otherwise, node i are j connected by an additional edge. Then, for every edges (i, j), the indoor structure related attributes are set — the angle encoding $(sin(\theta_{ij}), cos(\theta_{ij}))$ and distance $d(i, j)$.

V. Q-network

In this section, we describe the Q-network configurations including the reward design, and action filtering.

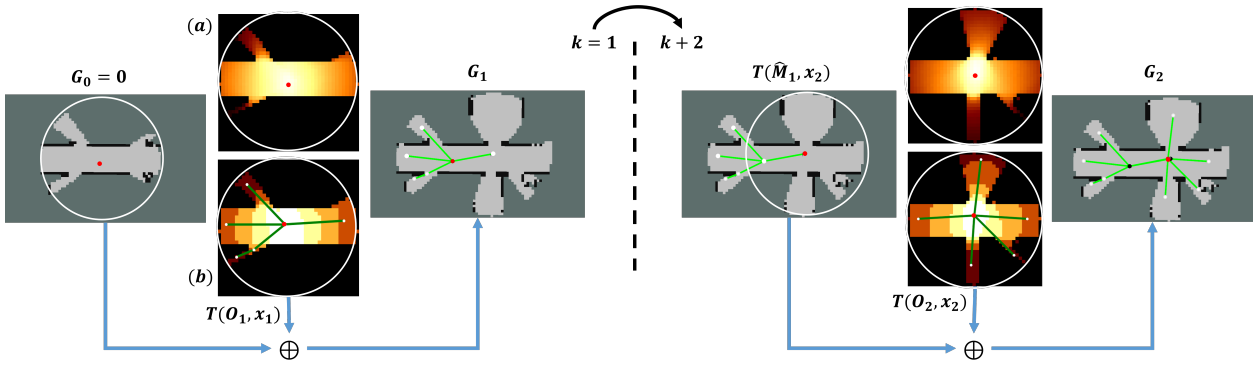

Fig. 3. Example of ENN process. At initial time $k = 1$, it starts with a empty graph G_0. (a) is the Visibility map of O_k and (b) is the level map and local navigation network $T(O_k, x_k)$. The node attributes are represented as the colors: red (the robot location), white (near frontier), black (otherwise).

Algorithm 1 Nodes Generation

Set area threshold θ, the number of level N_L
Input one frame observation O_k and visibility map B_k
Generate level map L_k by solving Jenks method with the input (B_k, N_L)
$C_k = \{\}$
for all $l \in \{1, 2, ..., N_L\}$ **do**
  Get masked level map $L_{l,k}$
  Get centroid nodes $C_{l,k}$ using CCL with input $L_{l,k}$
  $C_{l,k} \leftarrow C_{l,k} \setminus (\{i|A(i) < \theta\} \cup \{i|v(i) = 0\})$
  $C_k \leftarrow C_k \cup C_{l,k}$
end for
for all $i \in C_k$ **do**

$$X_{i,u} = \begin{cases} 1 & \text{(if } i \text{ is adjacent to unknown region)} \\ 0 & \text{(otherwise)} \end{cases} \tag{9}$$

$$X_{i,t} = \begin{cases} 'peak' & (\forall j \in N(i), l_j < l_i) \\ 'pit' & (\forall j \in N(i), l_i < l_j) \\ 'pass' & \text{(otherwise)} \end{cases} \tag{10}$$

end for

Algorithm 2 Local Navigation Network Generation

Input one frame observation O_k and nodes set C_k
Initialize empty local navigation network $G_L = (\mathcal{V}_L, \mathcal{E}_L)$
$\mathcal{V}_L \leftarrow \mathcal{V}_L \cup \{i|X_{i,t} \in 'peak', 'pit'\}$
for all $n \in \{n|X_{n,t} =' pit'\}$ **do**
  $i \leftarrow n$
  $flag_{peak} \leftarrow False$
  while $flag_{peak} = False$ **do**
    $J \leftarrow \{j|X_{j,t} =' peak'\} \cap \{j|\text{visible}(i,j) = True\}$
    $j \leftarrow \arg\min_{j \in J} ||x_i - x_j||$
    if $j \neq \emptyset$ **then**
      $\mathcal{E}_L \leftarrow \mathcal{E}_L \cup \{(n,j),(j,n)\}$
      $flag_{peak} \leftarrow True$
    else
      $K \leftarrow \{k|k \in N(i)\} \cap \{k|\text{visible}(i,k) = True\}$
      $k \leftarrow \arg\min_{k \in K} A(k)$
      $\mathcal{E}_L \leftarrow \mathcal{E}_L \cup \{(n,j),(j,n)\}$
      if $X_{k,t} =' peak'$ **then**
        $flag_{peak} \leftarrow True$
      end if
      $i \leftarrow k$
    end if
  end while
end for
$S \leftarrow$ connected subgraphs in G_L
while $|S| > 1$ **do**
  $G_s \leftarrow S(1)$
  $E \leftarrow \{(i,j),(j,i)|i \in \mathcal{V}(G_s)$ and $j \in (C_k \setminus \mathcal{V}(G_s))$ and visible$(i,j) = True\}$
  $e \leftarrow \arg\min_{e \in E} d(E)$
  if $e \neq \emptyset$ **then**
    $\mathcal{E}_L \leftarrow \mathcal{E}_L \cup e$
  else
    $G_L \leftarrow G_L \setminus G_s$
  end if
  $S \leftarrow$ connected subgraphs in G_L
end while

A. Network Configuration

G_k lies on non-Euclidean graph domain with a varying structure (expanding and arbitrarily ordered) and the edge attributes, such as distances and angles between nodes, should be considered. Edge-conditioned convolution [23] was selected for the Q-network for the following properties — local filtering for varying structure and edge conditioned filter generation. The ECC layer is formalized as follows:

$$Z_i^l = \sum_{j \in N(i) \cup i} MLP^l(E_{ji}) X_j^{l-1} + b^l \tag{11}$$

where $l \in \{0, ..., l_{max}\}$ is the layer index of the neural network, $N(i) = \{j|(j,i) \in \mathcal{E}\}$ is the neighbor nodes of i, $E_{ji} \in \Re^{d_E}$ is the attribute of edge (j,i), $MLP^l(\cdot)$ is the trainable neural network, and $b^l \in \Re^{d_l}$ is the bias. In our application, the node and edge attributes are defined as

follows:

$$X_i = \{\mathbb{I}_{r,i}, \mathbb{I}_{f,i}\}$$
$$E_{ij} = \{sin(\theta_{ij}), cos(\theta_{ij}), d(i,j)\} \quad (12)$$

where $\mathbb{I}_{r,i}$ is the indicator function and is 1 if the robot is at node i and 0 otherwise, $\mathbb{I}_{f,i} \in \{0,1\}$ is 1 if region i is adjacent to unknown region, and θ_{ij} is the angle of the edge (i,j). Our Q-network has 3 ECC layers and the configuration is described as DO(0.5)-ECC(Relu,32,32)-DO(0.5)-ECC(Relu,32,32)-ECC(Linear,1,16), where DO(p) denotes dropout with probability p, ECC(l,c,m) is the ECC layer with activation function l, c output channels, and m hidden neurons in the MLP.

B. Reward Function

Considering the similarity between exploration and the TSP problem, we define the reward function as follows:

$$\rho_{g,k} = \begin{cases} 1 & (\text{if } v_{r_k} \in N(v_{r_{k-1}}) \text{ and } X_{r_k,u} = 1) \\ -0.5 & (\text{if } v_{r_k} \notin N(v_{r_{k-1}}) \text{ and } X_{r_k,u} = 1) \\ -1 & (\text{otherwise}) \end{cases} \quad (13)$$

For the clarity of the reward signal, we exclude continuous distance or area metrics, which will be compensated in the action filtering process (section V-C). When the robot reaches a node that is a neighbor of the previous node and is adjacent to the unknown area, it receives $+1$ reward. Likewise, if the robot reaches to a node that is adjacent to the unknown area but not a neighbor of the previous node, it receives -0.5 reward. The -1 reward is given when the robot visits nodes that provide no information gain ($X_{r_k,u} = 0$). The reward structure above induces the robot to minimize passing through already visited nodes, since moving to a node that is not the neighbor of the current position will require transiting through some node(s) that has already been visited. With this reward function, we expect the robot to learn the pattern of the navigation network and explores in the direction of open space, as much as possible, at the start of the exploration by minimizing revisits. The Q-values are updated in the training phase using the Bellman equation as an iterative update:

$$Q_{k+1}(G,a) = \mathbb{E}_G[\rho_{g,k} + \gamma \max_{a'} Q_k(G',a')|G,a] \quad (14)$$

C. Action Filtering

It is difficult to differentiate between superior and inferior nodes, of the same setting, when simply relying on the above-mentioned equation (13). To compensate, we introduce the filtering process for Q-values by considering the distances:

$$q'_i = \begin{cases} q_i & (\text{if } d(i, v_{r_{k-1}}) < \theta_d) \\ q_i - \alpha d(i, v_{r_{k-1}}) & (\text{otherwise}) \end{cases} \quad (15)$$

We discount the Q-values of the nodes at a distance greater than an arbitrary threshold θ_d to discourage traveling over long distances in a single time-step. Using the filtered Q-values $Q' = \{q'_1, q'_2, ..., q'_{|\mathcal{V}|}\}$, the action that has the maximum value is selected:

$$a = \arg\max_{v_i}(Q') \quad (16)$$

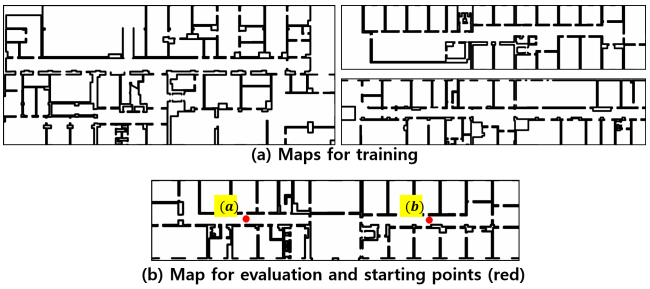

Fig. 4. (a) The three maps were used for training the Q-network. (b) A single map was used for evaluation of the Q-network. Two different starting positions were considered in the exploration evaluation.

VI. Experiment

A. Environment Setup

The GAZEBO simulation is used to generate the indoor environment and evaluate the exploration. The KTH floor plan dataset [27] is used as the indoor environment. The dataset provides floor plans of actual buildings on the KTH campus. We selected three floor plans for policy learning and one for evaluation (Figure 4). In the training step, for simplicity, we assume that the robot is equipped with a 360°LIDAR sensor as it moves from one node to another. In the evaluation phase, we transfer the trained Q-network to the GAZEBO environment, using the Turtlebot model equipped with a 2D LIDAR sensor. Gmapping [28] is used to build the cumulative slam map $\hat{M}_k$ and and the dynamic window approach (DWA) planner is used to guide the robot from one node to another. The simulation example is shown in Figure 5.

B. Analysis Method

To evaluate our framework, we use distance-cost-based frontier exploration (FE) as the baseline. Additionally, we implemented a similar cost-based greedy approach in the graph domain (ENN-greedy) for comparison. In the ENN-greedy approach, a node which is adjacent to the unknown area and is of the minimum distance from the current robot position is selected as the action for each time-step:

$$a = \arg\min_{v_i}(d(v_{r_k}, v_i)) \quad (17)$$

The evaluation metrics are as follows:

1) Cumulative reward gains from steps (ENN-RL vs ENN-greedy)
2) Information gain (in m^2) against travel distance (in m) (ENN-RL vs ENN-greedy vs Frontier exploration)

The first evaluation metric provides a measure of the learned ENN-RL model's performance improvement compared to the ENN-greedy. Five Monte Carlo simulations of the training phase with 20 episodes each are performed, and the average value is taken every step for the last 10 episodes judged to have been sufficiently learned. The second evaluation metric provides a measure of the information gain throughout the exploration. As described in Section V-B, we expect the robot to try to obtain as much information as possible in

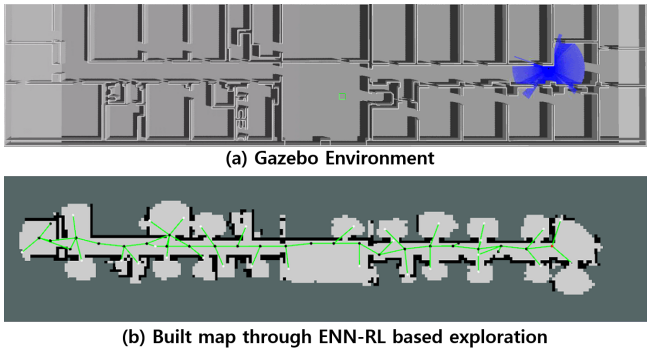

Fig. 5. Simulation example

the early phase of the exploration. From the two starting positions shown in Figure 4-(b), 5 episodes of exploration were conducted for each position with the mean values used to compute both evaluation metrics.

C. Result

The simulation results are shown in Figure 6 $\sim$ 8. Figure 6 shows that the learned ENN-RL model has a faster expected reward acquisition rate compared to ENN-greedy. The ratio of average, minimum, and maximum cumulative reward sizes is up to 10%, 17%, and 16%, respectively (Figure 7).

ENN-greedy approach and frontier exploration provides similar levels of information gain throughout the exploration process (Figure 8). Despite ENN being a sparse representation of the high-dimensional indoor Euclidean space, no obvious degradation in performance was observed. The comparable results show that our strategy of exploration using ENN, rather than the 2D Euclidean map, is feasible.

ENN-RL obtains higher levels of information gain at early stages of the exploration as seen from the gradient of the curve in Figure 8 and distance needed for 80% coverage. This performance improvement can be attributed to the design of the reward function in the Q-network. As mentioned in Section V-B, the reward function was designed to minimize revisits which motivates the robot to explore in the direction of open space as much as possible. The ability of the robot to obtain higher amounts of information at the early stages of exploration is a desirable characteristic since most exploration related tasks, such as search and rescue operations, can occur concurrently and is usually limited by the information of the environment. It is evident that a non-myopic graph-based deep reinforcement learning approach can achieve higher levels of efficiency for exploration in indoor environments compared to traditional myopic approaches.

VII. Conclusion and Future Work

In this paper, the indoor robot exploration problem using a reinforcement learning framework was investigated. A methodology for converting the map in high-dimensional Euclidean domain to a graph representation was introduced and ECC embedded Q-network was used for the policy learning. The superior results obtained from the application

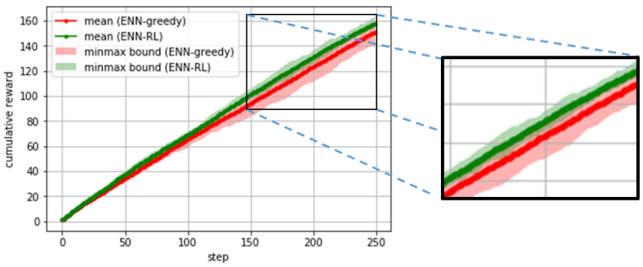

Fig. 6. Cumulative reward gains of ENN-greedy and ENN-RL

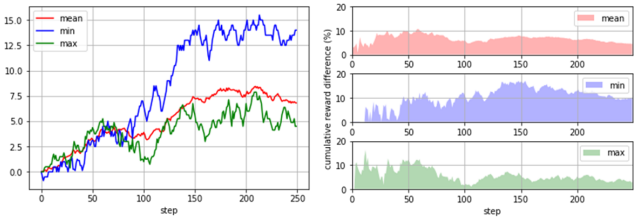

Fig. 7. Differences (left) and ratio (right) of cumulative reward gains between ENN-greedy and ENN-RL

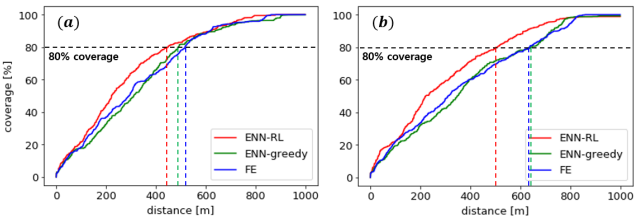

Fig. 8. Information gain against travel distance from the two starting points (a) and (b) shown in Figure 4.

of ENN-RL in the indoor environment of real buildings shows that pattern cognitive exploration can be a feasible way to achieve efficient exploration in indoor environments.

The ENN, in terms of proper expression of indoor spaces, can be improved in future works. The capability of the graph network is not limited to the storage of only geometric information and additional information of explored spaces, such as semantic descriptions, can be included in the ENN to enhance the exploration efficiency. In this paper, the merging operation between two graphs was performed in a simplistic and deterministic manner with only consideration on distance and visibility. Improvements may include a more holistic and complete approach to the clustering of nodes such that (possibly) redundant nodes (e.g., multiple nodes within a room) can be minimized.

Acknowledgment

This research was supported by Unmanned Vehicles Core Technology Research and Development Program through the National Research Foundation of Korea(NRF), Unmanned Vehicle Advanced Research Center(UVARC) funded by the Ministry of Science and ICT, the Republic of Korea (2020M3C1C1A01082375)

REFERENCES

[1] 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)*, pp. 1197–1204, IEEE, 2019.

[2] F. Niroui, K. Zhang, Z. Kashino, and G. Nejat, "Deep reinforcement learning robot for search and rescue applications: Exploration in unknown cluttered environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 610–617, 2019.

[3] S. Oßwald, M. Bennewitz, W. Burgard, and C. Stachniss, "Speeding-up robot exploration by exploiting background information," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 716–723, 2016.

[4] L. Ly and Y.-H. R. Tsai, "Autonomous exploration, reconstruction, and surveillance of 3d environments aided by deep learning," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 5467–5473, IEEE, 2019.

[5] 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'*, pp. 146–151, IEEE, 1997.

[6] H. Umari and S. Mukhopadhyay, "Autonomous robotic exploration based on multiple rapidly-exploring randomized trees," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1396–1402, IEEE, 2017.

[7] F. Bourgault, A. A. Makarenko, S. B. Williams, B. Grocholsky, and H. F. Durrant-Whyte, "Information based adaptive robotic exploration," in *IEEE/RSJ international conference on intelligent robots and systems*, vol. 1, pp. 540–545, IEEE, 2002.

[8] M. G. Jadidi, J. V. Miro, and G. Dissanayake, "Mutual information-based exploration on continuous occupancy maps," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6086–6092, IEEE, 2015.

[9] S. Bai, J. Wang, F. Chen, and B. Englot, "Information-theoretic exploration with bayesian optimization," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1816–1822, IEEE, 2016.

[10] J. M. Pimentel, M. S. Alvim, M. F. Campos, and D. G. Macharet, "Information-driven rapidly-exploring random tree for efficient environment exploration," *Journal of Intelligent & Robotic Systems*, vol. 91, no. 2, pp. 313–331, 2018.

[11] F. Niroui, B. Sprenger, and G. Nejat, "Robot exploration in unknown cluttered environments when dealing with uncertainty," in *2017 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS)*, pp. 224–229, IEEE, 2017.

[12] L. Tai and M. Liu, "A robot exploration strategy based on q-learning network," in *2016 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, pp. 57–62, IEEE, 2016.

[13] S. Bai, F. Chen, and B. Englot, "Toward autonomous mapping and exploration for mobile robots through deep supervised learning," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2379–2384, IEEE, 2017.

[14] Y. Pang, L. Zhou, B. Lin, G. Lv, and C. Zhang, "Generation of navigation networks for corridor spaces based on indoor visibility map," *International Journal of Geographical Information Science*, vol. 34, no. 1, pp. 177–201, 2020.

[15] A. Kneidl, A. Borrmann, and D. Hartmann, "Generation and use of sparse navigation graphs for microscopic pedestrian simulation models," *Advanced Engineering Informatics*, vol. 26, no. 4, pp. 669–680, 2012.

[16] A. Jamali, A. A. Rahman, P. Boguslawski, P. Kumar, and C. M. Gold, "An automated 3d modeling of topological indoor navigation network," *GeoJournal*, vol. 82, no. 1, pp. 157–170, 2017.

[17] G. Gröger and L. Plümer, "Derivation of 3d indoor models by grammars for route planning," *Photogrammetrie-Fernerkundung-Geoinformation*, vol. 2010, no. 3, pp. 191–206, 2010.

[18] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," *arXiv preprint arXiv:1312.6203*, 2013.

[19] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in neural information processing systems*, pp. 3844–3852, 2016.

[20] D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams, "Convolutional networks on graphs for learning molecular fingerprints," in *Advances in neural information processing systems*, pp. 2224–2232, 2015.

[21] J. Atwood and D. Towsley, "Diffusion-convolutional neural networks," in *Advances in neural information processing systems*, pp. 1993–2001, 2016.

[22] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[23] M. Simonovsky and N. Komodakis, "Dynamic edge-conditioned filters in convolutional neural networks on graphs," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3693–3702, 2017.

[24] M. B. Dillencourt, H. Samet, and M. Tamminen, "A general approach to connected-component labeling for arbitrary image representations," *Journal of the ACM (JACM)*, vol. 39, no. 2, pp. 253–280, 1992.

[25] G. F. Jenks, "The data model concept in statistical mapping," *International yearbook of cartography*, vol. 7, pp. 186–190, 1967.

[26] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[27] A. Aydemir, P. Jensfelt, and J. Folkesson, "What can we learn from 38,000 rooms? reasoning about unexplored space in indoor environments," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4675–4682, IEEE, 2012.

[28] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.