Learning Frontier Selection for Navigation in Unseen Structured Environment

Ganesh Iyer*
giyer@andrew.cmu.edu

Vidhi Jain*
vidhij@andrew.cmu.edu

Abstract

Learning end-to-end policy for navigation with a focus on intelligent exploration is a difficult task in robotics. While methods like soft-Q learning and ensembles of policies can demonstrate navigation behaviors in completely observed maps, we currently do not have ways of extending these policies to unexplored or partially explored environments. To this end, we propose a hierarchical formulation to tackle the problem of learning based efficient exploration. We decompose the the task into two sub-problems: selecting the next best goal in the visible space, followed by efficiently navigating to this space in the partial map setting. We propose a global policy network that learns the selection of the next best goal('frontier') in the observable space, followed by a local policy that learns low-level navigation conditioned on different goal embeddings in known environments. Our key approach to this decomposition is to enable the independent training of both of these components by creating randomized gridmaps on a large-scale for both subtasks. We demonstrate our approach by training both policies for navigation in the `FourRooms` environment in gym-minigrid. The final combined policy is, therefore, an end-to-end differentiable policy for map exploration. We evaluate the performance of the goal-conditioned local policy and present analysis of sparse reward based the global policy.

Navigation, Exploration, Reinforcement Learning

1 Introduction

Navigation in known environments is considered to be solved efficiently by both planning and reinforcement learning (RL) based methods. This stems from the fact that if the complete information in map is available, optimal methods can be developed to solve point goal navigation tasks. While planning based approaches offer completeness and strong formal guarantees, the next feasible goals is often recorded and selected based on heuristics, like graph-based methods, or probabilistic sampling based methods for continuous spaces. However, tackling efficient navigation in unexplored maps is difficult cause it does not have a complete solution. We propose a solution for the problem of unexplored-map navigation in gridmap environments, that is based on the selection of '**frontiers**'. Classically, we consider frontiers to be points at the boundary of known regions of a map, such that they can be considered important points to facilitate exploration in a map. Therefore, we define such frontiers as sub-goals, and learn a policy that can select such frontiers on partial grid maps.

When navigating in unknown environments, humans often invoke a decision criteria. This could be the next subgoal, and the decision could either be based on some semantic context, or some such prior. Learning how to select the next subgoal for higher level task such as coverage of the map is a challenging open question.

Assume that we have a policy that can choose the sutiable frontier to navigate, we also require another low level policy that can facilitate navigation to these sub-goals. This poses a challenge, since it can be difficult to develop differentiable planning frameworks. We, therefore, rely on a hierarchical approach and employ a second policy for action selection to navigate in known environments.

Our key contribution is the decomposition of the map exploration process into a two stage approach that results in a hierarchical and modular differentiable policy for navigation in unknown environments – a “global" policy that can propose frontier locations that lead to high coverage, and a “local" policy trained on randomly generated submaps. This enables us to learn data-driven exploration strategies to later improve the global policy with different kinds of reward functions.

2 Related Work

Robot Navigation Our work heavily derives from frontier based robot exploration literature. The concept of frontiers for exploration of unknown spaces was first introduced in Yamauchi [1997] by Yamauchi, in which regions on the boundary between known and unknown regions are considered valuable targets to explore to increase information about the environment. Traditionally, frontier based exploration is focused on geometric methods for navigation using SLAM, as demonstrated in works like Stachniss et al. [2004], Dai et al. [2017], and more recently deep learning based methods like Bloesch et al. [2018].

More recently, active learning methods for navigation have been proposed as a solution to downstream robot control tasks, that propose to navigate end-to-end with implicit mapping. In many of these works, the selection of frontiers is implicit, and is attributed to semantic priors (Yang et al. [2019]), or geometric priors (Chen et al. [2019]). Some of the work that comes close to our work regarding the explicit use of frontiers is Stein et al. [2018], where the expected cost value for each state action pair is calculated by observing trajectories of an optimistic planner.

Hierarchical Reinforcement Learning (HRL) We consider decomposing the actual reinforcement learning problem down into sub-problems such that solving them leads to a more efficient or powerful solution to the original problem. With the advent of Deep Learning, hierarchical representations are considered as an interaction between multiple policy networks. Such representations have been useful in solving all kinds of tasks. Kulkarni et al. [2016] was the first to introduce a hierarchical DQN (H-DQN) structures operating at different time scales to solve the sparse reward problem of Montezuma’s revenge. Al-Shedivat et al. [2018] also propose a hierarchical PPO formulation for the gym-minigrid Chevalier-Boisvert et al. [2018] environments. More recently, hierarchical policies have also been used for embodied question answering. Das et al. [2018] decomposes the problem of navigation for question-answering into a planner and controller framework, with the controller pre-trained by imitation learning based method.

In contrast, our local policy, while similar in spirit to a controller focused on grid-like environments, is trained by reinforcement learning in randomized grid maps to enable generalization. Our work is very close to Chaplot et al. [2020] which demonstrated such a hierarchical policy for pointgoal navigation tasks in 3D environments. Our method is different in two respects. Firstly, our local and global policies can be trained in a disjoint manner and combined to solve the task. Secondly, our global policy model can be decomposed to account for a soft representation, such that we can provide different exploration strategies for complete map exploration. Further, it is easy to see how our proposed global frontiers could also become pointgoals for navigation, although that is not the main focus of this work.

Learning based Planning We discuss this topic, since our local policy can be considered as a short term goal driven planner but trained as goal-conditioned RL, and our global policy utilizes an A* star planner for learning high-level policy. Eysenbach et al Eysenbach et al. [2019] propose search on the replay buffer that utilizes the value function of goal-conditioned reinforcement learning policy to add edge weights and create a graph structure of states represented as nodes. This serves the graph based planning techniques to plan waypoints to the goal.

Reinforcement Learning based approaches for planning have recently been proposed as differentiable planners for higher level tasks. These works leverage on unrolled dynamic computation graphs created in popular libraries to create differentiable planning mechanisms. Value Iteration Networks Tamar et al. [2016] by Tamar et. al couple attention mechanisms with value maps to account for differentiable planning in grid like discrete environments. Gated Graph Planning Networks Lee et al. [2018] further improves on this by including long-term memory into their framework using LSTMs.

In contrast to such explicit differentiable planning modules, our local policy is a goal driven policy network that is trained on randomized maps to improve efficiency. On grid-like environments, this has proven to be effective for local planning, as seen in works focusing on goal conditioned navigation, like Goyal et al. [2019].

3 Method

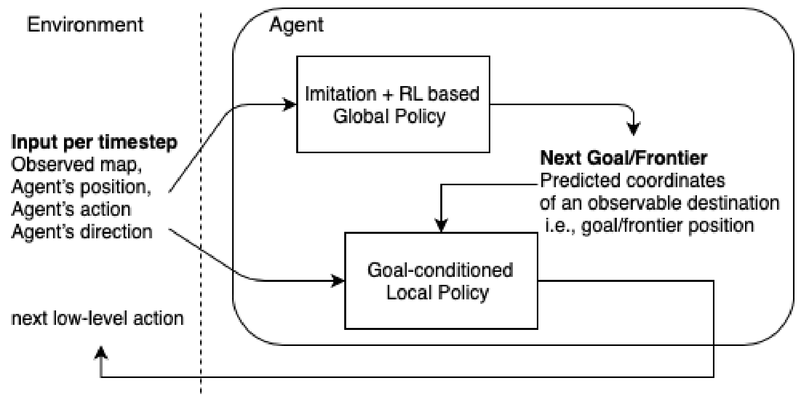


Figure 1: Proposed Architecture for integration of local and global policy

In partially observed environments, the reward signal is sparse because it is only given to the agent when it reaches a goal state. Further, this location in the map may not be currently observable, making this task difficult to train using off-the-shelf RL algorithms. However, abstracting the action space from low-level actions to frontier selection can help to alleviate this problem.

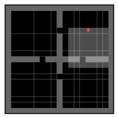
We decompose the navigation problem into sub-components and learn them separately. First, we learn a local policy which can navigate to a given goal position in a known observed environment. Second, we learn a global policy to predict intended frontier/goal position in the observable space based on agent's map of the partially revealed environment, agent's position, direction, and previous history of actions. Figure 1 illustrates the integration architecture that shows how the global policy could provide the next frontier/goal coordinates to the local policy.

The global policy is trained to strategically cover all areas of the map when the agent can observe only limited field of view at a location per timestep. We maintain the agent's view of the map observed so far by initializing the agent in the center of a 2D grid of maximum map size. Given the observed map view of the agent, the task of the global policy is to propose the coordinates where the agent should go next. The teleportation to the predicted coordinates are executed with a planner using A* search. The coordinates are continuous valued inputs in $\{-1, 1\}$, and represented in terms of the distance r and the angle θ from the agent's frame of view. If the coordinates are outside the observed area so far, no path can be found by the planner and the agent does not move in this case. The objective task is to maximize the coverage, as used in Chaplot et al. [2020]. We discuss the rewards in the experiments section in detail.

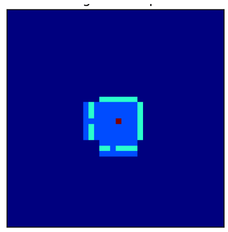
The local policy is a goal-conditioned reinforcement learning model which takes in the intermittent goal and executes next low level actions for navigation. Upon reaching the goal, this observation should trigger the global policy to predict another intermittent goal. This policy is trained to navigate to a random goal in fully observable environment such that the agent position, goal position and the observed partial map mask are all randomly selected for a given episode. We construct partial map masks as an environment to train the local policy as shown in Fig 3. These submaps are extracted based on agent's construction of the observed map, obtained during execution of A* planner.

In order to promote generalization to all kinds of partially observable submaps, we provide a curriculum-learning based strategy to introduce submaps during training, particularly, based on the the size of the submap. The intuition is that the bigger the submap, the higher is its complexity due to a larger possibility space of goal and agent starting points. Therefore, by providing a natural curriculum of increasingly diverse submaps, we hope alleviate the training complexity that comes with the training for long horizon navigation tasks.

4 Experimental setup



(a) Environment



(b) Agent's observed map

Figure 2: The environment and agent's observed map at initialization. (a) the classic `Fourrooms` for **coverage** task, (b) the agent's observed view

We used the `FourRooms` environment in gym-minigrid Chevalier-Boisvert et al. [2018], which is a 20x20 grid with a goal position, as shown in Figure 2a. Note that the maps are randomized in terms of the doors for each of the rooms and the agent's start location. Based on the partial observation of the agent, we construct the observed map as shown in 2b to serve as the input to both global and local policy networks. Though our approach is tested with this setup, it can be generalized to other kinds of partially observable discrete environments. Note that our observation model in gridworld follows a partially observable markov decision process (POMDP), since we do not have access to the complete state information, mainly the map.

First, we extract the current observed map of the agent, agent's action and direction, along with the selected goal/frontier coordinates over multiple trajectories while exploring the map. These trajectories are collected using a graph-based planner, where sub-goals are provided as the nearest boundary value frontier from the current location of the robot. Note that this step could be replaced with actual human trajectories or human selected frontiers.

Given that the environment size is s, then agent view of the map is created such that the agent is always initialized at the center of a grid sized $2s$. However, the agent starts in the environment from a random location that not known a-priori. Thereon, the partial views from agent's observation are extracted and aggregated to represent the area observed so far by the agent. This map of observed area contains a compact encoding of whether the grid location is unseen, empty or is a wall for the `FourRooms` environment. In real world, this is analogous to discretizing the space into grid, using object detection module and maintaining a numeric encoding of different object types seen so far.

We train both the policies using the current observed map as seen by the agent. The agent is initialized with a random position and orientation in the grid map. It is important to note that we assume the agent has memory and as it traverses in the map, its belief over the map grows as visible regions. Further, this results in different partial grid-maps based on the initialization of the agent.

4.1 Local policy

Our custom environment in gym-minigrid enables the construction of fully visible environments with the partial map masks as shown in Figure 3. These masks are created based on a expert trajectories, which is a frontier based planner in our case. Due to navigation with complete observability in partial maps, we can modify the reward signal to incorporate a dense reward. We consider two types of possible reward functions:

Euclidean Distance:

$$R_t = \begin{cases} 100, & \text{if } p_{goal} = p_t \\ -\gamma(\sqrt{p_{goal} - p_t}, & \text{otherwise} \end{cases}$$

Shortest Path Length reward:

$$R_t = \begin{cases} 100, & \text{if } p_{goal} = p_t \\ -\gamma C_{p_{goal} \sim G}[p_t], & \text{otherwise} \end{cases}$$

Here, p_{goal} is the goal position, p_t is the position of the agent at time t, γ is a weight term, and $C_{p_{goal} \sim G}$ indicates the shortest distance in pixels from the goal p_{goal}. (For reference the heatmap in Figure 4(b) indicates the cost from the goal). For the experiments, we chose γ = 0.1. The negative value of reward when agent is not at the goal location indicates penalty based on the distance from the goal.

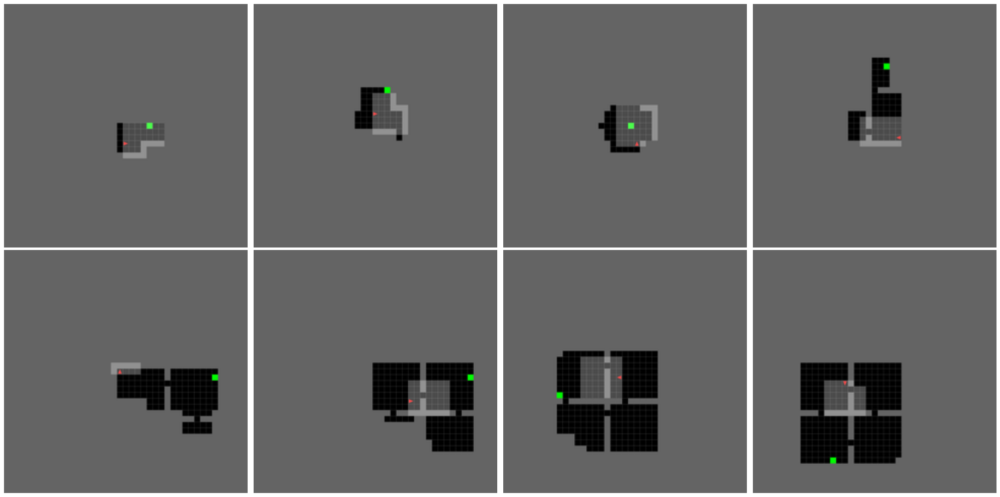


Figure 3: Left-Right: An example of the curriculum provided during training. We assume that the agent starts at the center (20x20) and proceeds to expand outwards based on its current direction and view point. Therefore, the agent builds a map as it completes the `FourRooms` environment. This curriculum simulates this experience explicitly in training.

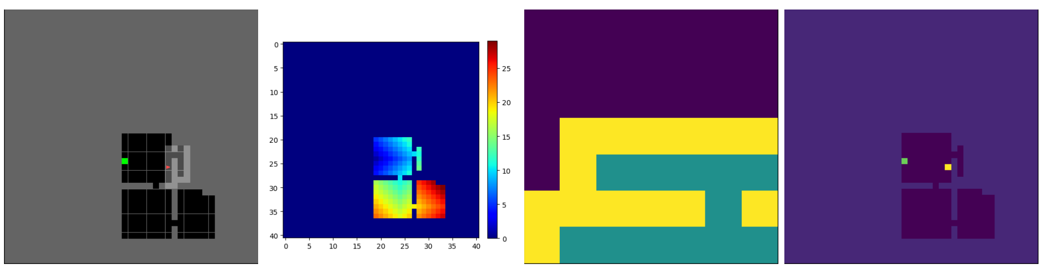


Figure 4: Left-Right (a-d): (a) shows the actual submap the agent is navigating in. (b) shows the cost to reach goal from all points in the submap. (c) indicates the local observation of the agent (baseline) (d) shows the global view as the observation

To further understand the curriculum, Fig 3 shows some variations of the partially observable maps in FourRooms environment of gym-minigrid in the sequential order that they were provided at training time. Note that since we obtain these using a graph based planner, each room may be variably explored based on the initial starting position, leading to a rich diversity in possible submaps.

To train the local policy, we use an asynchronous variant of the Proximal Policy Optimization (PPO) for learning this task. We use a convolutional LSTM network, that takes 40x40 input RGB gridmaps as input, and provides a discrete action selection as output. The convolutional netwoks consist of 4 convolutional layers with 64 filters and 2x2 kernel size. We use a LSTM hidden dimension size of 128. Both the actor and critic models have the same structure. We train for a total of 20000 submaps, with a repetition frequency of 20 episodes per submap and 8 parallel environments (3.2 million episodes). As an additional baseline, we also consider the task of exploration using only local views. An example of local views vs global views is best explained in the Figure 4. For this task, only a sparse reward is provided using the same curriculum strategy to promote exploration.

4.2 Global Policy

To solve the coverage task in minimum number of steps, we train the global policy network with two variants in reward structures:

Sparse Reward The sparse reward is given only upon completion of the task of coverage of the map.

$$R^{sparse} = 1. - 0.9 \times (\text{step count / max steps})$$

where the max steps refers to the maximum number of coordinates that the global policy can propose.

Dense Reward The dense reward is provided at every coordinate proposed by the global policy. If the predicted coordinate is outside the observed area, the model is penalized by how far the coordinate is from the agent. If the predicted coordinate is within the observed area, the model is rewarded based on the change in the observed area by moving to the new coordinate from previous one. Upon completion of coverage, the model is awarded a large bonus depending on the map and a similar penalty as in sparse reward setting. Overall, the dense reward is as follows:

$$R_t^{dense} = \begin{cases} -(\mathbf{g} - \mathbf{a})^2 & \text{if } M_t[\mathbf{g}] = 0 \\ M_{t+1} - M_t & \text{if } M_t[\mathbf{g}] = 1 \\ B - \text{step count} & \text{if coverage task is done} \end{cases}$$

where **a** represents the agent's position, **g** represents the proposed coordinates by the global policy, M_k represents the observed area mask at timestep k where unseen is 0, seen is 1, and B represents the large bonus on task completion, which is dependent on the map size.

To train the global policy, we again use the PPO with 2D continuous action space to represent the frontier as polar coordinate from the agent as the origin. We use a MLP policy with 2 layers of 64 hidden units. The input is same as the local policy as 40x40 gridmap. The environment had to be constructed to execute actual path for the agent to the predicted frontier location. Note that we could not use teleportation for the agent as a feasible path may not exist to predicted frontier by the policy.

5 Results

5.1 Local Policy

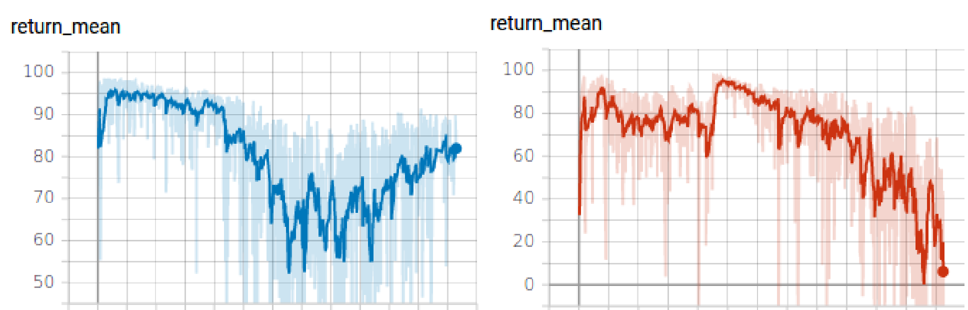


Figure 5: Mean Return of the local policy. Left: Trained using Euclidean distance reward, Right: Trained using Shortest Path Length reward

Figure 5 shows the mean return for the local policy. Note that this is with the curriculum training, so y-axis indicates the increasing map size. During training with euclidean distance, we notice a decrease in the mean return, but this improves over time, until the agent is comparably efficient at reaching the goal before the episode terminates. However, this is not the case for training with shortest path length reward. We hypothesize that since we were using the same submap update frequency it is possible that this is not enough for the shortest path reward and it may need additional epsiodes in the same submap before understanding this relationship.

5.2 Global Policy

In order to compare the performance of policies trained with different reward structure, we create an evaluation environment where both policies are evaluated in terms of the rewards obtained over 20K total proposed coordinates. The reward structure for the evaluation environment is:

$$1. - 0.9 \times (\text{step count / max count}), \quad \text{where max count} = 1000.$$

The results shown in the figure 6 are evaluated for sparse and dense variants of global policy where model was trained for 1 million total timesteps. Here timestep refers to the number of coordinates proposed by the policy. If the coverage task was done or maximum of 1000 coordinates were proposed, the environment was reset with different agent position and openings within the doors. We can see that the policy trained with dense reward successfully performs coverage on 92 episodes

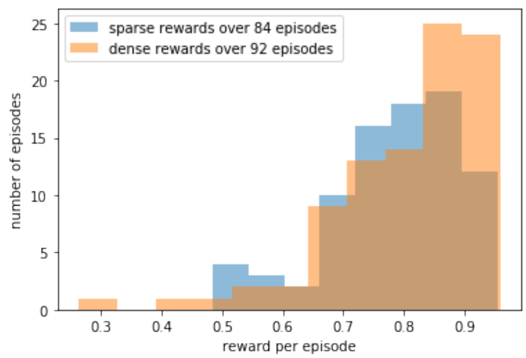


Figure 6: Comparison between sparse and dense reward training approach for global policy

with mean reward of 0.81. On the other hand, sparse policy performs slightly worse by successfully completing only 85 episodes with mean reward of 0.78.

On training for both the global policy variants for 2 million timesteps, we observe that the sparse reward based policy improves as shown in figure 7 but the dense reward based policy collapses to a fixed value. This is evident in the drastic dip in the mean episode reward and a corresponding peak in the policy gradient loss in figure 8.

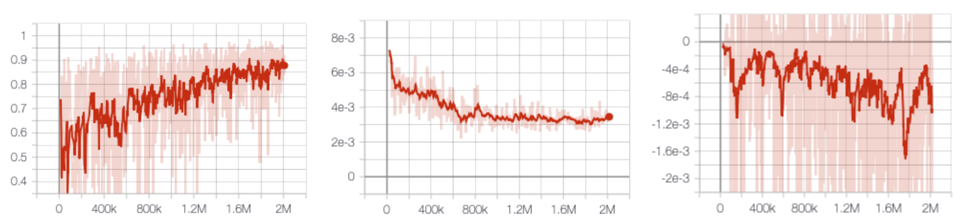


Figure 7: Training for sparse reward based global policy: (a) Episode reward. (b) value function loss. (c) policy gradient loss.

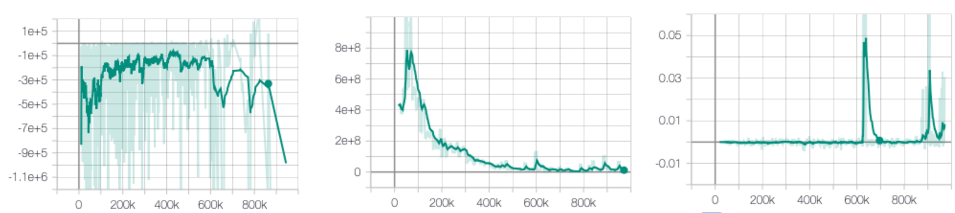
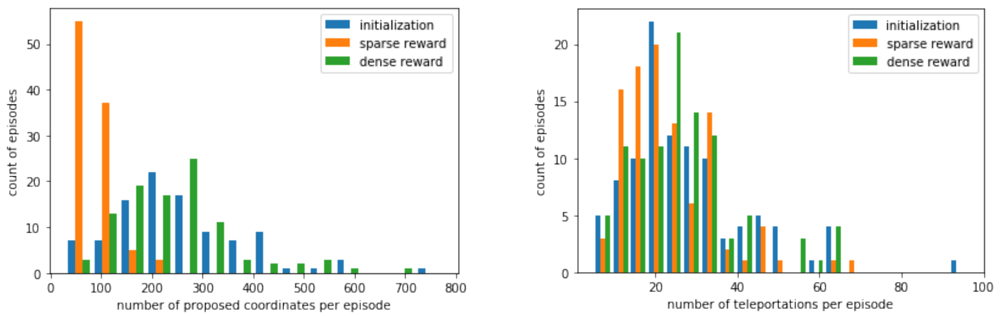


Figure 8: Training for dense reward based global policy: (a) Episode reward. (b) value function loss. (c) policy gradient loss.

To analyze both the policy variants, we evaluate the performance in terms of the number of proposed coordinates and the number of coordinates that are indeed used for teleportation. Note that we consider the policies after 2 million timesteps of training, and a better comparison could be between the best policy checkpoints of both the variants. Figure 9a shows that the sparse reward based policy uses fewer coordinates across episodes with most values as less than 150 timesteps, whereas the dense reward based policy is comparable to random initialization. In figure 9b, the number of coordinates used for teleportation for sparse reward based policy is lesser than the dense variant and random initialization. Note that the model was not trained to minimize the number of coordinates used to teleport, but its decrease is correlated with decrease in the overall proposed coordinates.



(a) the number of coordinates proposed per episode (b) the number of coordinates used for teleportation

Figure 9: Histograms to compare variants of global policy over 100 episodes for evaluation. The policy that proposes lesser number of coordinates and uses lesser steps to teleport is better.

6 Discussion

The current approach to train the global policy involves continuous space of predictions to correspond to the coordinates. This could be made simpler in discrete action space where the model learns to predict the direction in which the nearest frontier should be explored. However, we would be relying on a strong assumption that nearest frontiers are the ones to be selected for good map coverage, but that is not true as in a limited budget to teleport, the nearest frontier within the current room might be a poor choice than an opening leading to an another room. We did not study how to represent the action space for learning global policy efficiently and believe it to be an open question.

Given a diverse dataset of frontiers in a map, we could train a global policy that could imitate such frontier selections. These demonstrations can be used to train a neural network to output the frontier that will be selected. In our case, this network is trained to find the nearest feasible frontier for exploration. We consider training this global policy in a supervised manner first. We hope to extend this to a reinforcement learning based policy that learns to provide feasible frontier locations in the map that can maximize time constrained map exploration as a reward.

7 Conclusion

We presented a hierarchical decomposition for learning navigation in partially observed environments as: (1) local policy for learning goal-conditioned navigation (2) global policy for learning strategic selection of coordinates for the tasks like coverage of the map. We have evaluated both the components in `FourRooms` environment with randomization in terms of position of room openings and agent position for both global and local policy. We also randomize the goal positions, and the submaps for the environment for training the local policy. We have demonstrated the performance of goal-conditioned local policy with curriculum based randomization introduced during training. We also show the global policy's performance with sparse reward and compare with a dense reward variant.

Broader Impact

Learning can improve frontier selection by incorporating the environment's structure and semantics. Apart from learning from human behavior, learning based approach to selecting the frontiers can be useful to learn the structure of the environment. For example, among different indoor environment, the semantic proximity of objects in the kitchen and the objects in washroom still holds. If the robot is tasked to fetch something from refrigerator, it will select frontiers which have a high probability to reach the kitchen or dining area.

References

Maruan Al-Shedivat, Lisa Lee, Ruslan Salakhutdinov, and Eric P. Xing. On the complexity of exploration in goal-driven navigation. *CoRR*, abs/1811.06889, 2018. URL http://arxiv.org/abs/1811.06889.

Michael Bloesch, Jan Czarnowski, Ronald Clark, Stefan Leutenegger, and Andrew Davison. Codeslam - learning a compact, optimisable representation for dense visual slam. pages 2560–2568, 06 2018. doi: 10.1109/CVPR.2018.00271.

Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HklXn1BKDH.

Tao Chen, Saurabh Gupta, and Abhinav Gupta. Learning exploration policies for navigation. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=SyMWn05F7.

Maxime Chevalier-Boisvert, Lucas Willems, and Suman Pal. Minimalistic gridworld environment for openai gym. https://github.com/maximecb/gym-minigrid, 2018.

Angela Dai, Matthias Nießner, Michael Zollöfer, Shahram Izadi, and Christian Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface re-integration. *ACM Transactions on Graphics 2017 (TOG)*, 2017.

Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied Question Answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Benjamin Eysenbach, Ruslan Salakhutdinov, and Sergey Levine. Search on the replay buffer: Bridging planning and reinforcement learning. *CoRR*, abs/1906.05253, 2019. URL http://arxiv.org/abs/1906.05253.

Anirudh Goyal, Riashat Islam, DJ Strouse, Zafarali Ahmed, Hugo Larochelle, Matthew Botvinick, Sergey Levine, and Yoshua Bengio. Transfer and exploration via the information bottleneck. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJg8yhAqKm.

Tejas D. Kulkarni, Karthik R. Narasimhan, Ardavan Saeedi, and Joshua B. Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 3682–3690, Red Hook, NY, USA, 2016. Curran Associates Inc. ISBN 9781510838819.

Lisa Lee, Emilio Parisotto, Devendra Singh Chaplot, Eric P. Xing, and Ruslan Salakhutdinov. Gated path planning networks. *CoRR*, abs/1806.06408, 2018. URL http://arxiv.org/abs/1806.06408.

C. Stachniss, D. Hahnel, and W. Burgard. Exploration with active loop-closing for fastslam. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, volume 2, pages 1505–1510 vol.2, 2004.

Gregory J. Stein, Christopher Bradley, and Nicholas Roy. Learning over subgoals for efficient navigation of structured, unknown environments. In Aude Billard, Anca Dragan, Jan Peters, and Jun Morimoto, editors, *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 213–222. PMLR, 29–31 Oct 2018. URL http://proceedings.mlr.press/v87/stein18a.html.

Aviv Tamar, Sergey Levine, and Pieter Abbeel. Value iteration networks. *CoRR*, abs/1602.02867, 2016. URL http://arxiv.org/abs/1602.02867.

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'*, pages 146–151, 1997.

Wei Yang, Xiaolong Wang, Ali Farhadi, Abhinav Gupta, and Roozbeh Mottaghi. Visual semantic navigation using scene priors. In *ICLR*, 2019.