2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

RGBD-based Image Goal Navigation with Pose Drift: A Topo-metric Graph based Approach

Shuhao Ye[1*], Yuxiang Cui[1*], Hao Sha[1], Sha Lu[1], Yu Zhang[1], Rong Xiong[1], Yue Wang[1†]

***Abstract*— Image-goal navigation in unknown environments with sensor error is of considerable difficulty for autonomous robots. In this paper, we propose a drift-resisting topo-metric graph to map the environment and localize the robot using only relative poses. The error-sharing mechanism under this representation effectively reduces the impact of accumulated drifts commonly encountered in navigation tasks. A Reinforcement Learning based policy was proposed for sub-goal selection on this topo-metric graph, which improves navigation efficiency by handling task-driven features taking both image correlation and topological layout into account. We adopt a modular system design with this map representation and graph policy, leaving the low-level motion planning problems to classical controllers for better stability and generalizability. Experimental results demonstrate that our method can achieve robust navigation performance in a variety of unknown environments and even 50% higher success rate over existing methods in complex environments with odometry drift.**

I. Introduction

Image-goal navigation has attracted much attention for its wide applications ranging from domestic service to search-and-rescue missions. In these tasks, the robot is required to navigate to locations represented by high-level instructions like target images instead of precise position targets. Task difficulty even increases for robots deployed in unknown environments, in which the robots should explore and map the surrounding environments while approaching the target location with the image instruction.

Existing methods can be divided into two categories, end-to-end pipeline and modular pipeline. Compared to end-to-end methods that directly map sensor inputs to robot control commands, modular learning-based pipelines consist of units like mapping, sub-goal selection, and local navigation. Recent research results show that modular systems normally achieve better performance compared to end-to-end methods in both generalizability and stability, and learning-based techniques can further improve these methods in practical applications [1] [2].

Most modular systems rely on an expensive global consistent mapping that requires perfect global pose estimation, or with only mild noise. However, odometry drift commonly exists in actual deployment, especially for indoor navigation where GNSS signals are lost. The drifted odometry could lead to severe accumulated global map error as the navigation proceeds. To address the drift in mapping, an effective method is to replace the dense semantic map with a sparse topological graph that distributes the localization error across the edges. However, most existing topological approaches still utilize global pose for node-level localization [3], which weakened the role of the topological map in drift resisting. And works that build topological maps using only image correlations instead of global constant maps usually need an explicit off-line human-controlled mapping phase [4] [5], which is not feasible in unknown environment deployments.

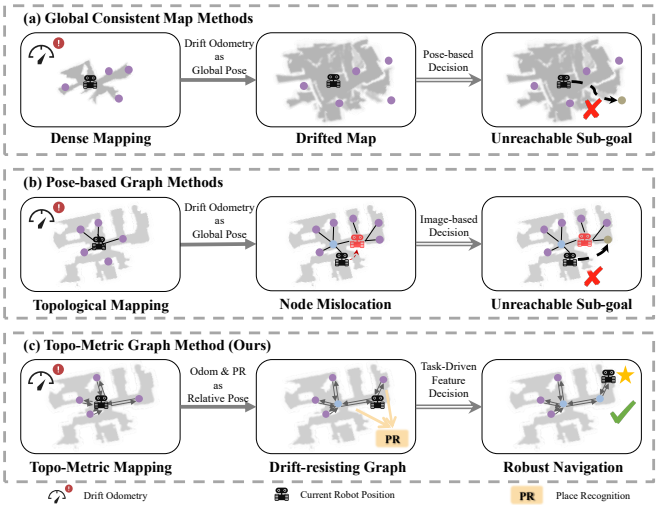


Fig. 1: Comparison about three approaches' pipelines. Our method utilizes PR and drifted odometry to estimate relative pose, avoiding the accumulated errors in global pose. It uses target-driven features that contain image-related and topological information for robust and goal-oriented navigation.

To deal with the problems mentioned above, the key issue is to design a drift-resisting topological graph suitable for autonomous navigation in an unknown environment. How to build this topological map and how to use it for downstream navigation are the main problems that have to be solved in this framework. To build the map, the robot should be able to locate itself at the node level and recognize the relative positional relationship with other nodes during the exploration. To efficiently use this topological map, a sub-goal selection mechanism should be designed to help the robot navigate in the graph paying attention to both image correlation and topological layout.

In this paper, we propose a modular RGBD-based image-goal navigation system using pure onboard odometry without accessing the global pose to reduce the impact of sensor drift. A topo-metric graph is designed as the map representation

This work was supported by the National Key R&D Program of China under Grant No. 2022YFB4701502.

*Indicates equal contribution. †Corresponding author: Yue Wang (email: wangyue@iipc.zju.edu.cn).
The State Key Laboratory of Industrial Control and Technology, Zhejiang University, Hangzhou, P.R. China.

with both global topology and local metrics. Local occupancy grid maps are maintained in explored nodes on the global graph for frontier detection, providing hints for exploration value assessment. The nodes are connected to each other by relative pose so that the sensor drift errors can be borne among all the nodes. To help loop detection and topological mapping with relative poses, we employ the classical method for place recognition. We adopt a modular design with the intermediate representation being the topo-metric map built on the fly, leaving the decision to topo-level graph RL policy, and leaving the motion to mature planning techniques. Rather than making decisions using only poses or images, our RL-based sub-goal selection policy learns task-driven features taking both image similarity and topological features into account, leading to non-myopic navigation performance. Benefiting from the modular learning-based design, the system can be naturally trained as a whole. Experimental results show that our model can achieve a higher success rate and SPL than existing baselines in highly realistic unseen scenes in the Habitat Simulator. In summary, this paper presents the following contributions:

- We propose a drift-resisting topo-metric graph as the map representation in globally topological and locally metric style, reducing the impact of sensor error with the help of relative poses.
- We propose a modular RGBD-based image-goal navigation system, aided by a topo-level graph RL policy for sub-goal selection, in which the policy handles task-driven features taking both image correlation and topological layout into consideration.
- We validated our system performance in highly realistic environments. The results indicate that our model has improved the performance of SOTA models by approximately 50% in success rate.

II. Related Works

A. Space Representation

The classical way to record unknown environments is a global metric map. Researchers [6] [7] utilize the depth images or laser data to maintain a global consistent metric map, which can be directly used in decision-making and motion planning. However, this approach requires the accuracy of both depth values and localization. Once an error occurs, it causes map noise, which in turn affects navigation performance.

A portion of the recent works map the environment by a learned module. This kind of representation benefits by learning, having a higher frame rate, and is less brittle to sensor noise. A simple approach is to add memory modules in neural networks like LSTMs [8] or transformers [9], which record the environment and trajectories implicitly. However, besides the inexplicability, their performance in navigation or exploration is also weak. Thus, a map is still needed for more explicit learning for neural networks. The learned maps are constructed by both odometry readings and images. They can be either a metric version [10] or a fuzzy version which can only be exploited by neural network [11]. The learning approaches can accept only RGB images as input, rather than RGBD, thus getting rid of the impact of depth value errors. However, they still need the odometry reading for pose transformation, which means the odometry drift is still directly related to their navigation performance.

Topological map is another space representation that can easily scale with the environment. The key point about topological maps for navigation is how to perform node-level localization and update the graph. Some approaches [3] simply use the global pose, which can help calculate the distance between nodes easily. While [4] [5] don't rely on a GPS signal, but need a pre-built topological map or a human-controlled navigation video. In this work, we combine the topological map with the classical metric map. Applying place recognition for node-level localization and maintaining local maps in each node for decision-making and motion planning.

B. Navigation Approach

Navigation in an unknown environment always needs a period of goal-oriented exploration, especially for image-goal navigation. The reactive strategies [12] don't model this process explicitly. They just output an action based on current and goal observation. The frontier-based strategies [7] [13] [14] utilizes the metric map to detect frontiers, which are the boundaries of unknown and free space. A rule is needed for choosing a sub-goal from all frontiers. Experimental results in [14] indicate that choosing the closest frontier performs better than choosing a random one. The path planner based on the metric map then plans a path from the current place to the chosen frontier. Learning-based strategies predict a sub-goal directly from the map [10] [15] or choose a sub-goal from the topological map based on the image similarity between nodes and the goal [3]. In this work, we integrate both frontier-based and learning-based strategies. We detect frontiers in each explored node as sub-goal candidates. And train a graph policy network to select one of them for efficient goal-oriented exploration and navigation.

III. Task Setup

We consider the image-goal navigation with RGBD setting, in which a policy $\pi(a_t|s_t, I_g)$ should be designed to guide the robot to the goal position represented by image I_g. At the beginning of an episode, the robot is initialized at a given pose in an unknown environment. It receives a panoramic RGB image I_g taken at the goal position simultaneously. Then, at each step, the robot takes an action a_t and receives a series of observations s_t, which includes a panoramic RGB image and a panoramic noisy depth image taken at the current pose and a drifted odometry sensor reading.

IV. Methods

We adopt modular architecture in our model. The core component is the topo-metric graph, which is maintained and utilized by 4 modules: The **Graph Update** module builds

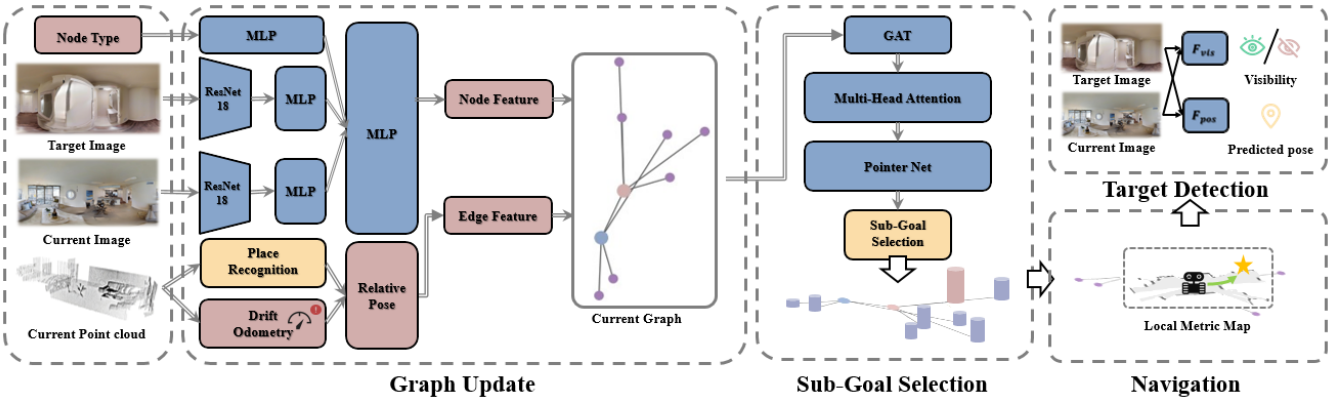


Fig. 2: Model Overview. This figure shows how the four modules of our model work together as a whole. The Graph Update combines both Place Recognition and drifted odometry readings to get a relative pose and update graph (In this figure, the red, blue, and purple nodes in the topological graph represent the current node, explored nodes, and frontier nodes, respectively). The Sub-goal Selection utilizes node features consisting of RGB and node type information, and edge features contain relative pose information as the input for its graph-based policy network. The output is the probability distribution over each sub-goal candidate. The Navigation receives a sub-goal as input and navigates the robot to the sub-goal location, utilizing the node connectivity of the topological map and the local metric maps within each explored node. Target Detection evaluates whether the image-goal appears in real-time. If so, it further estimates the relative pose. Once successfully detected, the input for Navigation is reset to the location of the image-goal.

the global topological map G_t, which includes two types of node, explored node N_e and frontier node N_f. The **Sub-goal Selection** module selects a sub-goal from all frontier nodes. It takes the whole graph G_t as input and outputs a N_f's index. The **Navigation** module guides the robot from the current node to the given sub-goal. The **Target Detection** module compares the similarity of current and goal images to judge if the goal appears. Next, we will first introduce the topo-metric graph-based map followed by detailed explanations of the aforementioned four modules.

A. Topo-metric Graph Based Map

Our topo-metric map consists of a global topological map and local node-wise metric maps. Each N_{ei} stores a frame of point clouds p_t^3 and current RGB features f_i, goal RGB features f_g, and maintains a local metric map m_i. In detail, the p_t^3 and f_i are derived from RGBD observation at the time when this node is created. We unproject the depth image into 3D point clouds to get p_t^3 and pass the RGB image into a Resnet18 encoder to get f_i. The f_g is the RGB feature of the goal image, which is also derived by the same Resnet18 encoder. We add f_g into each N_{ei} for comparison between the current image and the target image. Each N_{fj} contains f_i-sized zero vector and only connects to a N_{ei}, which indicates a frontier at a specific position relative to the N_{ei}, available for exploration. The node feature for N_{ei} is $F_i = \{f_i, f_g, b, 0\}$, where b is a boolean variable indicating whether the node is the current node, 0 means it's not a frontier node. The node feature for N_{fj} is $F_j = \{\mathbf{0}, \mathbf{0}, 0, 1\}$.

Nodes in the topological map are connected to each other by bidirectional edges which store the relative pose of this pair of nodes. In detail, we design the edge feature as $F_{ij} = (d_{ij}, \rho_{ij}, \alpha_{ij})$, where d_{ij} means the relative distance between two nodes, ρ_{ij} is the direction of j in the perspective of i, and α_{ij} is the angle between the current node i's heading direction and the neighboring node j's heading direction. Do note that $F_{ij} \neq F_{ji}$.

B. Graph Update

Node-level localization. To build a topological map, we first need to design an algorithm about when to create a new node and how to localize the robot on the graph. In this paper, we apply the RING method [16] [17] [18], a place recognition (PR) module, to tackle both issues above . The RING takes point clouds of two scenes as input and outputs the similarity and relative pose estimation between them. At every step, the robot first extracts the point clouds from current depth observation d_t and compares it with all the existing N_e's point clouds respectively. If the maximum similarity score M_s is above the threshold, then the robot just locates itself to the node that M_s points to and gets the relative pose. Otherwise, the robot creates a new N_{ei}. To initialize or update the local probabilistic occupancy grid map m_i, it flattens the three-dimensional point clouds p_t^3 into fake two-dimensional point clouds p_t^2. After that, we update the probabilistic occupancy grid map by ray tracing at the time when the robot is localized in this node.

Intra-node localization. In our work, the robot gets both drifted odometry readings and PR-estimated pose changes. Although PR itself can output the relative pose we need, the one accumulated by drifted odometry readings is more accurate within a certain step range. Once it's localized in a new node, the robot gets the relative pose estimated by PR as an initial value and updates it by odometry readings. However, if the accumulated relative pose shows a significant discrepancy with the pose estimated by PR in two consecutive frames, the robot will re-estimate the relative pose using PR as a new initial value and continue to update it with odometry readings.

Frontier detection and noise filter. The above process allows the robot to maintain mapping and localization of

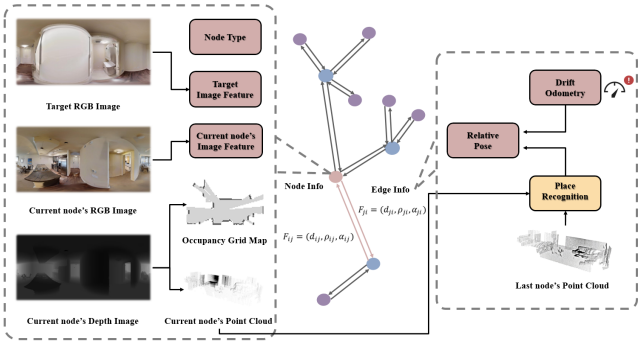


Fig. 3: Map Structure. In this work, each explored node (red or blue circles) stores three different kinds of data: a frame of point clouds, a local probability occupancy grid map, and RGB features which implies both the current node's and goal's RGB information. The frontier node (purple circles) stores nothing. Node type is a two-dimensional boolean vector. Node features are concatenated from RGB features and node type vector. Edges are bidirectional. Each edge stores a three-dimensional vector indicating the relative pose between two adjacent nodes.

the unknown environment while moving, but it still needs to know where to explore. We utilize the two-dimensional point clouds p_t^2 and local map m to detect frontiers as sub-goal candidates and add them to the topological map, connected with the current node. The classical frontier detection algorithms [7] are easily affected by map noise and have low efficiency in dense map traversal. We propose to use p_t^2 for frontier detection at every step. Moreover, we employ a similar principle for noise point filtering. The distinction lies solely in the varying criteria for judgment. In detail, for each point $p_t^2[k]$ in p_t^2, we calculate:

$$D_1 = \left|d(p_t^2[k]) - d(p_t^2[k-1])\right| \tag{1}$$

$$\begin{aligned} D_2 = (&\left|d(p_t^2[k-1]) - d(p_t^2[k-2])\right| \\ &+ \left|d(p_t^2[k-2]) - d(p_t^2[k-3])\right| \\ &+ \left|d(p_t^2[k-3]) - d(p_t^2[k-4])\right|)/3 \end{aligned} \tag{2}$$

where $d(*)$ means the depth value of $*$. If $D_1 > cD_2$, it indicates the presence of a cross-section between this point and the previous one. Here, we take c to be 2.5. Similarly, we compare $p_t^2[k]$ with next four points, and calculate D_3 and D_4. Only if $D_1 \leq cD_2$ and $D_3 \leq cD_4$, the point lies on a continuous plane. If $D_1 > cD_2$ and $D_3 > cD_4$, the point is a noise point. Otherwise, it's a frontier. The robot first performs noise point filtering and then proceeds with frontier detection. A frontier will be eliminated if the corresponding regions in its neighboring node's grid map become known. The frontiers close to an existing one within a grid map will be merged together, and further removed together if any of them become known in case of repeated exploration of the same location.

C. Sub-goal Selection

We build our Sub-goal Selection policy $\pi(a_t|s_t, I_g)$ with an attention-based neural network. The input is the current global topological map G_t represented by node features F_i and edge features F_{ij}. The action space is the set of frontier nodes in G_t. The policy network consists of a GAT as graph encoder and a Muti-Head Attention as decoder, followed by a Pointer Net [19] to produce the final output. The decision output is the probability distribution over each N_f. The critic is a Graph Q Net which shares a similar structure with the policy network, with the exception of the final layer. At runtime, the robot just chooses the N_f corresponding to the highest probability as a sub-goal.

To guide the policy optimization, we have designed a reward function taking approaching goal, detecting goal, and step cost into account:

$$R(s_t) = c_1 \cdot R_a(s_t) + c_2 \cdot R_d(s_t) - c_3 \cdot R_c(s_t) \tag{3}$$

In particular, the agent gets $R_a(s_t)$ for getting closer to its goal and gets $R_d(s_t)$ for detecting the goal by Target Detection Module. To encourage the robot to choose more path-efficient frontiers, it gets penalized with $R_c(s_t)$, where $R_c(s_t)$ is the step cost for arriving at the chosen frontier.

D. Navigation

The navigation module is responsible for navigation between adjacent nodes. It should be noted that by reasonably designing the size of the local metric map and the threshold for M_s, the adjacent nodes can always be included within the local metric map of the current node. Therefore, this module essentially functions as an intra-node navigation module. With the help of m, we simply apply the RRT* algorithm as the path planning method, and the DWA algorithm as the local trajectory planning method.

E. Target Detection

Inspired by [3], we design two networks F_{vis} and F_{pos} for target detection. The F_{vis} is trained to predict whether the goal position is visible from the current position, while the F_{pos} is trained to estimate the relative pose between them. Both of them take an RGB image pair as input.

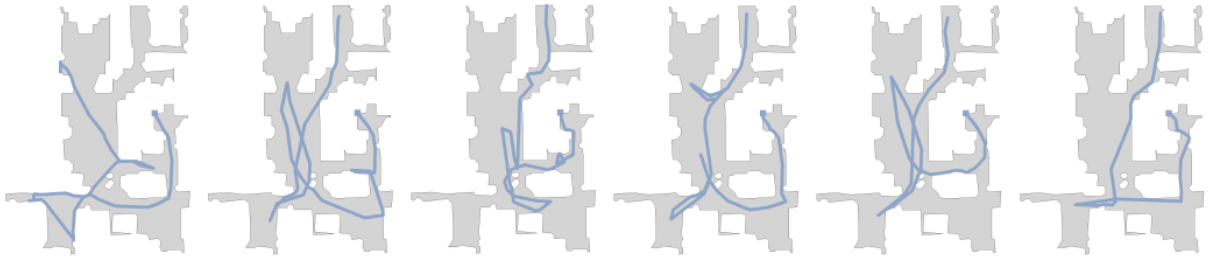
V. Experiments

A. Implementation Details

Task setup. We conduct all experiments in Habitat Simulator [20] coupled with Gibson Dataset [21]. The 86 scenes in Gibson Dataset are split into sets of 68/4/14 scenes for train/val/test. The maximum episode length is 500 steps. An episode is considered successful if, by the end, the robot's distance to the goal is within 1m; otherwise, the navigation is considered unsuccessful. The episodes are split into easy, medium, and hard modes as in [3].

The robot is equipped with a noisy actuator [10] that can move forward approximately 25 cm, turn left or right by about 10 degrees, and stop. Additionally, we introduce RedWoodNoise [22] to our depth observation. For handling odometry drift, we employed the drift probability distribution as estimated in [23].

Training Details. There are three learning modules in our work: the Sub-goal Selection module, the Target Detection module, and the Resnet18 encoder. We train the Target Detection module using the same method as in [3]. The



(a) OURS-GM Total step=500 (b) OURS-RP Total step=314 (c) OURS-NM Total step=306 (d) OURS-NF Total step=287 (e) OURS-NP Total step=219 (f) OURS Total step=153

Fig. 4: Visualization of navigation trajectories of all ablations. The navigation starts at the blue square, and the goal is at the room above, as shown in (b) to (f). Our method reaches the goal with minimum steps. It only explored one additional room apart from the target.

TABLE I: Navigation performance of our model and all ablations.

	Easy		Medium		Hard	
	SR	SPL	SR	SPL	SR	SPL
OURS-RP	0.86	0.64	0.47	0.24	0.31	0.15
OURS-NP	0.90	**0.69**	0.63	0.38	0.49	0.29
OURS-NF	**0.91**	0.65	0.60	0.31	0.45	0.23
OURS-GM	0.88	0.61	0.42	0.25	0.18	0.12
OURS-NM	**0.91**	0.66	0.56	0.23	0.37	0.15
OURS	**0.91**	0.68	**0.69**	**0.52**	**0.60**	**0.44**

image encoder is pre-trained in the form of an autoencoder. We take 100 images in each environment to construct the training dataset. After the pre-training, we proceeded to train the image encoder and Sub-goal Selection online. They are jointly trained by the SAC algorithm [24]. We generate 500 episodes in each environment.

B. Ablation Study

We first conduct experiments to analyze the effectiveness of our designed modules. We consider five ablations:

OURS-RP. Replace the Sub-goal Selection policy with a random policy.

OURS-NP. Replace the Sub-goal Selection policy with Semantic Score Prediction (F_S) function in [3].

OURS-NF. Sub-goals are distributed around the circumference of the robot with a radius of 3m [3]. We just set the distance between every frontier node and their connected explored nodes to be 3m.

OURS-GM. Replace the local metric map with a global metric map, while still keeping the remaining topological structure. The robot uses the global point clouds, which are composed of point clouds of all explored nodes after transformation, to perform PR.

OURS-NM. We remove the frontier merge process from the original model. It means in OURS-NM, frontier nodes connected to different explored nodes may point to the same region and they cannot be eliminated together.

The results are shown in Table I. It indicates that all ablations' performance is similar in easy mode. However, As the difficulty increases, the performance of all ablations decreases to varying degrees. We analyze the ablation experiment from four perspectives:

Sub-goal selection. OURS-RP and OURS-NP are two ablations about the sub-goal selection module. For OURS-RP, it shows relatively weak performance because of the lack of high-level navigation experience, thus wasting lots of steps in repeated paths or exploring regions away from the goal. OURS-NP compares the image information at frontier nodes with the goal image. The robot keeps taking greedy actions to nodes with similar images in this module. But it takes much more detour than OURS because of the lack of information of the topological structure, which is related to the path cost from the current node to the sub-goal.

Sub-goal definition. OURS-NF sets fixed distance frontiers as frontier nodes, which leads to two problems: eliminating the frontier within $r = 3m$ and truncating the frontier to locate at $3m$ away. The former problem may result in unexplored regions and the latter may result in longer path costs. Results show that the short horizon of the action space may also lead to trajectory oscillation, indicating that the classical definition of the frontier is a good choice for more efficient exploration.

Metric map. For OURS-GM, the performance drop may be due to the localization error, which increase as the navigation proceeds. Because it only maintains a global metric map, the relative pose actually becomes a global pose. The accuracy of PR drops as the scale of the global point clouds expands. Hence, the global map becomes noisy and thus influences modules like Frontier Detection and Navigation. OURS-GM performs poorly in hard mode. It shows the necessity for building local metric maps with relative poses, rather than a single global metric map.

Frontier merge. For OURS-NM, the robot may revisit the same region several times caused by different frontier nodes representing the same locations. This greatly reduces the efficiency of exploration. We can see that compared to other ablations, the SPL/SR of OURS-NM is much lower, which is below 0.5 in medium and hard modes.

C. Drift Resistance

To analyze the odometry drifts on navigation success rates, we compared the SR of three methods in hard mode under different levels of odometry drifts, as shown in Fig 6. We tested each method 200 times at each drift level. OccAnt [23] is a model built on top of Active Neural Slam [10]. It predicts the occupancy in the unseen region based on current RGBD observation. We adapt it to the image-goal navigation task by

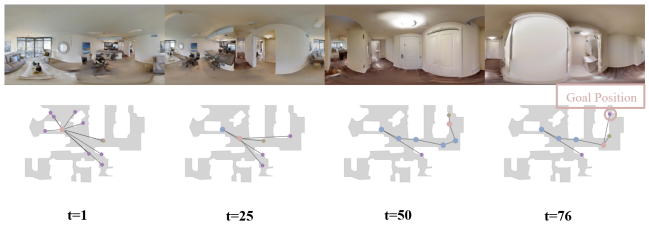


Fig. 5: Navigation visualization. We show the trajectory of our model in a test episode. The top shows the current panoramic RGB image and the bottom shows the topological map at that time.

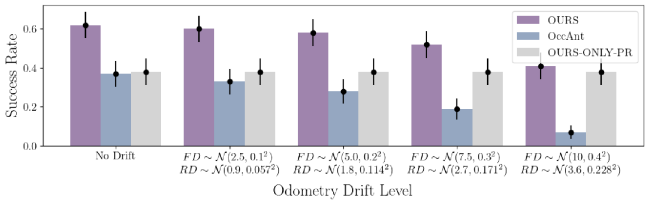


Fig. 6: The success rate in hard mode navigation under five levels of drifts. We present three models here. The FD and RD means forward and rotation drift respectively. The former uses the unit of centimeters, while the latter uses the unit of degrees.

adding our Target Detection module to it. OURS-ONLY-PR is a lower bound of OUR method, it only uses PR for relative pose estimation. The results indicate that OUR method is more drift-resisting than neural slam-based methods. The SR of OURS drops by 21% from no drift to maximum drift, while OccAnt's SR drops by 30%.

D. Comparative Study

We further provide a thorough evaluation of the performance of our method by comparing it with several baselines. OccAnt is adapted and tested by us and others are taken from [3], with pose noise setting rather than drifted odometry. We test our model with both drifted odometry setting and pose noise setting, named separately as OURS+Drift and OURS+Pose Noise. The results can be seen in Table II.

ResNet + GRU + IL. A simple baseline consisting of ResNet18 image encoder and a GRU-based policy trained with imitation learning.

Target-Driven RL [12]. A vanilla baseline trained with reinforcement learning.

Metric Spatial Map + RL [25]. An end-to-end RL model that uses geometric projections of the depth image to create a local map and passes it to the RL policy.

Metric Spatial Map + FBE + Local. A hand-designed baseline creates a map using depth images and uses FBE [7] to greedily explore the nearest unexplored frontier.

Active Neural SLAM (ANS) [10]. An exploration model based on metric maps, adapted to navigation.

Occupancy Anticipation (OccAnt) [23]. An exploration model built on top of ANS, adding an occupancy anticipation module to predict unseen regions. Adapted to navigation.

Neural Topological SLAM (NTS) [3]. Model based on neural SLAM and endowed with a topological graph for long-term planning.

TABLE II: Navigation performance of our model and baselines.

	Easy		Medium		Hard	
	SR	SPL	SR	SPL	SR	SPL
ResNet+GRU	0.72	0.32	0.16	0.09	0.05	0.02
Target-driven RL	0.68	0.28	0.21	0.08	0.09	0.03
Metric+RL	0.69	0.27	0.22	0.07	0.12	0.04
Metric+FBE+RL	0.77	0.56	0.36	0.18	0.13	0.05
ANS	0.76	0.55	0.40	0.24	0.16	0.09
OccAnt	0.83	0.61	0.47	0.28	0.33	0.17
NTS	0.87	0.65	0.58	0.38	0.43	0.26
OURS+Pose Noise	**0.91**	**0.69**	**0.72**	**0.53**	**0.62**	**0.45**
OURS+Drift	**0.91**	0.68	0.69	0.52	0.60	0.44

The simple learning-based end-to-end models, namely ResNet+GRU and Target-driven RL, get quite low SR and SPL in medium and hard modes. This is because they don't have a module to memorize where has been explored. Thus, with the increase in path length, their actions may tend to be repetitive without having a clear orientation.

The metric map-based methods, namely Metric+RL, Metric+FBE+RL, ANS, and OccAnt, perform better than end-to-end methods. However, their navigation SR is still low in hard mode. In the case of the Metric+RL approach, where only a local metric map is maintained, the performance drop mainly lies in the lack of global memory. The Metric+FBE+Local approach, on the contrary, maintains a global map but may suffer from it because of the noisy global map caused by pose noise. The ANS and OccAnt approaches, which can be regarded as learning versions of the metric-based approach, cannot reach satisfying results in hard mode, either. Its drawbacks are similar to the above two.

The NTS approach improves navigation performance significantly, especially in hard mode, while our approach achieves even better performance to 62% SR and 0.45 SPL in hard mode, improving the performance of NTS by approximately 50%. In easy and medium modes, our approach also ranks first among all methods above.

Pose noise and drifted odometry. The drifted odometry is different from the pose noise. The former's error accumulates along with time while the latter's error doesn't. OURS+Drift exhibits the performance of our model with a drifted odometry setting. Compared to other baselines with noisy pose, it still holds the advantages, and there is not a substantial performance decline when compared to OURS+Pose Noise.

VI. Conclusions

This paper presents a modular model for RGBD-based image-goal navigation problem, with drifted odometry. We have designed a topo-metric map representation that can record both topological and geometric features of the unknown environment and be updated automatically. A graph-based Sub-goal Selection module is built to utilize the features in the topo-metric map, path-efficient and goal-oriented sub-goals can be chosen with the help of it. We leave the downstream navigation to mature algorithms for its stability. We compared our model's performance with other baselines. The results show that we improved the navigation performance by almost 50%, even with drifted odometry settings.

REFERENCES

[1] T. Gervet, S. Chintala, D. Batra, J. Malik, and D. S. Chaplot, "Navigating to objects in the real world," *Science Robotics*, vol. 8, no. 79, p. eadf6991, 2023.

[2] R. McAllister, Y. Gal, A. Kendall, M. Van Der Wilk, A. Shah, R. Cipolla, and A. Weller, "Concrete problems for autonomous vehicle safety: Advantages of bayesian deep learning," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, International Joint Conferences on Artificial Intelligence Organization, 2017.

[3] D. S. Chaplot, R. Salakhutdinov, A. Gupta, and S. Gupta, "Neural topological slam for visual navigation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12875–12884, 2020.

[4] D. Shah, B. Eysenbach, G. Kahn, N. Rhinehart, and S. Levine, "Ving: Learning open-world navigation with visual goals," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 13215–13222, IEEE, 2021.

[5] N. Savinov, A. Dosovitskiy, and V. Koltun, "Semi-parametric topological memory for navigation," *arXiv preprint arXiv:1803.00653*, 2018.

[6] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

[7] 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.

[8] Y. Lyu, Y. Shi, and X. Zhang, "Improving target-driven visual navigation with attention on 3d spatial relationships," *Neural Processing Letters*, vol. 54, no. 5, pp. 3979–3998, 2022.

[9] K. Fang, A. Toshev, L. Fei-Fei, and S. Savarese, "Scene memory transformer for embodied agents in long-horizon tasks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 538–547, 2019.

[10] D. S. Chaplot, D. Gandhi, S. Gupta, A. Gupta, and R. Salakhutdinov, "Learning to explore using active neural slam," *arXiv preprint arXiv:2004.05155*, 2020.

[11] S. Gupta, J. Davidson, S. Levine, R. Sukthankar, and J. Malik, "Cognitive mapping and planning for visual navigation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2616–2625, 2017.

[12] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi, "Target-driven visual navigation in indoor scenes using deep reinforcement learning," in *2017 IEEE international conference on robotics and automation (ICRA)*, pp. 3357–3364, IEEE, 2017.

[13] C. Dornhege and A. Kleiner, "A frontier-void-based approach for autonomous exploration in 3d," *Advanced Robotics*, vol. 27, no. 6, pp. 459–468, 2013.

[14] D. Holz, N. Basilico, F. Amigoni, and S. Behnke, "Evaluating the efficiency of frontier-based exploration strategies," in *ISR 2010 (41st International Symposium on Robotics) and ROBOTIK 2010 (6th German Conference on Robotics)*, pp. 1–8, VDE, 2010.

[15] Q. Wu, J. Wang, J. Liang, X. Gong, and D. Manocha, "Image-goal navigation in complex environments via modular learning," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6902–6909, 2022.

[16] S. Lu, X. Xu, H. Yin, Z. Chen, R. Xiong, and Y. Wang, "One ring to rule them all: Radon sinogram for place recognition, orientation and translation estimation," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2778–2785, IEEE, 2022.

[17] S. Lu, X. Xu, L. Tang, R. Xiong, and Y. Wang, "Deepring: Learning roto-translation invariant representation for lidar based place recognition," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1904–1911, IEEE, 2023.

[18] X. Xu, S. Lu, J. Wu, H. Lu, Q. Zhu, Y. Liao, R. Xiong, and Y. Wang, "Ring++: Roto-translation-invariant gram for global localization on a sparse scan map," *IEEE Transactions on Robotics*, 2023.

[19] O. Vinyals, M. Fortunato, and N. Jaitly, "Pointer networks," *Advances in neural information processing systems*, vol. 28, 2015.

[20] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, *et al.*, "Habitat: A platform for embodied ai research," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9339–9347, 2019.

[21] F. Xia, A. R. Zamir, Z. He, A. Sax, J. Malik, and S. Savarese, "Gibson env: Real-world perception for embodied agents," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 9068–9079, 2018.

[22] S. Choi, Q.-Y. Zhou, and V. Koltun, "Robust reconstruction of indoor scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5556–5565, 2015.

[23] S. K. Ramakrishnan, Z. Al-Halah, and K. Grauman, "Occupancy anticipation for efficient exploration and navigation," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pp. 400–418, Springer, 2020.

[24] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*, pp. 1861–1870, PMLR, 2018.

[25] T. Chen, S. Gupta, and A. Gupta, "Learning exploration policies for navigation," *arXiv preprint arXiv:1903.01959*, 2019.