2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Multiple Visual Features in Topological Map for Vision-and-Language Navigation

Ruonan Liu[1], Ping Kong[2] and Weidong Zhang[1]

***Abstract*— Vision-and-Language Navigation (VLN) in continuous environments aims to navigate robot agents in unseen environments following natural language instructions. The majority of existing approaches rely on constructing semantic maps or topological maps to record information. However, semantic maps overlook the detailed information of objects and the correspondence among views during navigation, while topological maps lack the spatial representation between entities. To address these limitations, we propose a novel visual feature representation method for continuous VLN, called Multiple Visual Features in Topological Map (MV-Topo). MV-Topo utilizes three distinct visual encoders to extract visual features, which are integrated in the dynamically generated topological map. These fused features actively participate in the subsequent cross-modal planning to derive a long-term path towards a subgoal, effectively guiding the agent to reach the final location. We experimentally demonstrate the effectiveness of our approach and achieve competitive results on the full VLN-CE test splits. Notably, our method outperforms the state-of-the-art by 3.5% in terms of the Navigation Error (NE) metric, indicating that the utilization of multiple visual features significantly enhances the agent's perception of semantic targets.**

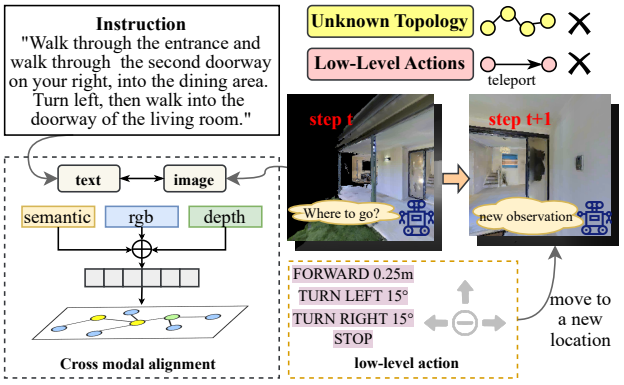

Fig. 1. VLN-CE requires an agent to navigate through a continuous environment following a given natural language instruction. At each step, our proposed model incorporates three distinct visual features into the topological map to facilitate the alignment between the instruction and the environment. Subsequently, the agent executes low-level actions to move towards a new location, where it can gather new observations.

I. INTRODUCTION

The desire for an interactive assistant robot capable of following human instructions has been a longstanding pursuit. To explore this area, Vision-and-Language Navigation (VLN) [1] has been introduced. This task requires a robot agent to reach the target location based on a natural language instruction and the surrounding environment. Initial efforts focused on the discrete VLN setup [2], [3], [4], [5], a simplified version where agents only need to select predefined waypoints in an environment with a navigation graph.

However, discrete VLN task relies on many unrealistic assumptions, so Vison-and-Language Navigation in Continuous Environments (VLN-CE) [6] has been proposed. VLN-CE provides a more challenging and realistic setting by removing the navigation graph and requiring the agent to navigate freely in the environment using low-level actions. Due to the large domain gap between the two tasks, the performance of an agent that excel in discrete VLN will be drastically degraded in continuous VLN, thus numerous methods are dedicated to narrowing the gap [7], [8], [9], [10]. Most current approaches utilize the waypoint predictor [10], decomposing the task into waypoint generation, cross-modal planning, and navigation control. This method is much clearer and more straightforward compared to an end-to-end action prediction directly from instructions and environment.

Existing approaches to VLN-CE often rely on constructing maps to record information. [11], [12], [13] construct self-centered semantic maps during navigation and perform cross-modal alignment to align semantic maps and instructions. Although these approaches strengthen the agent's semantic perception and spatial reasoning, they are prone to confusion when multiple objects of the same kind are present. Additionally, they are limited in correspondence among views in continuous environments, which makes it easy to forget previously visited locations and actions during long-horizon navigation. To address the problem of historical memory and information storage for long-horizon navigation, topological maps have been employed [14], [15]. Topological maps facilitate structured memory and path recording over long distances. However, they lack spatial representation between objects, and the visual features used in these approaches generally contain only implicit semantic information, failing to represent object attributes such as texture or shape.

In this paper, a novel visual representation for VLN-CE is proposed to solve the above two problems, namely Multiple Visual Features in Topological Map (MV-Topo), as shown in Fig. 1. The proposed method unites the advantages of semantic maps and topological maps. We leverage a pre-trained semantic segmentation model to encode RGB images, receiving features that encompass both fine-grained (e.g., texture, color) low-level features and coarse-grained (e.g., object

[1]Ruonan Liu and Weidong Zhang are with Department of Automation, Shanghai Jiao Tong University, Shanghai, China. ruonan.liu@sjtu.edu.cn, wdzhang@sjtu.edu.cn
[2]Ping Kong is with the College of Intelligence and Computing, Tianjin University, Tianjin 300350, China. kongping@tju.edu.cn
Corresponding author: Weidong Zhang.

type) high-level features. These features are then fused with the RGB and depth features to form visual representations, which are contained diverse image information, stored in the nodes of the topological map. The proposed approach can not only enhance the agent's perception of the landmarks in the environment, but also avoid catastrophic forgetting during navigation.

The proposed method is evaluated on the widely used VLN-CE dataset [6], which is the standard benchmark for continuous VLN environments. Experimental results demonstrate that incorporating multiple visual features in topological maps significantly improves the agent's performance. Compared to the previous works that only utilized simple RGB and depth encodings, MV-Topo reduces Navigation Error (NE) by 3.5% and achieves competitive Success Rate (SR) in test unseen split, which proves the effectiveness of our proposed multiple visual features approach.

II. Related Work

A. Vision-and-Language Navigation

Vision-and-language navigation (VLN) has attracted increasing attention in recent years due to its potential applications. Several tasks and datasets have been proposed to advance research in this field. R2R [1] provides detailed instructions and serves as the most widely used dataset in VLN. Many datasets extend R2R to cater to diverse requirements, such as RxR [16] containing instructions for multiple languages, and R4R [17] with longer paths. VLN-CE [6] removes the unrealistic assumptions in discrete VLN settings and transfers discrete paths to continuous environments through the Habitat Simulator [18]. In addition, [19] extends VLN to aerial scenarios. These tasks necessitate the agent's ability to associate observations with text for decision-making at each step. Early VLN approaches employed LSTM for visual-textual alignment [20], [21], [22]. Subsequent works have utilized attention mechanisms to enhance the correspondence between visual and textual modalities [23], [24], [25], [26] and applied powerful language models such as BERT [27] and Transformer [28] to VLN to obtain improved performance. Another line of works are investigated to build larger training datasets [29], [30] or augment existing data to increase the diversity of environments and instructions [3], [31], [32], [33]. In this work, we focus on improving visual features for VLN-CE.

B. Maps for Visual Navigation

For navigation tasks, it is beneficial to construct various spatial representations to perceive the environment effectively. Previous studies have constructed semantic maps to display semantic information in the environment or predict missing information beyond the agent's field of view [11], [12], [34], [35]. However, when it comes to long-range navigation, constructing static semantic maps disregards the relationships with other views during navigation and can be computationally expensive. On the other hand, topological maps can encode the relationships between different waypoints in the environment, with each node storing unique visual information [5], [14], [15], [36], [37]. Nevertheless, topological maps lack the ability to extract rich object information as well as spatial reasoning. In contrast to these works, our proposed MV-Topo approach avoids the need for extensive resource allocation to build semantic maps and strengthens the ability of topological maps to extract object information.

C. Visual Representations for Navigation

With the rise of Embodied AI, a great variety of models have emerged to generate visual representations suitable for Embodied AI navigation tasks. The large-scale visual-language model CLIP [38] has demonstrated impressive performance in extracting object features and has found widespread application across vision domains [39], [40]. Furthermore, OVRL [41] leverages knowledge distillation to facilitate visual representation learning for indoor navigation. Numerous works are dedicated to obtaining valuable information from visual features. SPTM [42] utilizes visual observations of nodes to determine the furthest reachable waypoints on a topological map. SplitNet [43] predicts depth and reconstructs RGB input using encoded visual features. More recent work, Ego^2-Map [44] proposes a navigation-specific visual representation learning method with semantic map supervision. Most existing studies lack the ability to reason about fine-grained semantic correspondences between specific visual regions in pictures and phrases in text. In this work, we introduce semantic information and extract visual features with different emphases for fusion, thus obtaining comprehensive visual representation which can contribute to efficient cross-modal alignment.

III. Approach

We propose a new visual representation approach MV-Topo to address the issue of sketchy and coarse visual information in topological maps. As illustrated in Fig. 2, the general network architecture is divided into three modules: mapping, planning, and control. In this section, the definition of the VLN-CE task and the individual modules comprising our proposed MV-Topo framework are described and presented in detail.

A. Task Definition

The problem of Vision-and-Language Navigation in Continuous Environments (VLN-CE) is focused in this paper [6]. In this task, an agent is required to navigate in continuous environments based on a natural language instruction, employing a sequence of low-level actions. The action space is discretized and consists of `FORWARD` (0.25m), `TURN LEFT/RIGHT` (15°), and `STOP`. We follow the panoramic VLN-CE settings adopted in previous works [7], [9], [10], [15], which exhibit superior performance. At each step t, the agent has access to panoramic RGB-D observations consisting of 12 RGB images and 12 depth images, uniformly spaced at 30° intervals, i.e., (0°, 30°, ..., 330°). Each image has a 90° HFOV at a resolution of 256 × 256. During a navigation episode, the agent receives a natural language instruction.

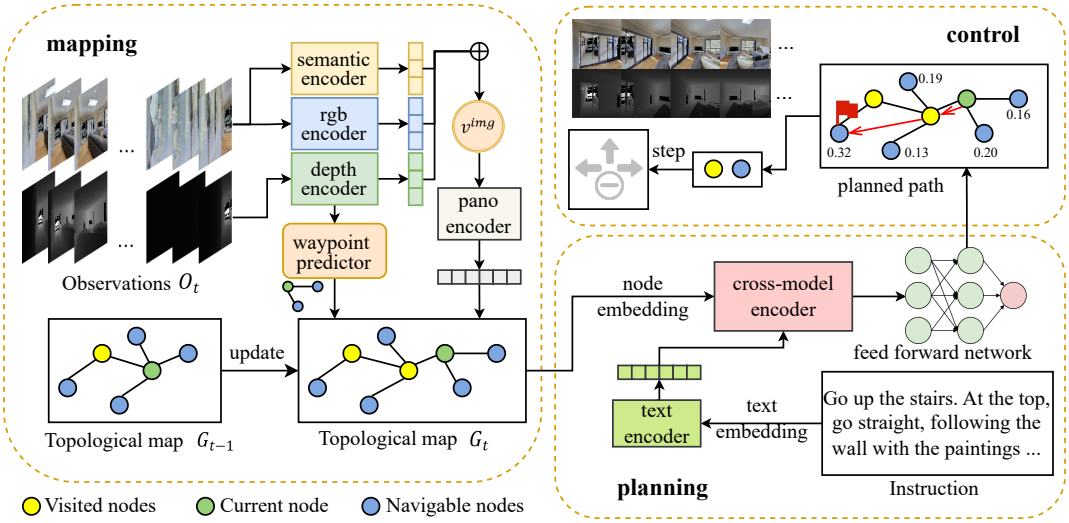

Fig. 2. An overview of the proposed MV-Topo model. First of all, a topology map that is updated at each step is constructed in the navigation episode. This map records the fusion features extracted from visual inputs and establishes connections between reachable nodes. Subsequently, the topological map and the given instruction are fed to the cross-modal planning module to predict a long-term goal. Finally, the control module takes charge of driving the agent using low-level actions, enabling it to execute the long-term plan incrementally until the target location is reached.

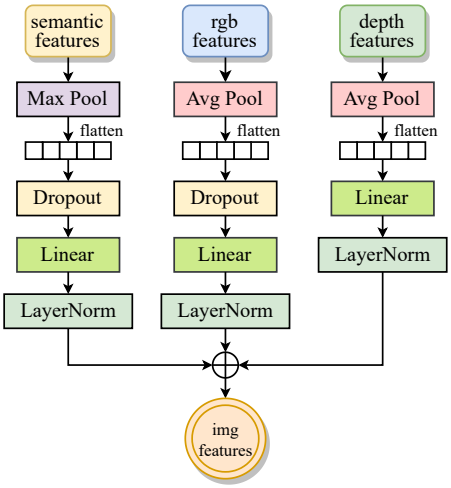

Fig. 3. The fusion process of multiple visual features.

B. Topological Mapping with Multiple Visual Features

In this paper, the agent gradually builds topological maps online during navigation. At each step t, the topological map records information about the currently located node and the nodes that have been observed but not yet visited. The previous step's topology map G_{t-1} is then updated to generate a new topology map $G_t = \{N_t, E_t\}$. The map comprises three types of nodes: visited nodes, the current node, and navigable nodes that have been observed but left unexplored. These nodes store visual and position information, while edges indicate the reachability between nodes, with the weight representing the Euclidean distance.

To generate the reachable nodes for the current node, the waypoint predictor proposed in previous work is employed [10]. The depth features and orientation features from the current position are inputted into the waypoint predictor to get K candidate waypoints. These candidate waypoints are then transformed into nodes within the topological map by a waypoint localization function [15]. Specifically, the function calculates the Euclidean distance of each candidate waypoint from all existing nodes in the map, a node in map is localized if the minimum distance falls below a threshold γ. If the localized node corresponds to a visited node, the input candidate waypoint is discarded, and an edge is added between the current node and the localized node. In the case of a localized navigable node, we update its information by averaging the information from all candidate nodes localized to this navigable node. If no node is localized in the map, the candidate waypoint is added to the map as a new navigable node. Furthermore, to represent the `STOP` action, a 'stop' node is incorporated in the map that is connected to all other nodes.

Multiple visual features. At each time step t, the agent receives the currently observed RGB images and depth images. These images are encoded separately using three different visual encoders to generate distinct visual features. For the RGB images, a RGB encoder and a semantic encoder are employed to get the RGB features f^r and semantic features f^s, respectively. The semantic encoder is based on a semantic segmentation model [12]. Instead of directly using the classification layer output of the semantic segmentation model, we use the feature from the last layer in the model as the semantic encoding. Thanks to the skip connection mechanism in the model, this feature incorporates both fine-grained low-level semantics (e.g., texture, color) and coarse-grained high-level semantics (e.g., object type). Regarding the depth images, we adopt the same depth encoder following previous VLN-CE works [10], [13], [15] to get the depth features f^d. The features fusion process is shown in Fig. 3.

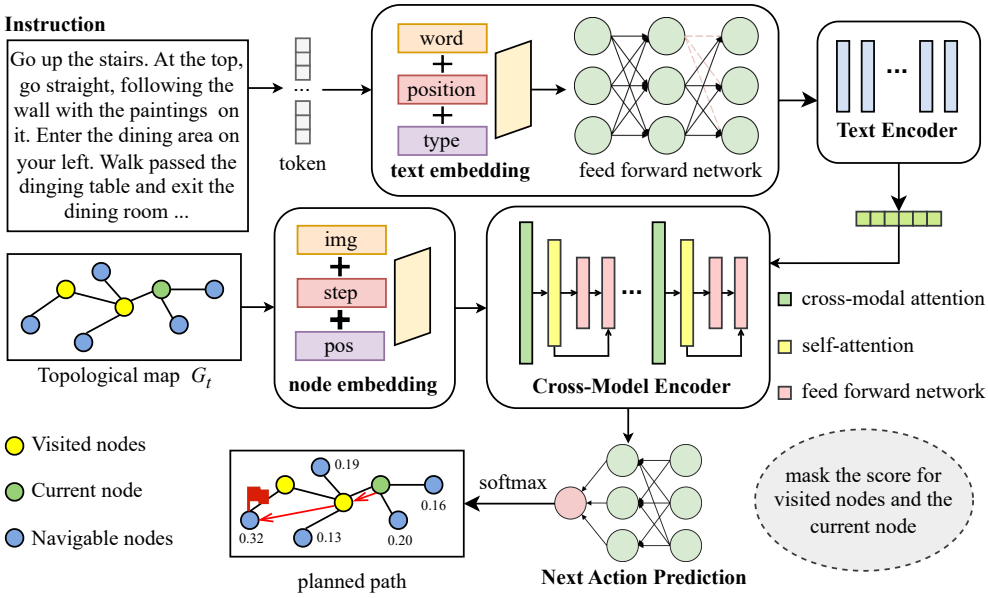

Fig. 4. Overall architecture of the cross-modal planning module.

The fusion of the three visual features is achieved through a simple and effective process. Specifically, to get the key information, RGB features and semantic features perform average pooling and max pooling respectively. These features are then flattened and passed through a Dropout layer and a linear layer. Subsequently, normalization is applied to obtain the processed RGB feature v^r and semantic feature v^s. Similarly, for the depth features, average pooling is performed followed by flatten and normalization to obtain the processed depth feature v^d. The three features are summed to get the final visual features v^{img}:

$$\widetilde{f^r} = AP\left(f^r\right), \widetilde{f^d} = AP\left(f^d\right), \widetilde{f^s} = MP\left(f^s\right) \tag{1}$$

$$\begin{aligned} v^r &= LN\left(Dropout\left(\widetilde{f^r}\right)\right), \\ v^s &= LN\left(Dropout\left(\widetilde{f^s}\right)\right), \\ v^d &= LN\left(\widetilde{f^d}\right) \end{aligned} \tag{2}$$

$$v^{img} = v^r + v^d + v^s \tag{3}$$

where AP denotes average pooling, MP denotes max pooling, LN is the normalization layer, and the superscripts r, d, s respectively represent RGB, depth, semantic.

The relative angles for images can be represent as $o^{img} = (sin\theta^o, cos\theta^o, sin\varphi^o, cos\varphi^o)$, where θ^o and φ^o are the relative heading and elevation angles to the agent's orientation. The fused image features v^{img} along with the orientation information o^{img} are sent to the panoramic encoder to produce the panoramic encoding $pano^{img}$, which includes comprehensive information about vision and location. Due to the panoramic setup, $pano^{img} = \{pano_i\}_{i=1}^{12}$ comprises visual features from 12 different angles. Taking the average of the visual features from all angles as the current node's visual representation $\overline{pano}^{img}$:

$$\overline{pano}^{img} = \frac{\sum_{i=1}^{12} pano_i}{12} \tag{4}$$

For navigable nodes that have been observed but not visited, we select the corresponding visual embeddings of the angles at which they can be observed as their visual features.

By incorporating semantic features, the representation of multiple visual features is enriched, thereby enhancing the agent's capability to capture semantic information within images.

C. Cross-Model Planning

As illustrated in Fig. 4, the cross-modal planning module takes the topology map encoding and instruction representation as inputs and performs reasoning to predict a long-term goal node. Ultimately, the module crafts a topological path plan to the goal.

Text encoding. Based on the experience of prior researches, BERT [27] is adopted as the text encoder to effectively extract the feature vector for each word in the instruction. Firstly, we combine the word embeddings e_T^{word} with the positional embeddings e_T^{pos} and the type embeddings e_T^{type} for individual words in the instruction to get the text embeddings [45] as below:

$$embed_T = e_T^{word} + e_T^{pos} + e_T^{type} \tag{5}$$

$$\hat{w} = Dropout\left(LN\left(embed_T\right)\right) \tag{6}$$

The text embeddings of the instruction with L words are denoted as $\hat{W} = \{\hat{w}\}_{i=1}^{L}$. These embeddings are then fed into multiple BertLayer layers within the text encoder. The final text features contain comprehensive representation of the instruction based on contextual understanding.

Map encoding. To begin with, the fused panoramic visual features $\overline{pano}^{img}$ for each node in the graph are augmented with the time step encoding e_M^{step} and location encoding e_M^{loc}, which embeds the most recent visited time step of the corresponding node, and the relative position with respect to the current node, respectively. The encoding of node n_i ($n_i \in N_t$) is denoted as $\hat{n}_i$:

$$\hat{n} = \overline{pano}^{img} + e_M^{step} + e_M^{loc} \tag{7}$$

Cross-model encoding. The encoded node representations in the map, along with the encoded text features, are jointly fed into the cross-modal encoder similar to LXMERT [45]. Within each layer of the cross-modal encoder, we employ the graph-aware self-attention (GASA) mechanism mentioned in [5] and [15] for the self-attention sublayer, which further takes into account the topology of the graph to compute attention as follows:

$$GASA(X) = S\left(\frac{XW_q\left(XW_k\right)^T}{\sqrt{d}} + EW_e\right)XW_v \tag{8}$$

where S denotes the softmax function, X represents node representations in the map, E is the distance matrix constructed by all shortest path pairs derived from the edges of the topological map, and W_q, W_k, W_e, W_v are learnable matrices. The final result is a graph cross-modal attention encoding that captures the correspondence between the visual information associated with each node and the corresponding text.

Planning. The cross-modal attention encoding is passed through a feed-forward network to predict stop score for each node in the map. In order to avoid visiting a node repeatedly, we mask the stop scores of visited nodes and the current node. Based on the stop scores, the agent selects an unvisited navigable node $\tilde{n}_t$ in the global topological map as a long-term goal, which is a previously observed but unvisited node. To plan a path towards this long-term goal, the agent executes Dikjstra's algorithm according to the distance information recorded in the topology map to compute the shortest path between the current node and $\tilde{n}_t$. Finally, the agent acquires a path plan from the current node to $\tilde{n}_t$ expressed as $P_t = \{p_n\}_{n=1}^{N}$, where p_n and N separately represent the position and number of the route nodes in the path. If $\tilde{n}_t$ is the 'stop' node, the agent stops at the current location.

D. Control

The control module is responsible for translating the long-term goal path plan into a sequence of low-level actions for the agent: `FORWARD` (0.25m), `TURN LEFT/RIGHT` (15°), and `STOP`, which control the agent to move to the goal.

To reach the subgoal node p_n, the agent first acquires its current state, including information such as position and orientation. In the second place, the agent calculates the angle and distance between the current node and p_n. Next, the agent rotates the direction and moves forward in this direction. Since the agent can only perform low-level actions, both rotation and forward movement are achieved by executing multiple low-level actions to reach the correct direction and distance.

For a sequence of nodes in the path from the current node to the long-term goal $\tilde{n}_t$, a similar control process is iterated. Each time a node is reached, it is removed from the path P_t. The next node in the path becomes the new subgoal, and the control steps are repeated until there are no more nodes in the path. At this time, the long-term goal node $\tilde{n}_t$ is reached, all operations of the current time step are completed, and the agent obtains new observations in its new location.

IV. Experiments

A. Experimental Setting

Datasets. The experiments are conducted on the VLN-CE [6] dataset. VLN-CE dataset is converted by Habitat Simulator [18] from the R2R dataset [1] in discrete VLN, which is the most widely used dataset for continuous VLN. It comprises 16,844 path-instruction pairs over 90 real-world environments from the Matterport3D [46] dataset. VLN-CE dataset is divided into train, seen validation (*val_seen*), unseen validation (*val_unseen*) and test splits. We adhere to the typical evaluation scenarios and report results on two validation sets. Both *val_seen* and *val_unseen* include novel paths and instructions, scenes in *val_seen* were observed during training but scenes in *val_unseen* were not.

Evaluation metrics. An episode is considered successful if the agent calls `STOP` action within 3m of the goal. We evaluate the navigation performance using five metrics. TL (Trajectory length) measures the average length of the predicted navigation trajectories. NE (navigation error) measures the average distance between the final position and the target, SR (success rate) is the ratio of the agent successfully reaches the target, OS (oracle success rate) is the proportion of the closest point in the predicted trajectory to the target within 3m. SPL (success weighted by path length) integrates SR and TL, which measures both the accuracy and the efficiency of navigation. More details of these evaluation metrics can be found in [1], [47].

Implementation details. The proposed method is implemented on Pytorch [48] framework and Habitat Simulator [18]. Four NVIDIA RTX 3090 GPUs were utilized for 20,000 iterations (4 days on average), with each GPU concurrently running 8 environments. The AdamW optimizer with a learning rate of 1e-5 is employed. We adopt the pre-trained model from [15] and apply scheduled sampling to continue training the model, ultimately selecting the checkpoint with the highest SPL on *val_unseen* set. For visual encoding, several pre-trained models are employed. The ViT-B/32 model, pre-trained in CLIP [38], is used to extract overall RGB features. Additionally, the ResNet-50 pre-trained in point-goal navigation [49] is used to extract depth features. Furthermore, a pre-trained semantic segmentation model [13] based on ResNet-18 was employed to obtain semantic features for the fusion of low-level and high-level features. The parameters of these models are fixed.

TABLE I

COMPARISON BETWEEN OUR MV-TOPO AND STATE-OF-THE-ART METHODS ON VLN-CE DATASET.

Methods	Val-Seen					Val-Unseen					Test				
	TL	NE↓	OS↑	SR↑	SPL↑	TL	NE↓	OS↑	SR↑	SPL↑	TL	NE↓	OS↑	SR↑	SPL↑
Seq2Seq [6]	9.37	7.02	46.0	33.0	31.0	9.32	7.77	37.0	25.0	22.0	8.85	7.91	36	28	25
LAW [50]	9.34	6.35	49.0	40.0	37.0	8.89	6.83	44.0	35.0	31.0	-	-	-	-	-
CMTP [14]	-	7.10	45.4	36.1	31.2	-	7.9	38.0	26.4	22.7	-	-	-	-	-
HPN [7]	8.54	5.48	53	46.0	43.0	7.62	6.31	40.0	36.0	34.0	8.02	6.65	37	32	30
CM2 [12]	12.05	6.10	50.7	42.9	34.8	11.54	7.02	41.5	34.3	27.6	13.90	7.70	39	31	24
CWP-CMA [10]	11.47	5.20	61.0	51.0	45.0	10.90	6.20	52.0	41.0	36.0	11.85	6.30	49	38	33
CWP-RecBERT [10]	12.50	5.02	59.0	50.0	44.0	12.23	5.74	53.0	44.0	39.0	13.31	5.89	51	42	36
Sim2Sim[9]	11.18	4.67	61.0	52.0	44.0	10.69	6.07	52.0	43.0	36.0	11.43	6.17	52	44	37
WS-MGMAP [13]	10.12	5.65	51.7	46.9	43.4	10.00	6.28	47.6	38.9	34.3	12.30	7.11	45	35	28
Ego2-Map [44]	-	-	-	-	-	-	4.94	-	51.8	46.1	13.05	5.54	56	47	41
ETP [15]	11.78	3.95	**72.0**	**66.0**	59.0	11.99	**4.71**	**65.0**	57.0	49.0	12.87	5.12	**63**	**55**	**48**
MV-Topo (ours)	11.25	**3.69**	71.3	65.6	**59.7**	11.85	4.72	63.0	**57.5**	**49.1**	12.71	**4.94**	60	54	**48**

Fig. 5. Navigation example using our method MV-Topo.

B. Comparison with the State of the Art

We compare our MV-Topo with state-of-the-art methods on two validation sets of the VLN-CE dataset in Table I, a navigation example is shown in Fig. 5. The experimental results demonstrate that our model outperforms existing models on *val_seen* in terms of NE and SPL, while obtaining comparable results with the powerful model ETP [15] in terms of OS and SR. Regarding performance on *val_unseen*, MV-Topo yields the best results on SR and SPL, and is nearly on par with [15] on NE. These results indicate the excellent performance of our model and prove the effectiveness of our proposed multiple visual features approach. In particular, compared with CMTP [14] which also employs topology, MV-Topo achieves superior performance on all metrics. CMTP requires a prior scene exploration phase to gain knowledge of the scene topology, whereas our approach constructs the topology map online during navigation without any prior exploration process. WS-MGMAP [13] applies a semantic segmentation model to construct a fine-grained graph, after which it utilizes the fine-grained graph to generate a semantic graph, and the two graphs are fused to obtain a multi-granularity map. In contrast, MV-Topo employs the topological map as the primary body for carrying visual information. This structure strengthens the correlation between views and eliminates the step of fusing two separate maps, improving the efficiency as well as the model performance. Additionally, for Ego^2-Map [44] which focuses on visual feature representation, it constructs semantic maps and uses two learnable visual encoders to enrich egocentric representations. Instead of building a semantic map, MV-Topo utilizes a semantic segmentation model to extract fine-grained semantic information from RGB images, which are later fused with RGB features and depth features. This approach of integrating multiple visual features reduces computational costs and enhances performance.

VLN-CE Leaderboard. We also submitted MV-Topo on the held-out test-unseen set used for VLN-CE leaderboard[1]. Table I shows that MV-Topo drops NE by 3.5% while main-

[1]VLN-CE Leaderboard: `https://eval.ai/web/challenges/challenge-page/719/leaderboard/1966`

TABLE II

ABLATION OF VISUAL FEATURES IN TRAINING ON VLN-CE VALIDATION SPLITS.

rgb+depth	rgb+depth+semantic		Val-Seen					Val-Unseen				
	early fuse	late fuse	TL	NE↓	OS↑	SR↑	SPL↑	TL	NE↓	OS↑	SR↑	SPL↑
✓			11.23	3.94	70.4	64.5	58.3	11.89	4.81	**63.5**	56.2	47.9
	✓		12.13	3.75	**71.9**	64.8	58.5	12.49	4.92	60.5	53.6	45.8
		✓	11.25	**3.69**	71.3	**65.6**	**59.7**	11.85	**4.72**	63.0	**57.5**	**49.1**

TABLE III

ABLATION OF POOLING MANNER IN TRAINING ON VLN-CE VALIDATION SPLITS.

Pooling	Val-Seen			Val-Unseen		
	NE↓	SR↑	SPL↑	NE↓	SR↑	SPL↑
avg	3.69	65.6	59.7	4.72	57.5	49.1
max	4.40	62.1	54.3	4.96	56.2	47.8

taining similar SPL and SR, demonstrating that our model is competitive and multiple visual features are beneficial to VLN-CE.

C. Ablation Studies

Multiple visual features vs. normal visual features. To explore the effectiveness of multiple visual features on navigation performance, we conduct ablation experiments. We compare the impacts of our proposed multiple visual features with normal visual features on two validation splits of the VLN-CE dataset. The results of these ablation experiments are presented in Table II. The agent utilizing late fused multiple visual features outperforms the agent using normal visual features on several metrics, improving SR by 1.7% and 2.3%, improving SPL by 2.4% and 2.5%, and dropping NE by 6.3% and 1.9% on *val_seen* and *val_unseen*, respectively. The results indicate that incorporating multiple visual features greatly improves the agent's ability to utilize visual information, which is beneficial to navigation. Especially in seen environments, NE achieved a significant decrease, which suggests that the agent has a stronger ability to perceive seen objects.

Early Fusion vs. Late Fusion. We further investigate the effect of early fusion and late fusion of multiple features on the performance of the agent. Considering that both semantic and RGB features are extracted from RGB images, we construct an early fusion variant. This variant fuses RGB features and semantic features in advance. Specifically, the two features are connected and then fed to a convolution layer to obtain a new feature representing the RGB image, which is later fused with the depth feature. In comparison, late fusion processes RGB, semantic, and depth features in parallel and fuses them at the end. Table II shows that late fusion is more beneficial and outperforms early fusion in all metrics for both validation sets, except for OS in *val_seen*. We suspect that the performance difference between early fusion and late fusion could be attributed to the additional convolutional operation performed in early fusion, which condenses both the RGB features and semantic features, potentially diminishing their individual contributions in guiding navigation.

Average Pooling vs. Max Pooling. When introducing semantic features, we applied a max pooling layer to process these features, while previous approaches [4], [15] employed average pooling for visual features. Therefore, we conduct a comparative experiment to assess the influence of these two pooling methods for semantic features on the navigation performance. The experimental results, presented in Table III, demonstrate that max pooling surpasses average pooling in terms of NE, SR and SPL on both validation sets. The result suggests that for the newly integrated semantic information, max pooling is the more suitable and effective choice.

V. CONCLUSION

In this paper, we propose a new visual feature representation method, MV-Topo, for vision-and-language navigation in continuous environments. Our approach incorporates fine-grained semantic features containing rich information into the visual representation of nodes within the topological map. By doing so, the agent improves its visual perception and comprehension of the environment, which benefits the subsequent planning and control. Through a series of experiments, we demonstrate the effectiveness of our proposed method, and our results are competitive with existing methods.

ACKNOWLEDGMENT

This work was partly supported by the National Natural Science Foundation of China under Grants No. 62206199 and U2141234, Tianjin Applied Basic Research Project under Grant No. 22JCQNJC00410, Young Elite Scientist Sponsorship Program under Grant No. YESS20220409, Alexander von Humboldt Foundation under Grant No. 1226831, State Key Laboratory of Reliability and Intelligence of Electrical Equipment under Grant No. EERI-KF2022001, Shanghai Science and Technology program under Grant No. 22015810300 and Hainan Province Science and Technology Special Fund under Grant No.ZDYF2024GXJS003.

REFERENCES

[1] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. Reid, S. Gould, and A. Van Den Hengel, "Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments," in *CVPR*, 2018, pp. 3674–3683.

[2] X. Wang, Q. Huang, A. Celikyilmaz, J. Gao, D. Shen, Y.-F. Wang, W. Y. Wang, and L. Zhang, "Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation," in *CVPR*, 2019, pp. 6629–6638.

[3] H. Tan, L. Yu, and M. Bansal, "Learning to navigate unseen environments: Back translation with environmental dropout," in *NAACL-HLT*, J. Burstein, C. Doran, and T. Solorio, Eds., 2019, pp. 2610–2621.

[4] Y. Hong, Q. Wu, Y. Qi, C. Rodriguez-Opazo, and S. Gould, "Vln bert: A recurrent vision-and-language bert for navigation," in *CVPR*, 2021, pp. 1643–1653.

[5] S. Chen, P.-L. Guhur, M. Tapaswi, C. Schmid, and I. Laptev, "Think global, act local: Dual-scale graph transformer for vision-and-language navigation," in *CVPR*, 2022, pp. 16 537–16 547.

[6] J. Krantz, E. Wijmans, A. Majumdar, D. Batra, and S. Lee, "Beyond the nav-graph: Vision-and-language navigation in continuous environments," in *ECCV*. Springer, 2020, pp. 104–120.

[7] J. Krantz, A. Gokaslan, D. Batra, S. Lee, and O. Maksymets, "Waypoint models for instruction-guided navigation in continuous environments," in *ICCV*, 2021, pp. 15 162–15 171.

[8] P. Anderson, A. Shrivastava, J. Truong, A. Majumdar, D. Parikh, D. Batra, and S. Lee, "Sim-to-real transfer for vision-and-language navigation," in *Conference on Robot Learning*. PMLR, 2021, pp. 671–681.

[9] J. Krantz and S. Lee, "Sim-2-sim transfer for vision-and-language navigation in continuous environments," in *ECCV*. Springer, 2022, pp. 588–603.

[10] Y. Hong, Z. Wang, Q. Wu, and S. Gould, "Bridging the gap between learning in discrete and continuous environments for vision-and-language navigation," in *CVPR*, 2022, pp. 15 439–15 449.

[11] M. Z. Irshad, N. C. Mithun, Z. Seymour, H.-P. Chiu, S. Samarasekera, and R. Kumar, "Semantically-aware spatio-temporal reasoning agent for vision-and-language navigation in continuous environments," in *ICPR*. IEEE, 2022, pp. 4065–4071.

[12] G. Georgakis, K. Schmeckpeper, K. Wanchoo, S. Dan, E. Miltsakaki, D. Roth, and K. Daniilidis, "Cross-modal map learning for vision and language navigation," in *CVPR*, 2022, pp. 15 460–15 470.

[13] P. Chen, D. Ji, K. Lin, R. Zeng, T. Li, M. Tan, and C. Gan, "Weakly-supervised multi-granularity map learning for vision-and-language navigation," in *NeurIPS*, vol. 35, 2022, pp. 38 149–38 161.

[14] K. Chen, J. K. Chen, J. Chuang, M. Vázquez, and S. Savarese, "Topological planning with transformers for vision-and-language navigation," in *CVPR*, 2021, pp. 11 276–11 286.

[15] D. An, H. Wang, W. Wang, Z. Wang, Y. Huang, K. He, and L. Wang, "Etpnav: Evolving topological planning for vision-language navigation in continuous environments," *arXiv preprint arXiv:2304.03047*, 2023.

[16] A. Ku, P. Anderson, R. Patel, E. Ie, and J. Baldridge, "Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding," in *EMNLP*, B. Webber, T. Cohn, Y. He, and Y. Liu, Eds., 2020, pp. 4392–4412.

[17] V. Jain, G. Magalhães, A. Ku, A. Vaswani, E. Ie, and J. Baldridge, "Stay on the path: Instruction fidelity in vision-and-language navigation," in *ACL*, A. Korhonen, D. R. Traum, and L. Màrquez, Eds., 2019, pp. 1862–1872.

[18] 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 *ICCV*, 2019, pp. 9339–9347.

[19] S. Liu, H. Zhang, Y. Qi, P. Wang, Y. Zhang, and Q. Wu, "Aerialvln: Vision-and-language navigation for uavs," in *ICCV*, 2023, pp. 15 384–15 394.

[20] D. Fried, R. Hu, V. Cirik, A. Rohrbach, J. Andreas, L.-P. Morency, T. Berg-Kirkpatrick, K. Saenko, D. Klein, and T. Darrell, "Speaker-follower models for vision-and-language navigation," in *NeurIPS*, vol. 31, 2018.

[21] H. Huang, V. Jain, H. Mehta, J. Baldridge, and E. Ie, "Multi-modal discriminative model for vision-and-language navigation," *arXiv preprint arXiv:1905.13358*, 2019.

[22] C. Ma, J. Lu, Z. Wu, G. AlRegib, Z. Kira, R. Socher, and C. Xiong, "Self-monitoring navigation agent via auxiliary progress estimation," in *ICLR*, 2019.

[23] D. An, Y. Qi, Y. Huang, Q. Wu, L. Wang, and T. Tan, "Neighbor-view enhanced model for vision and language navigation," in *ACM MM*, 2021, pp. 5101–5109.

[24] P.-L. Guhur, M. Tapaswi, S. Chen, I. Laptev, and C. Schmid, "Airbert: In-domain pretraining for vision-and-language navigation," in *ICCV*, 2021, pp. 1634–1643.

[25] Y. Qiao, Y. Qi, Y. Hong, Z. Yu, P. Wang, and Q. Wu, "Hop+: History-enhanced and order-aware pre-training for vision-and-language navigation," *IEEE TPAMI*, 2023.

[26] M. Hwang, J. Jeong, M. Kim, Y. Oh, and S. Oh, "Meta-explore: Exploratory hierarchical vision-and-language navigation using scene object spectrum grounding," in *CVPR*, 2023, pp. 6683–6693.

[27] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *NAACL*, J. Burstein, C. Doran, and T. Solorio, Eds., 2019, pp. 4171–4186.

[28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, vol. 30, 2017.

[29] A. Majumdar, A. Shrivastava, S. Lee, P. Anderson, D. Parikh, and D. Batra, "Improving vision-and-language navigation with image-text pairs from the web," in *ECCV*. Springer, 2020, pp. 259–274.

[30] Z. Wang, J. Li, Y. Hong, Y. Wang, Q. Wu, M. Bansal, S. Gould, H. Tan, and Y. Qiao, "Scaling data generation in vision-and-language navigation," in *ICCV*, 2023, pp. 12 009–12 020.

[31] C. Liu, F. Zhu, X. Chang, X. Liang, Z. Ge, and Y.-D. Shen, "Vision-language navigation with random environmental mixup," in *ICCV*, 2021, pp. 1644–1654.

[32] J. Li, H. Tan, and M. Bansal, "Envedit: Environment editing for vision-and-language navigation," in *CVPR*, 2022, pp. 15 407–15 417.

[33] S. Wang, C. Montgomery, J. Orbay, V. Birodkar, A. Faust, I. Gur, N. Jaques, A. Waters, J. Baldridge, and P. Anderson, "Less is more: Generating grounded navigation instructions from landmarks," in *CVPR*, 2022, pp. 15 428–15 438.

[34] D. S. Chaplot, D. P. Gandhi, A. Gupta, and R. R. Salakhutdinov, "Object goal navigation using goal-oriented semantic exploration," in *NeurIPS*, vol. 33, 2020, pp. 4247–4258.

[35] G. Georgakis, B. Bucher, K. Schmeckpeper, S. Singh, and K. Daniilidis, "Learning to map for active semantic goal navigation," in *ICLR*, 2022.

[36] D. S. Chaplot, R. Salakhutdinov, A. Gupta, and S. Gupta, "Neural topological slam for visual navigation," in *CVPR*, 2020, pp. 12 875–12 884.

[37] Y. Zhao, J. Chen, C. Gao, W. Wang, L. Yang, H. Ren, H. Xia, and S. Liu, "Target-driven structured transformer planner for vision-language navigation," in *ACM MM*, 2022, pp. 4194–4203.

[38] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *ICML*, 2021, pp. 8748–8763.

[39] S. Y. Gadre, M. Wortsman, G. Ilharco, L. Schmidt, and S. Song, "Clip on wheels: Zero-shot object navigation as object localization and exploration," *arXiv preprint arXiv:2203.10421*, vol. 3, no. 4, p. 7, 2022.

[40] S. Shen, L. H. Li, H. Tan, M. Bansal, A. Rohrbach, K. Chang, Z. Yao, and K. Keutzer, "How much can CLIP benefit vision-and-language tasks?" in *ICLR*, 2022.

[41] K. Yadav, R. Ramrakhya, A. Majumdar, V.-P. Berges, S. Kuhar, D. Batra, A. Baevski, and O. Maksymets, "Offline visual representation learning for embodied navigation," in *ICLR Workshop*, 2023.

[42] N. Savinov, A. Dosovitskiy, and V. Koltun, "Semi-parametric topological memory for navigation," in *ICLR*, 2018.

[43] D. Gordon, A. Kadian, D. Parikh, J. Hoffman, and D. Batra, "Splitnet: Sim2sim and task2task transfer for embodied visual navigation," in *ICCV*, 2019, pp. 1022–1031.

[44] Y. Hong, Y. Zhou, R. Zhang, F. Dernoncourt, T. Bui, S. Gould, and H. Tan, "Learning navigational visual representations with semantic map supervision," in *ICCV*, 2023, pp. 3055–3067.

[45] H. Tan and M. Bansal, "LXMERT: learning cross-modality encoder representations from transformers," in *EMNLP-IJCNLP*, K. Inui, J. Jiang, V. Ng, and X. Wan, Eds., 2019, pp. 5099–5110.

[46] A. X. Chang, A. Dai, T. A. Funkhouser, M. Halber, M. Nießner, M. Savva, S. Song, A. Zeng, and Y. Zhang, "Matterport3d: Learning from RGB-D data in indoor environments," in *3DV*, 2017, pp. 667–676.

[47] P. Anderson, A. Chang, D. S. Chaplot, A. Dosovitskiy, S. Gupta, V. Koltun, J. Kosecka, J. Malik, R. Mottaghi, M. Savva, *et al.*, "On evaluation of embodied navigation agents," *arXiv preprint arXiv:1807.06757*, 2018.

[48] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *NeurIPS*, vol. 32, 2019.

[49] E. Wijmans, A. Kadian, A. Morcos, S. Lee, I. Essa, D. Parikh, M. Savva, and D. Batra, "DD-PPO: learning near-perfect pointgoal navigators from 2.5 billion frames," in *ICLR*, 2020.

[50] S. Raychaudhuri, S. Wani, S. Patel, U. Jain, and A. X. Chang, "Language-aligned waypoint (LAW) supervision for vision-and-language navigation in continuous environments," in *EMNLP*, M. Moens, X. Huang, L. Specia, and S. W. Yih, Eds., 2021, pp. 4018–4028.