Temporal-Spatial Object Relations Modeling for Vision-and-Language Navigation

Bowen Huang, Yanwei Zheng, *Member, IEEE*, Dongchen Sui, Chuanlin Lan, Xinpeng Zhao, Xiao Zhang, Jingke Meng, Mengbai Xiao, Yifei Zou, *Member, IEEE*, and Dongxiao Yu, *Senior Member, IEEE*

***Abstract*—Vision-and-Language Navigation (VLN) is a challenging task where an agent is required to navigate to a natural language described location via vision observations. The navigation abilities of the agent can be enhanced by the relations between objects, which are usually learned using internal objects or external datasets. The relationships between internal objects are modeled employing graph convolutional network (GCN) in traditional studies. However, GCN tends to be shallow, limiting its modeling ability. To address this issue, we utilize a cross attention mechanism to learn the connections between objects over a trajectory, which takes temporal continuity into account, termed as Temporal Object Relations (TOR). The external datasets have a gap with the navigation environment, leading to inaccurate modeling of relations. To avoid this problem, we construct object connections based on observations from all viewpoints in the navigational environment, which ensures complete spatial coverage and eliminates the gap, called Spatial Object Relations (SOR). Additionally, we observe that agents may repeatedly visit the same location during navigation, significantly hindering their performance. For resolving this matter, we introduce the Turning Back Penalty (TBP) loss function, which penalizes the agent's repetitive visiting behavior, substantially reducing the navigational distance. Experimental results on the REVERIE, SOON, Touchdown and R2R datasets demonstrate the effectiveness of the proposed method.**

***Index Terms*—Vision-and-language navigation, temporal object relations, spatial object relations, turning back penalty.**

I. Introduction

In recent years, vision-and-language navigation (VLN) has shown great promise in intelligent transportation systems, offering more intuitive and effective ways for autonomous vehicles to interpret instructions and navigate complex urban environments [1], [2]. The goal of VLN [3], [4], [5], [6] is to guide an agent to a target location based on a natural language instruction. While many vision-and-language problems have been extensively explored [7], [8], [9], [10], VLN remains highly challenging. This is due to the dynamic nature of real-world environments and the complexity of the language instructions.

Significant progress has been made in the field of VLN [4], [11]. Most existing methods [12], [13], [14], [15] use RNNs (e.g., GRUs or LSTMs) or transformer-based models to process visual inputs and align them with the instruction for action prediction. Recently, several studies [16], [17], [18] have introduced topological maps and semantic graphs to store historical information, which leads to improved performance. Meanwhile, another line of research [19], [20], [21] highlights the importance of modeling object relationships in navigation environments.

A method for learning the relations between objects is constructing graph-structured feature representations [19], [22]. As depicted in Fig. 1a, a graph-based navigation state is maintained utilizing GCN at each location during the agent's navigation process (such as moving from position 1 to position 3). However, due to the issue of over-smoothing, GCN networks are typically kept shallow, which can impede their ability to accurately learn relationships. Another approach of modeling the object relations is introducing external knowledge. The external knowledge mainly comes from two sources. First, it can be obtained from publicly available image-text datasets [17], [23], [24]. Second, some methods use the pretrained ConceptNet system [25], [26] or large language models [27], [28], [29] to acquire the knowledge. However, as shown in Fig. 1b, this knowledge is not directly collected from within the navigation environment. As a result, there exists a significant gap between the learned object relations and those in the real environment.

To address the above problems, we propose two modules: the Temporal Object Relations (TOR) module and the Spatial Object Relations (SOR) module. As illustrated in Fig. 1c, the TOR module models object relations along the agent's trajectory. At each position of the trajectory, it uses a cross attention mechanism to compute a relation matrix between observed objects and instruction nouns. This matrix is updated as the agent moves. In this way, TOR captures how object relations change over time during navigation. On the other hand, the SOR module models spatial relations across the whole environment. It collects object co-occurrence information from all viewpoints. Each viewpoint is treated equally, regardless of

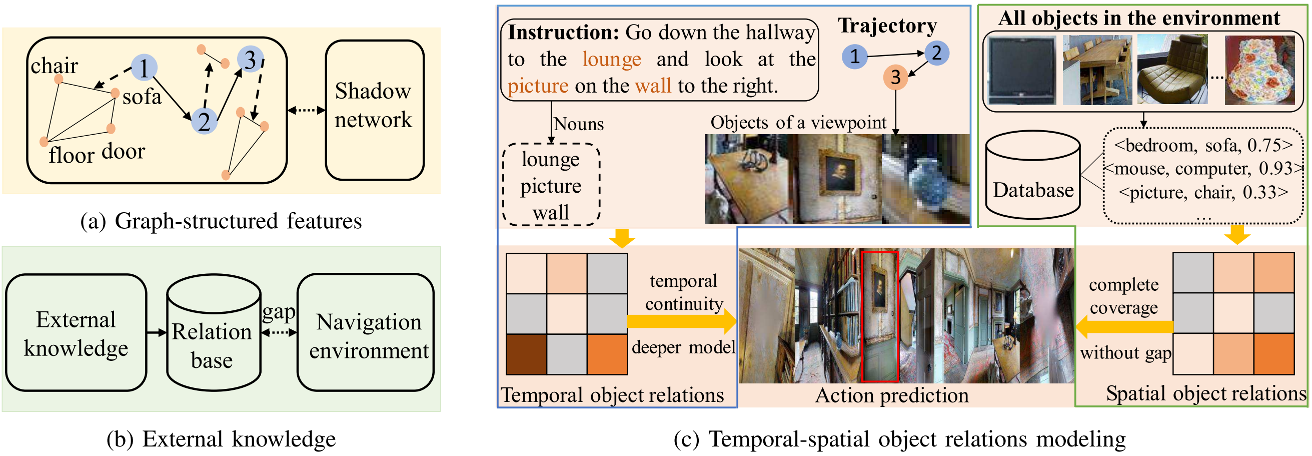
Received 17 May 2024; revised 12 June 2025 and 24 July 2025; accepted 12 August 2025. This work was supported by the Natural Science Foundation of Shandong Province, China, under Grant ZR2022ZD02. The Associate Editor for this article was B. De Schutter. *(Corresponding author: Yanwei Zheng.)*

Bowen Huang, Yanwei Zheng, Dongchen Sui, Chuanlin Lan, Xinpeng Zhao, Xiao Zhang, Mengbai Xiao, Yifei Zou, and Dongxiao Yu are with the School of Computer Science and Technology, Shandong University, Qingdao 266237, China (e-mail: huangbw@mail.sdu.edu.cn; zhengyw@sdu.edu.cn; suidongchen@mail.sdu.edu.cn; lancl@sdu.edu.cn; zhaoxp1001@gmail.com; xiaozhang@sdu.edu.cn; xiaomb@sdu.edu.cn; yfzou@sdu.edu.cn; dxyu@sdu.edu.cn).

Jingke Meng is with the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou 510006, China (e-mail: mengjke@gmail.com).

Code is available on-line at https://github.com/visee-sdu/TSOR

Digital Object Identifier 10.1109/TITS.2025.3599649



(a) Graph-structured features

(b) External knowledge

(c) Temporal-spatial object relations modeling

Fig. 1. Three methods of learning the connections between objects.

the current trajectory. The result is a global relation graph that covers the entire environment.

While these modules enhance the agent's ability to understand and navigate complex environments, they also introduce a new challenge. The detailed tracking and continuous updating of object relations can lead the agent to explore new locations that are not part of the correct path or to conduct multiple explorations at the same location. Exploration across new locations is helpful, granting agents critical environmental insights and informing their navigational decisions [16], [30]. However, repetitive revisits to the same viewpoint do not enhance navigational success but rather impair efficiency. To counteract this, we introduce the Turning Back Penalty (TBP) loss function. Specifically, during the training process of the agent, it penalizes the agent each time it passes a previously visited location. This effectively mitigates the issue of revisits, thereby improving navigation efficiency. Our primary contributions can be summarized as follows:

- We propose the TOR and SOR modules, which learn the interdependent relations among different objects from the dimensions of time and space, respectively.
- We introduce the TBP loss function, which effectively alleviates the problem of excessive path length caused by repeated visits to the same location by the agent.
- Extensive experiments have been conducted on the REVERIE [6], SOON [31], Touchdown [32] and R2R [4] datasets to demonstrate the superiority of our method over existing approaches in visual-and-language navigation.

The rest of the paper is organized as follows. Section II reviews relevant research about VLN. Section III introduces the details of our method. In Section IV, we present the training methodology and parameter settings of our model, and evaluate it on four datasets. We conclude the paper in Section V.

II. Related Work

A. Vision-and-Language Navigation

VLN [33], [34], [35], [36], [37] has received significant research interests in recent years with the continual improvement. Early methods [12], [33], [38], [39] usually utilize recurrent neural networks (RNNs) to encode historical observations and actions, which are represented as a state vector. In order to capture environment layouts, Wang et al. [35] employ a structured scene memory to accurately memorize the percepts during navigation. Tan et al. [14] propose a two-stage training approach to enhance the generalization ability of the agent. Ma et al. [40] use a progress monitor as a learnable heuristic for search. RPA [41] integrates model-free and model-based reinforcement learning through environment modeling and look-ahead planning, achieving strong generalization on the R2R task.

More recently, transformer-based architectures have been shown successful in VLN tasks [42], notably by leveraging pre-trained architectures. PRESS [43] proposes a stochastic sampling scheme to reduce the considerable gap between the expert actions in training and sampled actions in test. VLN-BERT [11] employs recurrent units in transformer architecture to predict actions. Loc4Plan [44] introduces spatial localization before action planning, leading to improved alignment between instructions and the environment. LOViS [45] separately models orientation and visual signals using a modular design and task-specific pretraining to improve spatial and visual grounding. To learn general navigation oriented textual representations, both AirBERT [3] and HM3D-AutoVLN [37] introduce expansive VLN dataset to enhance the interaction between various modalities. DUET [16] adeptly merges local observations with the overarching topological map through the use of graph transformers. This streamlines action planning and bolsters cross-modal comprehension. GridMM [18] builds a top-down egocentric and dynamically growing grid memory map to structure the visited environment. In contrast, our work proposes an object-relations model designed to enhance the agent's understanding of the environment.

B. Object Relations Modeling

Recently, some studies have begun to focus on utilizing the relationships between objects to guide agent navigation [19], [46], [47]. ORG [19] improves visual representation learning by integrating object relationships, including category closeness and spatial correlations. SEvol [22] proposes a novel structured state-evolution model to learn the object-level relationship. CKR [26] proposes a knowledge-enabled entity relationship reasoning module to learn the internal-external correlations among room- and object-entities. EXOR [48] aligns spatial relations between landmarks in instructions and visual objects in the environment to enhance spatial reasoning and interpretability. KERM [17] constructs an external knowledge base to assist in establishing relationships between the various entities described in the instructions. OAAM [49] utilizes two learnable attention modules to highlight language relating to objects and actions within a given instruction. VLMaps [50] translates natural language commands into a sequence of open-vocabulary navigation goals using large language models, resulting in objects that are spatially defined. While these methods demonstrate the benefits of incorporating object relationships, they often focus on static or local associations, lack temporal continuity, or are limited to predefined object scopes. In contrast, our method explicitly models object relations across both spatial and temporal dimensions, enabling the agent to learn richer and more dynamic object-level representations during navigation.

C. Training Regimes

Previous studies [16] mostly train the agent with the supervision from a pseudo interactive demonstrator similar to the DAgger algorithm [51]. Anderson et al. [4] introduce two distinct training regimes, teacher-forcing and student-forcing, and utilizes cross entropy loss at each step to maximize the likelihood of the ground-truth target action. Ma et al. [52] introduce a self-monitoring loss function that enhances the agent's performance by estimating its navigation progress. Tan et al. [14] introduce an environment dropout method and enable the agent to navigate in environments with incomplete information and improving its generalization. Wang et al. [33] introduce a loss function that integrates reinforcement learning and self-supervised learning to optimize the agent's matching capability across different modalities. These methods accumulate penalties for the agent as the number of exploratory steps increases. However, due to the poor balancing of penalty intensity, this may lead to either excessive revisitation of the same location by the agent or insufficient exploration of the environment. In contrast, our TBP loss function avoids such pitfalls by preventing the agent from retracing its steps while ensuring ample exploration of the environment.

III. Method

In the VLN task, the agent is initially located at a starting node in a previously unseen environment. The environment is represented by a weighted undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{A}\}$, where $\mathcal{V}$ denotes navigable nodes and $\mathcal{A}$ denotes edges. The agent needs to explore this environment to reach the target location, guided by a natural language instruction. The instruction embedding consisting of L words is $\mathcal{W} = \{\boldsymbol{w}_i\}_{i=1}^{L}$. At each time step t, the agent observes a panoramic view of the current node. The panorama is divided into n different perspective images $\mathcal{R}_t = \{\boldsymbol{r}_i\}_{i=1}^{n}$, where $\boldsymbol{r}_i$ denotes the image feature of the i-th perspective and the direction encoding of that perspective. In addition, m object features $\mathcal{O}_t = \{\boldsymbol{o}_i\}_{i=1}^{m}$ are extracted from the panorama. This is done using annotated object bounding boxes or automatic object detectors [53], enhancing the agent's fine-grained visual perception.

A. Overview of Our Approach

As shown in Fig. 2(a), we adopt the architecture of DUET [16] as the baseline, It consists of three inputs: panoramic visual features of the current location, a topological map, and an instruction. At time step t, the topological map is represented as $\mathcal{G}_t = \{\mathcal{V}_t, \mathcal{A}_t\}$. Here, $\mathcal{G}_t$ is a subset of the overall map $\mathcal{G}$ and encapsulates the state of the environment after t steps of navigation. $\mathcal{V}_t$ contains three kinds of nodes: visited nodes (circular nodes), the current node (pentagram nodes), and navigable nodes (triangular nodes).

Our method, as illustrated in Fig. 2(b), employs two modules to calculate temporal object features $\mathcal{M}_t$ and spatial object features $\mathcal{N}_t$ at each step t, respectively. These features are then combined with $\mathcal{O}_t$ and $\mathcal{R}_t$, in order to generate the panoramic feature $\mathcal{Q}_t$, which is fed into a dual-scale encoder to predict the agent's action. The panoramic image and object features are independently extracted at each time step without temporal accumulation, and the temporal modeling is solely performed by the TOR module via attention updates. To further enhance the agent's performance, we introduce the TBP loss function. It can help to prevent repeated explorations and reduce the length of the agent's path.

B. Object Relations

In the VLN task, there are connections between the various objects that the agent perceives. In our method, these connections are learned from two dimensions of time and space, significantly enhancing the accuracy of the navigation.

1) Object Nouns Features: To extract object-related noun features from the instruction, we begin by choosing word embeddings describing objects from $\mathcal{W}$. Specifically, our process begins by obtaining labels for all objects from the MatterPort3D simulator [54]. These labels are then compiled into a noun database, denoted as D. For a given natural language instruction $\mathcal{W}$, we iterate through each word. If a word is found within D, it is selected as an object-related token.

Upon acquiring these object-related embeddings, we enhance them with positional embeddings as described in [55]. These positional embeddings correspond to the respective word's location within the sentence. Additionally, a type embedding specific to text, as outlined in [56], is also incorporated. Next, we input all noun tokens into a text encoder that consists of a multi-layer transformer. This process generates contextual noun representations, which we refer to as $\hat{\mathcal{W}} = \{\hat{\boldsymbol{w}}_1, \hat{\boldsymbol{w}}_2, \ldots, \hat{\boldsymbol{w}}_{\hat{L}}\}$. This step ensures that only object-related nouns are considered for contextual modeling.

Our framework can also be extended to support open-vocabulary settings by replacing the fixed-category detector with an open-vocabulary object detector (e.g., GLIP [57] or OWL-ViT [58]). This enables the model to recognize novel objects and align them with instruction nouns beyond the predefined vocabulary.

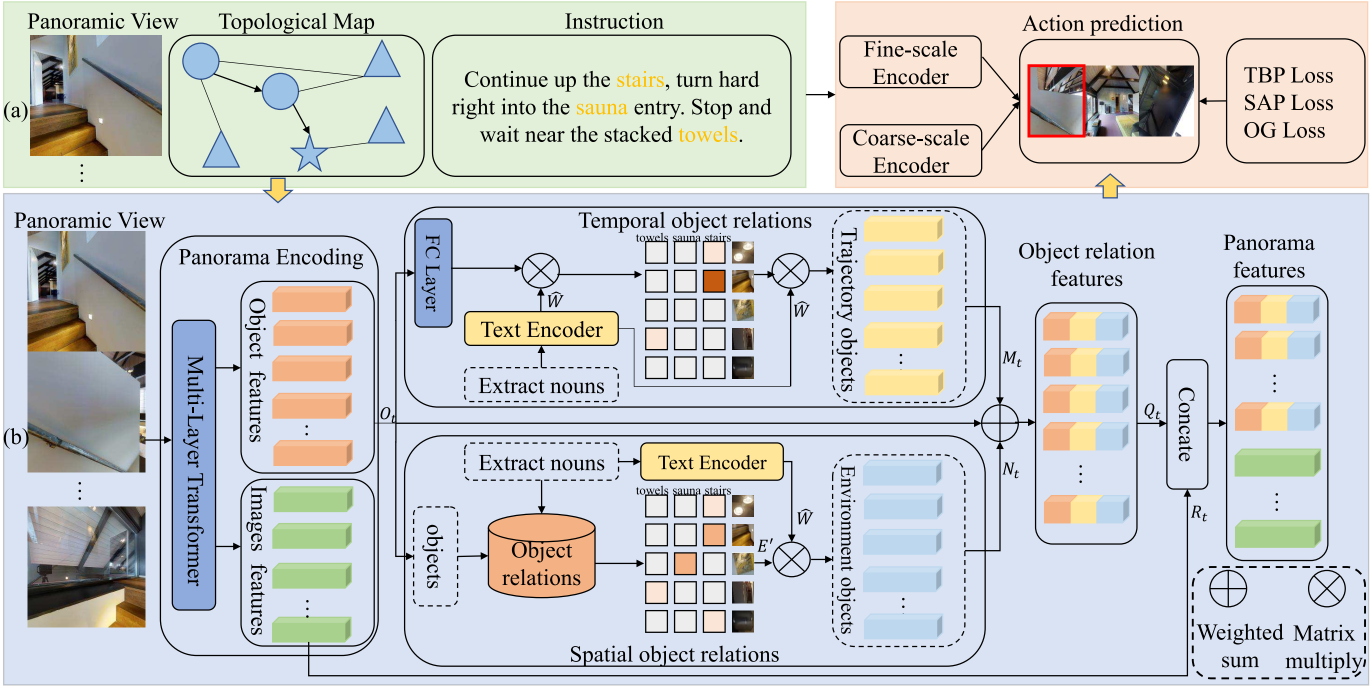
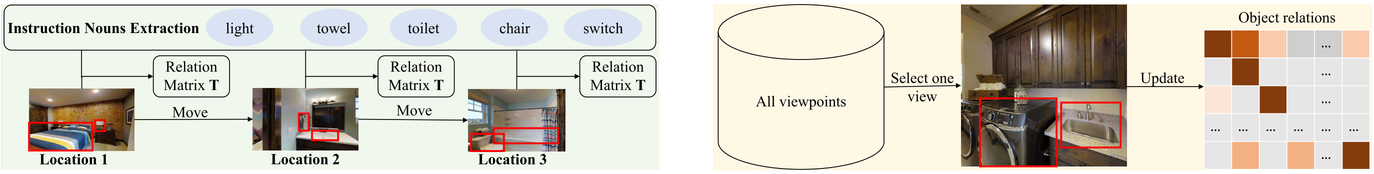


Fig. 2. The overall network architecture. (a) The baseline utilizes a dual-scale encoder to encode local panoramic features, global historical features, and instruction features for action prediction of the agent. (b) At each time step t, our method employ two modules to learn temporal and spatial object relations. Then the object relation features are combined with the image features for action prediction. Finally, we designs a novel TBP loss function to supervise the training of the agent in order to reduce its tendency to backtrack.



(a) At each location, the agent constructs a relation matrix $\mathbf{T}$ between the detected objects and the nouns in the navigation instruction.

(b) The spatial object relationship matrix is constructed by considering the objects visible at each node in the training environment.

Fig. 3. Learning methods for two kinds of relationships.

2) Temporal Object Relations: In the study of object relations, to circumvent the issue of inadequate learning capabilities resulting from the shallow nature of GCN, we have designed a temporal object relations (TOR) module. This module employs a cross attention mechanism to focus on objects observed during navigation and noun embeddings in the instruction. Through this approach, we learn a relationship matrix accurately as the agent progresses along its exploration trajectory. This ensures temporal continuity in the agent's learning process.

Fig. 3a depicts how the agent, when arriving at a new location, establishes connections between the objects it perceives and the relevant nouns from the navigation instruction. Specifically, when the agent reaches a location, it obtains a panoramic view of that position. The agent employs a cross attention mechanism to learn the associations between all objects discovered at this location and the nouns mentioned in the instruction. This process is consistently applied at each location along the agent's navigational path. It sets the stage for the agent to learn and progressively refine the inter-object relationships across temporal dimensions.

In our approach, we treat the object features, denoted as O_t, as the query, and the noun features, $\hat{W}$, as the key. We employ a cross attention mechanism to compute the relationship matrix $\mathbf{T}$, which can be expressed in the following way:

$$\mathbf{T} = \mathrm{FC}\left(\mathcal{O}_t\right)\hat{\mathcal{W}}, \tag{1}$$

where FC is a fully connected layer.

We leverage $\mathbf{T}$ to derive the temporal object features $\mathcal{M}_t$. It is formalized as follow:

$$\mathcal{M}_t = \mathbf{T}\hat{\mathcal{W}}. \tag{2}$$

3) Spatial Object Relations: Due to the gap between external knowledge and the navigation environment, the agent is unable to accurately learn the relationships between objects based on external knowledge. To bridge this gap, we introduce the spatial object relations (SOR) module. This module considers panoramic observations from all locations in the environment, covering full horizontal viewpoints, to ensure complete spatial coverage.

During the establishment of spatial object relations, the agent proceeds to update an relationship matrix $\mathbf{E}$ based on the objects identified at each respective location. At initialization, $\mathbf{E}$ is set as an identity matrix. During updates, we compute spatial correlations only between objects of different categories. Relations among objects of the same category are not updated further. As illustrated in Fig. 3b, for objects x and y observed concurrently at the same location, a shorter distance between them correlates with a stronger association. Consequently, we update the relationship matrix based on the distances between objects as seen by the agent at each location within the environment. The update rule for $\mathbf{E}$ is formalized as follows:

$$\mathbf{E}(x, y) += \frac{k_1}{k_2\|\boldsymbol{v}_x - \boldsymbol{v}_y\|_2 + k_3\|\boldsymbol{d}_x - \boldsymbol{d}_y\|_2}, \tag{3}$$

where $\boldsymbol{v}_x$ and $\boldsymbol{v}_y$ denote the perspectives from the current position in relation to objects x and y, respectively, and $\boldsymbol{d}_x$ and $\boldsymbol{d}_y$ represent the respective depths of objects x and y from the current position. Here, k_1, k_2, and k_3 are predefined constants, and $\|\cdot\|_2$ denotes the L2 norm.

In the course of the agent's training, the pertinent matrix $\mathbf{E}'$ is retrieved from $\mathbf{E}$, guided by the objects detected at the agent's current location and the nouns encapsulated within the instruction. To construct $\mathbf{E}'$, each noun in the instruction is first converted to its corresponding object category based on the predefined noun database D. Assuming the agent currently discovers c_1 objects, and the current instruction contains c_2 nouns, the calculation of $\mathbf{E}'$ is as follows:

$$\mathbf{E}' = \begin{bmatrix} \mathbf{E}_{p(1)q(1)} & \mathbf{E}_{p(1)q(2)} & \mathbf{E}_{p(1)q(3)} & \dots & \mathbf{E}_{p(1)q(c_2)} \\ \mathbf{E}_{p(2)q(1)} & \mathbf{E}_{p(2)q(2)} & \mathbf{E}_{p(2)q(3)} & \dots & \mathbf{E}_{p(2)q(c_2)} \\ \dots & \dots & \dots & \dots & \dots \\ \mathbf{E}_{p(c_1)q(1)} & \mathbf{E}_{p(c_1)q(2)} & \mathbf{E}_{p(c_1)q(3)} & \dots & \mathbf{E}_{p(c_1)q(c_2)} \end{bmatrix}, \tag{4}$$

where $p(i)$ is the index of the i-th object and $q(j)$ is the index of the object category corresponding to the j-th noun. Each entry $\mathbf{E}_{p(i)q(j)}$ represents the spatial relation between the detected object and the object category referred to by the j-th noun. Subsequently, matrix multiplication is employed to derive the environmental object feature $\mathcal{N}_t$, as per the following equation:

$$\mathcal{N}_t = \mathbf{E}'\hat{\mathcal{W}}. \tag{5}$$

Upon obtaining both the temporal object features $\mathcal{M}_t$ and the spatial object features $\mathcal{N}_t$, the final object relationship feature $\mathcal{Q}_t$ is computed utilizing the equation:

$$\mathcal{Q}_t = \alpha_1\mathcal{O}_t + \alpha_2\mathcal{M}_t + \alpha_3\mathcal{N}_t, \tag{6}$$

where α_1, α_2, and α_3 are fused weights. Subsequently, a concatenation of $\mathcal{Q}_t$ and the image feature $\mathcal{R}_t$ is performed to yield the panoramic feature $\mathcal{F}_t = [\mathcal{R}_t, \mathcal{Q}_t]$

C. Turning Back Penalty

In our framework, the agent incrementally constructs a topological map where each node represents a visited location and stores the fused feature composed of $\mathcal{N}_t$, $\mathcal{M}_t$, $\mathcal{O}_t$, and $\mathcal{R}_t$. At each step, the action prediction network—composed of a Fine-scale Encoder and a Coarse-scale Encoder—predicts the transition probabilities from the current node to all candidate nodes in the map. The agent will select the node with the highest score as the next navigation target.

While executing navigation actions based on this topological map, we observed that the agent tends to revisit the same location multiple times. This usually culminates in an elongated navigation path. For instance, an agent's journey initiates at point a and concludes at point e, successfully navigating through the route $a \rightarrow b \rightarrow c \rightarrow d \rightarrow b \rightarrow e$. It can be observed that the direct path composed of $a \rightarrow b \rightarrow e$ would be optimal. The exploration of vertices c and d represents additional exploration by the agent, while revisiting vertex b indicates redundant exploration. Such additional exploration is beneficial as it enables the agent to acquire new knowledge, thereby enhancing its navigational skills. Conversely, redundant exploration does not improve navigation efficiency and leads to unnecessarily prolonged paths, which is disadvantageous.

To address the issue of the agent frequently revisiting the same location, we have developed a new loss function, named Turning Back Penalty (TBP). This function introduces a punitive measure to discourage the agent from redundant navigation, fostering a more streamlined and direct trajectory. Concretely, let us consider a scenario wherein the agent is positioned at location a, and it has a set of r navigable positions, denoted as $\{b_1, b_2, \dots, b_r\}$.

$$L_{TBP} = \sum_{i=1}^{r} \frac{e^{p_i} d_i}{\sum_{j=1}^{r} e^{p_j}}, \tag{7}$$

where p_i symbolizes the probability of transitioning from location a to location b_i, and d_i represents the cumulative length of the paths that have been traversed repetitively by the agent in the course of navigating from a to b_i.

D. Training and Inference

1) Pretrainging: Previous work [42], [59], [60] has demonstrated the effectiveness of pretraining transformer-based models in Vision-and-Language Navigation (VLN). Following this general paradigm, we design four auxiliary tasks tailored to our proposed model to perform pretraining.

A. Masked Language Modeling (MLM): Following the BERT-style [55] setup, we randomly mask 15% of the instruction tokens and train the model to recover the masked words using the surrounding context.

B. Masked Region Classification (MRC): For MRC [61], we randomly mask 15% of the visual inputs and require the model to predict their semantic labels. Target labels are generated using an image classification model [62] pretrained on ImageNet, following a similar strategy to [16].

C. Single-step Action Prediction (SAP): In the SAP task [63], the agent learns to predict its next action based on the past trajectory. The SAP loss in behavior cloning given a demonstration path P^* is as follows:

$$L_{SAP} = \sum_{t=1}^{T} -\log p\left(a_t^* \mid \mathcal{W}, \mathcal{P}_{<t}^*\right), \tag{8}$$

where $\mathcal{P}^*_{<t}$ represents a partial demonstration path, a^*_t is the expert action of $\mathcal{P}^*_{<t}$.

D. Object Grounding (OG): For tasks with object annotations, the object grounding [64] loss is employed.

$$L_{OG} = -\log p\left(o^* \mid \mathcal{W}, \mathcal{P}_T\right), \tag{9}$$

where o^* refers to the object category at the agent's final destination $\mathcal{P}_T$.

2) Fine-Tuning and Inference: For downstream training, we fine-tune the model using a combination of three loss terms: Single-step Action Prediction (SAP), Object Grounding (OG), and our proposed Turning Back Penalty (TBP). Unlike the pretraining phase, which uses demonstration trajectories, fine-tuning is guided by a pseudo-interactive demonstrator. This demonstrator dynamically selects the next node based on shortest-path computation at each decision point, ensuring minimum remaining path length to the goal. This selection is made such that it adheres to the criterion of minimizing the overall path length from the agent's current location to the target destination. The cumulative loss function that governs the fine-tuning process is formulated as follows:

$$L = \lambda_1 L_{SAP} + \lambda_2 L_{OG} + \lambda_3 L_{TBP}, \tag{10}$$

In this expression, λ_1, λ_2, and λ_3 serve as balance factors, ensuring a harmonious integration of the individual loss components.

During inference, the agent predicts one action per time step. If the predicted action is not a stop action, the agent executes it and moves to the corresponding location. If the action is stop, or if the maximum number of steps is exceeded, the agent terminates and selects the node with the highest stop probability as the final location. At the end of navigation, the target object is selected as the one with the highest predicted grounding score.

IV. Experiments

A. Datasets

In this paper, we focus on using object relationships to improve agent performance in vision-and-language navigation. To evaluate our model, we select two datasets that provide object annotations: REVERIE [6] and SOON [31]. We also report results on the Touchdown [32] dataset, which represents urban navigation scenarios. Additionally, we present the results of our model on the R2R [4] dataset, which lacks object annotations.

REVERIE dataset mainly consists of instructions that describe target locations and objects of interest, averaging 21 words per instruction. It provides the agent with bounding boxes for each object in various panoramas. The agent must correctly identify and choose the right object bounding box at the end of its navigation. Paths demonstrated by experts in this dataset vary in length, ranging from 4 to 7 steps.

SOON dataset provides detailed instructions that accurately identify target rooms and objects, averaging 47 words in length. Unlike other datasets, SOON does not include predefined bounding boxes for objects. This requires the agent to predict objects' central locations within the panoramas. To facilitate this, we utilize an automatic object detection approach as described in [53]. This method helps us generate potential bounding boxes for the objects. The lengths of expert demonstrations in SOON are varied, ranging from 2 to 21 steps, with an average of approximately 9.5 steps.

Touchdown is a vision-and-language navigation dataset built on Google Street View in urban environments. It consists of 29,641 panoramic images collected from Manhattan, along with the corresponding connectivity graph. We follow the data split used in ORAR [65], which includes both seen and unseen environments. In seen environments, the training and testing instances share overlapping regions, while in unseen environments, there is no such overlap. For the seen split, the train, validation, and test sets contain 6,525, 1,391, and 1,409 instances, respectively. For the unseen split, the train, validation, and test sets contain 6,770, 800, and 1,507 instances, respectively.

R2R dataset encompasses a total of 21,567 words, with an average instruction length of 29 words. Due to the absence of object annotations in this dataset, we substitute object features with features from panoramic images. In experiments, we solely train the agent using temporal object relations.

B. Evaluation Metrics

1) Indoor Environment: To assess the performance of our method in comparison to previous works, we adopt the conventional evaluation metrics for visual-and-language navigation task, as delineated in [4] and [6]. These metrics encompass: (1) Trajectory Length (TL)—the agent's average path length in meters; (2) Navigation Error (NE)—average distance in meters between agent's final location and the target; (3) Success Rate (SR)—the percentage of instructions that are successfully executed, with an NE smaller than 3 meters; (4) Oracle SR (OSR)—SR given the oracle stop policy; (5) SPL—SR weighted by Path Length; (6) Remote Grounding Success (RGS)—the percentage of instructions that are executed successfully.; (7) RGSPL—RGS penalized by Path Length. Except for TL and NE, all metrics are higher the better.

2) Urban Environment: To evaluate navigation performance in urban settings, we employ three widely used metrics [32], [65], [66]: Task Completion (TC), Shortest-path Distance (SPD), and Success weighted by Edit Distance (SED). TC indicates the proportion of tasks where the agent successfully reaches the goal. SPD measures how close the agent stops to the target location in terms of the shortest path within the environment graph. SED reflects task success while considering the similarity between predicted and ground-truth paths based on the Levenshtein edit distance.

C. Implementation Details

1) Model Architectures: For the REVERIE dataset, we use the ViT-B/16 model [62] pretrained on ImageNet to extract object features, as it provides bounding boxes. For the SOON dataset, which lacks bounding box annotations, we use the BUTD object detector [53] to obtain object regions. For the Touchdown dataset, we extract object features at each

TABLE I

COMPARISON WITH THE STATE-OF-THE-ART METHODS ON THE REVERIE DATASET. THE BASELINE INDICATES THE REPLICATED RESULTS OF DUET

Methods	Val Seen						Val Unseen						Test Unseen					
	Navigation				Grounding		Navigation				Grounding		Navigation				Grounding	
	TL↓	OSR↑	SR↑	SPL↑	RGS↑	RGSPL↑	TL↓	OSR↑	SR↑	SPL↑	RGS↑	RGSPL↑	TL↓	OSR↑	SR↑	SPL↑	RGS↑	RGSPL↑
Seq2Seq [4]	12.88	35.70	29.59	24.01	18.97	14.96	11.07	8.07	4.20	2.84	2.16	1.63	10.89	6.88	3.99	3.09	2.00	1.58
RCM [38]	10.70	29.44	23.33	21.82	13.23	15.36	11.98	14.23	9.39	6.97	4.89	3.89	10.60	11.68	7.84	6.67	3.67	3.14
VLNBERT [11]	13.44	53.90	51.79	47.96	38.23	35.61	16.78	35.02	30.67	24.90	18.77	15.27	15.68	32.91	29.61	23.99	16.50	13.51
AirBERT [3]	15.16	49.98	47.01	42.34	32.75	30.01	18.71	34.51	27.89	21.88	18.23	14.18	17.91	34.20	30.28	23.61	16.83	13.28
HOP [36]	13.80	54.88	53.76	47.19	38.65	33.85	16.46	36.24	31.78	26.11	18.85	15.73	16.38	33.06	30.17	24.34	17.69	14.34
HAMT [63]	12.79	47.65	43.29	40.19	27.20	15.18	14.08	36.84	32.95	30.20	18.92	17.28	13.62	33.41	30.40	26.67	14.88	13.08
CKR [26]	12.16	61.91	57.27	53.57	39.07	-	26.26	31.44	19.14	11.84	11.45	-	22.46	30.40	22.00	14.25	11.60	-
DUET [16]	13.86	73.68	71.75	63.94	57.41	51.14	22.11	51.07	46.98	33.73	32.15	23.03	21.30	56.91	52.51	36.06	31.88	22.06
KERM [17]	12.84	79.20	76.88	70.45	61.00	56.07	21.85	55.21	50.44	35.38	34.51	24.45	17.32	57.58	52.43	39.21	32.39	23.64
BEVBert [69]	-	-	-	-	-	-	-	56.40	**51.78**	36.37	34.71	24.44	-	57.26	52.81	36.41	32.06	22.09
GridMM [18]	-	-	-	-	-	-	23.20	**57.48**	51.37	36.47	34.57	24.56	19.97	59.55	53.13	36.60	34.87	23.45
Baseline	13.84	73.79	71.68	63.90	57.34	51.11	22.12	51.09	46.98	33.75	32.15	23.05	18.19	55.31	50.67	36.27	31.87	22.65
Ours	14.00	**83.06**	**80.46**	**73.12**	**64.02**	**58.29**	22.00	55.55	50.30	**36.84**	**35.27**	**25.98**	17.61	**61.08**	**55.31**	**40.37**	**35.16**	**24.99**

viewpoint using the Mask R-CNN model pretrained on the COCO dataset. For the R2R dataset, we do not employ the spatial object relations module because of the unavailability of object annotations.

In the indoor environment, we incorporate a dual-scale graph transformer [16]. The specific configuration of this transformer includes setting the number of layers for the language encoder, panorama encoder, coarse-scale cross-modal encoder, and fine-scale cross-modal encoder to 9, 2, 4, and 4, respectively. The parameters for this segment of our model are initialized using the pretrained LXMERT model [56]. In the urban environment, instructions are encoded using a bidirectional LSTM [67] that generates token-level features. Visual representations are obtained from a ResNet [68] pretrained on ImageNet.

In the process of computing the relationship matrix $\mathbf{E}$, we assigned specific values to the parameters k_1, k_2, and k_3, setting them at 2, 2, and $5e^{-4}$, respectively. Furthermore, in order to compute the object relationship feature $\mathcal{Q}$, the fused weights α_1, α_2, and α_3 were set to 0.8, 0.1, and 0.1, respectively. Due to lacking spatial object relations module, we set α_1, α_2, and α_3 as 0.8, 0.2, and 0 when using R2R dataset. Analogously, for the precise calculation of the loss function L, we established the values of the weight parameters λ_1, λ_2, and λ_3 at 1, 1, and 0.2, respectively.

2) Training Details: For the three indoor datasets, we perform pretraining with a batch size of 32 using a single NVIDIA RTX 3090 GPU. For the outdoor dataset, no pretraining is applied. For the REVERIE dataset, we combine the original dataset with augmented data synthesized by DUET [16] to pretrain our model with 100k iterations. Then we fine-tune the pretrained model with the batch size of 16 for 20k iterations on 1 NVIDIA RTX3090 GPU. For the SOON dataset, we only use the original data with automatically cleaned object bounding boxes, sharing the same settings in DUET [16]. We pretrain the model with 40k iterations. Then we fine-tune the pretrained model with the batch size of 4 for 40k iterations on 1 NVIDIA RTX3090 GPU. For the Torchdown dataset, we use a batch size of 32 for training. Dropout with a rate of 0.3 is applied after each dense layer and recurrent connection. For the R2R dataset, additional augmented R2R data in [42] is used in pretraining. We pretrain the model for 200k iterations with batch size of 64 and then fine-tune it for 20k iterations with batch size of 8.

3) Graph Construction: We follow the standard navigation graph defined by the Matterport3D [54] simulator, where each node corresponds to a predefined panoramic viewpoint with a fixed 3D position and heading. During navigation, the agent incrementally constructs an undirected graph using visited and adjacent nodes, which are treated as candidate locations for decision-making. Since each location provides a 360-degree panoramic view, object appearance is determined by the current position rather than the movement direction.

D. Performance Comparison

The results of our method on the REVERIE and SOON datasets are depicted in Table I and Table II, respectively. Across most metrics on these two datasets, our approach achieves superior performance. Specifically, in the test split of REVERIE, as detailed in Table I, our method achieves notable improvements over the baseline: 4.64% on SR, 4.10% on SPL, 3.29% on RGS, and 2.34% on RGSPL. This substantial enhancement underscores the robustness and efficacy of our technique. Moreover, even when compared to the current state-of-the-art method, GridMM [18], our method still shows advancements of 2.18%, 3.77%, 0.29%, and 1.54% on SR, SPL, RGS, and RGSPL, respectively, highlighting the superior capability of our approach. As indicated in Table II, on the more intricate SOON dataset, our method also manifests exceptional performance, surpassing the current state-of-the-art. This underscores our method's proficiency in grasping inter-object relations, thereby enhancing the agent's navigational prowess.

To evaluate the effectiveness of our method in urban environments, we conducted experiments on the Touchdown dataset. As shown in Table III, our approach achieves significant improvements over the baseline. In unseen environments, our method improves the TC score by 2.4% on the validation set and 2.1% on the test set. In addition, compared to the current state-of-the-art method VLN-VIDEO [66], which relies on additional augmented data and auxiliary proxy tasks for pretraining, our method still achieves 0.6% and 0.8% higher TC on the unseen validation and test sets. In seen environments, our method also yields competitive results. These findings demonstrate the effectiveness and generalizability of our approach in urban navigation scenarios.

In our study, we also conducted experiments on the R2R dataset, which lacks object annotations. As shown in Table IV,

TABLE II

COMPARISON WITH THE STATE-OF-THE-ART METHODS ON THE SOON DATASET. THE BASELINE INDICATES THE REPLICATED RESULTS OF DUET

Methods	Val Unseen					Test Unseen				
	TL↓	OSR↑	SR↑	SPL↑	RGSPL↑	TL↓	OSR↑	SR↑	SPL↑	RGSPL↑
GBE [70]	28.96	28.54	19.52	13.34	1.16	27.88	21.45	12.90	9.23	0.45
DUET [16]	36.20	50.91	36.28	22.58	3.75	41.83	43.00	33.44	21.42	4.17
KERM [17]	35.83	51.62	38.05	23.16	4.04	-	-	-	-	-
GridMM [18]	38.92	53.39	37.46	24.81	3.91	46.20	**48.02**	36.27	21.25	4.15
Baseline	36.18	50.88	36.17	22.54	3.75	40.73	41.74	32.05	20.79	4.55
Ours	38.75	**55.46**	**40.12**	**26.00**	**5.04**	40.05	47.73	**37.09**	**23.60**	**6.35**

TABLE III

COMPARISON WITH THE STATE-OF-THE-ART METHODS ON THE TOUCHDOWN DATASET. THE BASELINE INDICATES THE REPLICATED RESULTS OF ORAR. THE * DENOTES THAT THE MODEL WAS PRE-TRAINED WITH ADDITIONAL AUGMENTED DATA AND AUXILIARY PROXY TASKS

Methods	Seen						Unseen					
	Val Set			Test Set			Val Set			Test Set		
	TC↑	SPD↓	SED↑	TC↑	SPD↓	SED↑	TC↑	SPD↓	SED↑	TC↑	SPD↓	SED↑
GA [71]	9.9	21.4	9.5	9.7	21.5	9.2	-	-	-	-	-	-
RCONCAT [71]	11.1	19.9	10.8	9.7	21.7	9.5	-	-	-	-	-	-
ARC + L2STOP [71]	19.5	17.1	19.0	16.7	18.8	16.3	-	-	-	-	-	-
ORAR [65]	30.1	11.1	29.5	29.6	11.8	28.9	16.9	19.6	16.2	15.1	20.5	14.3
ORAR-BERT [66]	30.6	10.3	29.9	30.0	11.3	29.0	17.5	20.6	16.8	15.7	21.6	15.0
VLN-VIDEO* [66]	**34.5**	**9.6**	**33.5**	31.7	11.2	31.0	18.2	20.2	17.5	16.3	21.2	15.7
Baseline	31.3	11.9	30.7	28.0	11.9	27.4	16.4	21.1	16.0	15.0	20.9	14.3
Ours	34.0	10.5	33.1	**31.8**	**10.9**	**31.1**	**18.8**	**19.5**	**17.9**	**17.1**	**19.8**	**16.5**

TABLE IV

COMPARISON WITH THE STATE-OF-THE-ART METHODS ON THE R2R DATASET. THE BASELINE INDICATES THE REPLICATED RESULTS OF DUET

Methods	Val seen				Val Unseen				Test Unseen			
	TL↓	NE↓	SR↑	SPL↑	TL↓	NE↓	SR↑	SPL↑	TL↓	NE↓	SR↑	SPL↑
Seq2Seq [4]	11.33	6.01	39	-	8.39	7.81	22	-	8.13	7.85	20	18
RCM [38]	-	3.33	70	67	-	5.28	55	50	-	5.15	55	51
PREVALENT [42]	10.32	3.67	69	65	10.19	4.71	58	53	10.51	5.30	54	51
EntityGraph [34]	10.13	3.47	67	65	9.99	4.73	57	53	10.29	4.75	55	52
VLNBERT [11]	11.13	2.90	72	68	12.01	3.93	63	57	12.35	4.09	63	57
AirBERT [3]	11.09	2.68	75	70	11.78	4.01	62	56	12.41	4.13	62	67
HOP [36]	11.26	2.72	75	70	12.27	3.80	64	57	12.68	3.83	64	59
HAMT [63]	-	-	-	-	11.46	2.29	66	61	12.27	3.93	65	60
DUET [16]	-	-	-	-	13.94	3.31	72	60	14.73	3.65	69	59
KERM [17]	12.16	2.19	79.73	73.79	13.54	3.22	71.95	60.91	14.60	3.61	69.73	59.25
BEVBert [69]	-	-	-	-	-	**2.81**	75	64	-	**3.13**	73	62
GridMM [18]	-	-	-	-	13.27	2.83	**75**	**64**	14.43	3.35	**73**	62
Baseline	13.41	2.35	79	72	13.92	3.25	71	60	15.2	3.42	70	60
Ours	12.67	**2.05**	**82**	**76**	13.57	3.05	71	61	15.4	3.27	72	**62**

the absence of explicit object information impeded the agent's ability to accurately learn the relationships between objects in the navigation environment. Our method does not exhibit significant improvements over other methods in both the val unseen and test splits across various metrics. Interestingly, we observes a remarkable phenomenon: on the val seen split, our method significantly outperforms both the baseline and other approaches. This can be attributed to the agent's repeated exposure to the panoramas in the val seen split during training. Despite the lack of explicit object annotations, the agent often manages to infer the presence and relationships of objects based on these panoramic features. These findings further illustrate the vital role of object relationships in enabling an agent to accurately complete navigation tasks. This aspect of our research highlights the significance of understanding and integrating object interactions within the navigational context for improved agent performance.

E. Ablation Study

1) Ablation of Object Relations: To explore the effects of our proposed modules, TOR and SOR, on the agent's navigation skills, we integrated them separately into the baseline method. We then conducted experiments with these integrations on the val unseen split of the REVERIE and SOON datasets. As illustrated in Table V, both TOR and SOR notably enhance the navigation performance of the agent. However, we observed that SOR contributes to a more modest improvement in navigation performance compared to TOR. This is attributed to the limited scale of the current datasets used for training the

TABLE V

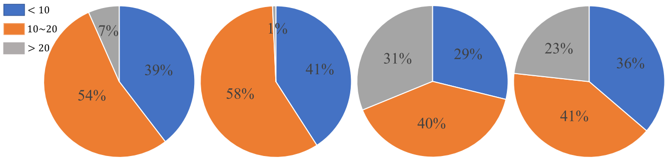
ABLATION OF THE OBJECT RELATIONS ON THE VAL UNSEEN SPLIT OF REVERIE AND SOON. IN THIS TABLE, TOR AND SOR RESPECTIVELY REPRESENT THE TRAJECTORY-OBJECT RELATIONSHIPS MODULE AND THE ENVIRONMENT-OBJECT RELATIONSHIPS MODULE. ALL EXPERIMENTS ARE CONDUCTED WITHOUT UTILIZING TBP LOSS FUNCTION

Dataset	TOR	SOR	TL↓	OSR↑	SR↑	SPL↑	RGS↑	RGSPL↑
REVERIE	×	×	22.12	51.09	46.98	33.75	32.15	23.05
	×	✓	22.31	52.85	48.08	33.77	32.55	23.07
	✓	×	23.07	**55.44**	49.22	33.32	33.46	22.90
	✓	✓	23.52	54.47	**49.64**	**34.56**	**34.05**	**23.41**
SOON	×	×	36.18	50.88	36.17	22.54	6.02	3.75
	×	✓	37.07	51.92	38.05	**25.31**	5.46	3.51
	✓	×	36.46	54.28	39.09	25.13	7.08	**4.61**
	✓	✓	39.91	**56.49**	**40.71**	24.98	**7.37**	4.56

TABLE VI

ABLATION OF TBP LOSS ON THE VAL UNSEEN SPLIT OF REVERIE AND SOON. BOTH THE TOR AND SOR MODULES ARE EMPLOYED IN THESE EXPERIMENTS

Dataset	TBP	TL↓	OSR↑	SR↑	SPL↑	RGS↑	RGSPL↑
REVERIE	×	23.52	54.47	49.64	34.56	34.05	23.41
	✓	22.00	**55.55**	**50.30**	**36.84**	**35.27**	**25.98**
SOON	×	39.91	**56.49**	**40.71**	24.98	7.37	4.56
	✓	38.75	55.46	40.12	**26.00**	**7.67**	**5.04**



Without TBP loss on the val seen set. With TBP loss on the val seen set. Without TBP loss on the val unseen set. With TBP loss on the val unseen set.

Fig. 4. The distribution of trajectory lengths predicted on the val seen and val unseen splits of REVERIE dataset.

agent. Solely relying on the objects observed at each position within the environment is inadequate to accurately capture the relationships among different objects. When both modules are employed in tandem, the navigational prowess of the agent is further amplified.

2) Ablation of TBP Loss: In Table V, we observes an intriguing phenomenon. The inclusion of object relationships does indeed significantly enhance the agent's success rates in navigation (OSR, SR, and RGS). However, this enhancement also leads to an increase in the trajectory length (TL) of the agent. As a result, there is no pronounced improvement in metrics such as SPL and RGSPL. This suggests that the object relationship module leads the agent to engage in excessive redundant exploration, resulting in elongated navigation paths. Upon integrating the TBP loss function, we observes a significant reduction in the agent's revisits to the same location. This is illustrated in Table VI. This change leads to a more efficient task completion and shows clear improvements in metrics such as TL, SPL.

Additionally, Fig. 4 depicts the distribution of navigational path lengths corresponding to successful navigation instances (specifically when SR is 1) for both the val seen and val unseen splits of the REVERIE dataset, comparing scenarios with and without the integration of the TBP loss function. The figure reveals that with the TBP loss, there's a notable increase in the proportion of agent paths falling between 0 and 10, while paths longer than 20 significantly diminish. This solidly validates that our TBP loss function effectively curtails the navigation path length of the agent.

TABLE VII

THE RESULTS OF PUNISHING TURNING BACK DURING INFERENCE. $\xi = i$ MEANS DIVIDING THE PROBABILITY OF THE AGENT REACHING A CERTAIN POSITION BY i

No.	ξ	TL↓	OSR↑	SR↑	SPL↑	RGS↑	RGSPL↑
1	0.5	28.10	**59.24**	38.00	22.17	27.78	16.66
2	1	22.00	55.55	**50.30**	**36.79**	**35.27**	**25.98**
3	2	22.53	51.32	47.71	32.91	32.06	22.22
4	4	22.01	49.53	46.49	32.35	31.47	21.91
5	6	21.90	48.79	45.81	32.03	31.10	21.75
6	8	21.84	48.59	45.61	31.93	30.99	21.73

TABLE VIII

ABLATION OF ALL RATIOS IN EQ.(3) ON THE VAL UNSEEN SPLIT OF REVERIE DATASET

k_1	k_2	k_3	TL↓	OSR↑	SR↑	SPL↑	RGS↑	RGSPL↑
2	2	$5e^{-4}$	22.00	**55.55**	**50.30**	**36.84**	**35.27**	**25.98**
20	2	$5e^{-4}$	23.42	53.39	49.25	34.54	33.20	23.51
2	20	$5e^{-4}$	24.76	54.59	49.45	33.74	33.91	23.06
2	2	$5e^{-3}$	23.54	53.56	50.13	35.10	35.02	24.38

3) Punish Turning Back During Inference: It is also a intuitive way punish turning back during inference. To assess whether penalizing the agent's repetitive visiting behavior during inference improves its navigational abilities, we have designed several experiments. The experimental outcomes are presented in Table VII.

Our findings indicate that penalizing the agent's repetitive visiting behavior during inference does not enhance its navigational performance. As evidenced in the last five rows of Table VII, the navigation success rate of the agent decreases with increasing penalty intensity. This decline in performance is attributed to the fact that such penalties during inference prevent the agent from correcting its navigational errors. Additionally, the first two rows of Table VII reveal that when repetitive visits by the agent are encouraged, there is a more significant drop in navigational ability. This is due to the agent engaging in more unproductive exploration, substantially increasing the length of the navigational path.

4) Ablation of All Ratios in Equation 3: To investigate the impact of the constants k_1, k_2, and k_3 in eq. (3), we conduct an ablation study on the REVERIE validation unseen split, as shown in Table VIII. In eq. (3), $\|\boldsymbol{d}_x - \boldsymbol{d}_y\|_2$ is approximately 4000 times larger than $\|\boldsymbol{v}_x - \boldsymbol{v}_y\|_2$. To balance their contributions in the denominator, we set k_3 to a small value of $5e^{-4}$ in our implementation.

The first row of Table VIII ($k_1 = 2$, $k_2 = 2$, $k_3 = 5e^{-4}$) achieves the best overall performance across all evaluation metrics. Increasing k_1 amplifies the relation scores uniformly, which weakens the relative differences and leads to performance degradation. Similarly, increasing k_2 or k_3 suppresses the contribution of view or depth differences, disrupting the balance between the two cues. These results confirm the rationality of our chosen parameter configuration.

TABLE IX

Ablation of Different Features on the Val Unseen Split of REVERIE Dataset

$\mathcal{O}_t$	$\mathcal{Q}_t$	$\mathcal{R}_t$	TL↓	OSR↑	SR↑	SPL↑	RGS↑	RGSPL↑
✓	✓	×	29.84	39.73	34.37	21.11	22.83	13.44
✓	×	✓	23.89	50.33	44.82	30.78	24.48	16.88
×	✓	✓	25.64	55.52	49.11	33.24	33.29	22.42
✓	✓	✓	22.00	**55.55**	**50.30**	**36.84**	**35.27**	**25.98**

TABLE X

Ablation of the Fused Weights on the Val Unseen Split of REVERIE Dataset

α_1	α_2	α_3	TL↓	OSR↑	SR↑	SPL↑	RGS↑	RGSPL↑
0.8	0.1	0.1	22.00	**55.55**	**50.30**	**36.84**	**35.27**	**25.98**
0.6	0.2	0.2	24.98	53.22	47.97	32.48	32.43	21.70
0.4	0.3	0.3	23.60	50.81	46.95	33.34	32.58	23.10
0.2	0.4	0.4	26.39	55.47	48.51	32.04	32.43	21.36

TABLE XI

Ablation of the Loss Weights on the Val Unseen Split of REVERIE Dataset

λ_1	λ_2	λ_3	TL↓	OSR↑	SR↑	SPL↑	RGS↑	RGSPL↑
1.0	1.0	0.0	23.52	54.47	49.64	34.56	34.05	23.41
1.0	1.0	0.2	22.00	**55.55**	**50.30**	**36.79**	**35.27**	**25.98**
1.0	1.0	0.5	21.34	49.59	46.01	33.31	31.01	22.35
1.0	1.0	1.0	20.75	51.66	46.21	33.09	29.37	21.24

TABLE XII

The Evaluation of Inference Efficiency. All Experiments Are Conducted on the REVERIE Validation Unseen Split, With a Batch Size of 4 and 3,521 Evaluation Samples. A Single NVIDIA RTX 3090 GPU is Used

Methods	Total Time (s)	GPU Memory (MB)
Baseline	877.87	4382
Ours	942.24	4462
Overhead	+7.33%	+1.83%

TABLE XIII

Comparison of Different Functional Forms in the TBP Loss on the REVERIE Validation Unseen Split

Functional forms	TL↓	OSR↑	SR↑	SPL↑	RGS↑	RGSPL↑
Linear Function	23.15	51.60	46.80	32.27	32.52	22.69
Square Function	24.76	54.10	47.00	31.11	32.26	21.35
Exponential (Ours)	22.00	**55.55**	**50.30**	**36.84**	**35.27**	**25.98**

5) Ablation of Different Features: To assess the importance of each feature in our model, we performed an ablation study by selectively removing $\mathcal{O}_t$, $\mathcal{Q}_t$, or $\mathcal{R}_t$. As shown in Table IX, removing any of the three features leads to a noticeable drop in performance. Specifically, excluding $\mathcal{Q}_t$ results in a decrease in SR and SPL by 5.48% and 6.06%, respectively. When $\mathcal{R}_t$ is removed, the model shows the worst overall performance, confirming the importance of panoramic visual features. Removing $\mathcal{O}_t$ causes a smaller decline in performance, as object-related cues have already been partially learned by the TOR and SOR modules. These results demonstrate that $\mathcal{O}_t$, $\mathcal{Q}_t$, and $\mathcal{R}_t$ are all essential and contribute jointly to navigation accuracy.

6) Ablation of the Fused Weights: We have conducted an ablation study to evaluate the impact of different fusion weights α_1, α_2, and α_3 in eq. (6). As shown in Table X, the best overall performance is achieved when $\alpha_1 = 0.8$, $\alpha_2 = 0.1$, and $\alpha_3 = 0.1$, which is the setting used in our main experiments. Other settings lead to performance degradation across multiple metrics. This demonstrates that assigning a higher initial weight to $\mathcal{O}_t$ contributes more directly to navigation success, while $\mathcal{M}_t$ and $\mathcal{N}_t$, which are derived from object-related information, serve as complementary cues.

7) Ablation of the Loss Weights: To improve navigation efficiency, we introduce the Turning Back Penalty (TBP) to discourage unnecessary revisits. However, in some cases, revisiting previously visited locations is necessary for the agent to correct past mistakes. If the penalty is too strong, it may prevent the agent from making such corrections, ultimately reducing navigation success. To explore this trade-off, we conduct experiments with different values of the TBP loss weight λ_3. The results are shown in Table XI. We observe that setting $\lambda_3 = 0.2$ achieves the best overall performance across all metrics. When λ_3 increases to 0.5 or 1.0, the agent's performance drops significantly. This suggests that applying a moderate penalty for revisiting improves navigation efficiency. However, overly strong penalties may harm performance, as revisiting certain locations is sometimes necessary to correct earlier decisions and reach the goal successfully.

8) Inference Efficiency Evaluation: To evaluate the efficiency of our proposed method, we compared the inference time and GPU memory usage with the baseline model. As shown in Table XII, our method increases the total inference time by only 7.33% and GPU memory consumption by 1.83%. These results demonstrate that the additional cross-attention computation in the TOR module introduces minimal overhead. The performance gains achieved by our model come at a modest cost in computational resources, indicating its practical applicability.

9) Ablation of Different Functional Forms in the TBP Loss: To evaluate the impact of different functional forms in the TBP loss, we compare three variants: a linear function, a square function, and our proposed exponential function. As shown in Table XIII, the exponential function achieves the best overall performance across all metrics. Specifically, it improves SR by 3.5% and SPL by 4.57% compared to the linear function, and also outperforms the square function by noticeable margins. The exponential function is particularly effective because it smoothly emphasizes higher transition probabilities, allowing the model to more strongly penalize highly probable redundant revisits while maintaining gradient stability. In contrast, the linear and square functions apply weaker or more uniform penalties, leading to suboptimal trajectory optimization. These results demonstrate the advantage of using an exponential weighting strategy in the TBP formulation.

F. Robustness Analysis

To verify the stability and reliability of our method, we conducted robustness experiments on four benchmark datasets: REVERIE, SOON, Touchdown, and R2R. For each dataset, we ran both our method and the baseline five times using different random seeds (based on runtime timestamps), and reported

TABLE XIV

Robustness Evaluation on the REVERIE Dataset. Each Method is Run Five Times With Runtime Timestamps as Random Seeds, and the Mean and Standard Deviation are Reported

Split	Methods	TL↓	OSR↑	SR↑	SPL↑	RGS↑	RGSPL↑
Val Seen	Bseline	14.60 ± 0.59	74.49 ± 0.58	72.06 ± 0.74	63.59 ± 0.30	58.00 ± 0.78	51.03 ± 0.28
	Ours	12.53 ± 0.83	**80.76** ± 1.33	**79.34** ± 0.74	**74.15** ± 0.57	**62.70** ± 0.81	**59.00** ± 0.40
Val Unseen	Baseline	24.24 ± 1.26	53.04 ± 1.16	47.96 ± 0.70	33.26 ± 0.35	32.42 ± 0.19	22.56 ± 0.29
	Ours	21.54 ± 0.51	**54.25** ± 0.74	**50.91** ± 0.38	**36.93** ± 0.12	**34.87** ± 0.28	**25.02** ± 0.57

TABLE XV

Robustness Evaluation on the SOON Dataset. Each Method is Run Five Times With Runtime Timestamps as Random Seeds, and the Mean and Standard Deviation are Reported

Split	Methods	TL↓	OSR↑	SR↑	SPL↑	RGSPL↑
Val Unseen	Bseline	35.92 ± 2.19	51.10 ± 1.94	35.64 ± 0.74	23.06 ± 0.39	3.90 ± 0.34
	Ours	40.18 ± 1.34	**56.29** ± 0.75	**39.96** ± 1.05	**26.45** ± 1.08	**4.76** ± 0.27
Test Unseen	Baseline	39.92 ± 1.10	43.36 ± 1.04	33.46 ± 1.11	21.92 ± 0.79	4.89 ± 0.51
	Ours	41.94 ± 1.30	**45.58** ± 1.37	**36.50** ± 0.63	**23.87** ± 0.68	**5.87** ± 0.28

TABLE XVI

Robustness Evaluation on the Touchdown Dataset. Each Method is Run Five Times With Runtime Timestamps as Random Seeds, and the Mean and Standard Deviation are Reported

Scene	Methods	Val Set			Test Set		
		TC↑	SPD↓	SED↑	TC↑	SPD↓	SED↑
Seen	Baseline	29.98 ± 0.96	11.84 ± 0.22	29.29 ± 0.94	28.47 ± 0.74	12.12 ± 0.26	27.78 ± 0.72
	Ours	**33.58** ± 0.39	**10.64** ± 0.09	**32.70** ± 0.38	**31.34** ± 0.47	**11.10** ± 0.12	**30.66** ± 0.47
Unseen	Baseline	15.77 ± 0.68	20.10 ± 0.27	15.33 ± 0.72	15.02 ± 0.46	21.11 ± 0.35	14.36 ± 0.45
	Ours	**18.72** ± 0.65	**19.90** ± 0.29	**17.94** ± 0.62	**16.60** ± 0.70	**20.06** ± 0.34	**16.00** ± 0.70

TABLE XVII

Robustness Evaluation on the R2R Dataset. Each Method is Run Five Times With Runtime Timestamps as Random Seeds, and the Mean and Standard Deviation are Reported

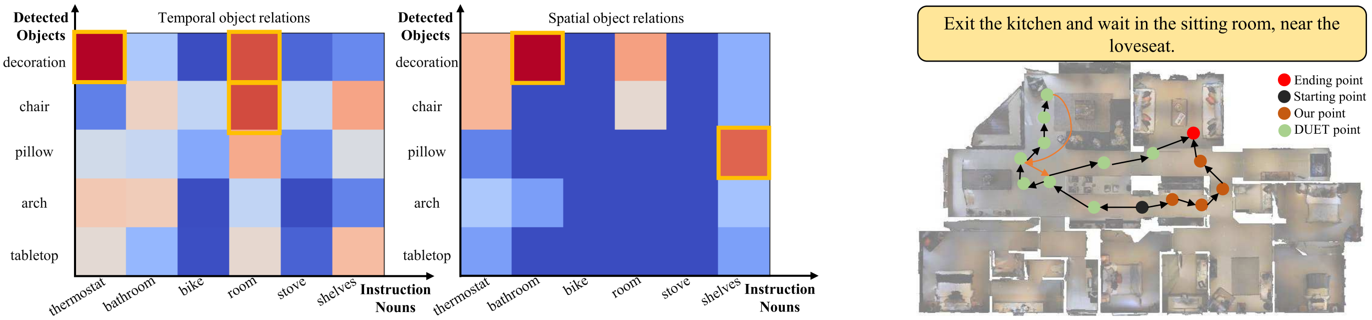
Split	Methods	TL↓	NE↓	SR↑	SPL↑
Val Seen	Bseline	13.18 ± 0.66	2.23 ± 0.08	79.58 ± 0.58	72.75 ± 1.11
	Ours	12.80 ± 0.18	**2.21** ± 0.15	**80.22** ± 1.26	**74.32** ± 1.26
Val Unseen	Baseline	14.31 ± 0.67	3.21 ± 0.10	**70.82** ± 0.75	59.11 ± 0.63
	Ours	14.01 ± 0.44	**3.15** ± 0.12	70.48 ± 0.79	**59.30** ± 0.86

TABLE XVIII

Robustness Evaluation of Different Methods on the REVERIE Dataset. Each Method is Run Five Times With Runtime Timestamps as Random Seeds, and the Mean and Standard Deviation are Reported

Split	Methods	TL↓	OSR↑	SR↑	SPL↑	RGS↑	RGSPL↑
Val Seen	DUET [16]	14.60 ± 0.59	74.49 ± 0.58	72.06 ± 0.74	63.59 ± 0.30	58.00 ± 0.78	51.03 ± 0.28
	KERM [17]	13.30 ± 0.22	77.01 ± 0.48	74.25 ± 0.43	67.61 ± 0.39	59.32 ± 0.31	53.97 ± 0.33
	BEVBert [69]	15.59 ± 1.08	80.30 ± 2.46	77.79 ± 2.28	67.82 ± 2.32	61.00 ± 1.91	53.31 ± 2.08
	GridMM [18]	15.98 ± 0.68	74.04 ± 2.06	71.74 ± 2.09	61.75 ± 1.58	56.37 ± 1.76	48.38 ± 1.45
	Ours	12.53 ± 0.83	**80.76** ± 1.33	**79.34** ± 0.74	**74.15** ± 0.57	**62.70** ± 0.81	**59.00** ± 0.40
Val Unseen	DUET [16]	24.24 ± 1.26	53.04 ± 1.16	47.96 ± 0.70	33.26 ± 0.35	32.42 ± 0.19	22.56 ± 0.29
	KERM [17]	22.44 ± 0.47	51.16 ± 1.26	48.89 ± 0.98	34.49 ± 0.21	32.81 ± 0.72	23.42 ± 0.37
	BEVBert [69]	26.99 ± 1.42	54.00 ± 1.91	47.92 ± 1.12	30.94 ± 0.95	32.82 ± 0.98	21.24 ± 0.56
	GridMM [18]	24.33 ± 2.07	**54.57** ± 1.26	48.62 ± 1.23	32.80 ± 2.38	32.45 ± 0.98	21.98 ± 1.83
	Ours	21.54 ± 0.51	54.25 ± 0.74	**50.91** ± 0.38	**36.93** ± 0.12	**34.87** ± 0.28	**25.02** ± 0.57

the mean and standard deviation of all major evaluation metrics in Table XIV to Table XVII. Across all datasets, our method consistently achieves better average performance than the baseline. For example, on the REVERIE validation unseen split, our method improves SR from 47.96 ± 0.70 to 50.91 ± 0.38, and SPL from 33.26 ± 0.35 to 36.93 ± 0.12, while also reducing the standard deviation. Similar trends can be observed on the SOON and Touchdown datasets,



(a) Attention maps from the Temporal Object Relations and Spatial Object Relations modules. Each cell represents the attention score between a detected object (y-axis) and an instruction noun (x-axis). The score is indicated from low to high with the color blue to red. Yellow boxes highlight top-attended pairs, showing how our model links objects to relevant instruction nouns.

(b) The navigation trajectories for a given instruction. Brown circles indicate the path taken by our method, while green circles represent the trajectory predicted by DUET. Orange lines highlight instances of backtracking behavior.

Fig. 5. Visualization of attention maps and navigation examples.

demonstrating that the performance gains are not only higher in magnitude but also more stable. On the R2R dataset, although the improvements are relatively smaller due to the absence of object annotations, our method still exhibits lower variance in most metrics, highlighting its robustness under different settings.

To further assess the consistency of our method beyond baseline comparison, we also evaluated it against other competitive approaches on the REVERIE dataset. As shown in Table XVIII, our method achieves the highest performance on most metrics with smaller or comparable standard deviations. These results collectively indicate that our method is robust across different random initializations and generalizes well across various datasets and evaluation metrics.

G. Qualitative Results

1) Visualization of Object Relations: To demonstrate that our method effectively captures object-noun associations, we visualize the attention heatmaps of the TOR and SOR modules after training. Fig. 5a shows the attention scores between detected objects and instruction nouns, where higher scores indicate stronger semantic relevance. The TOR module captures the temporal alignment between the agent's current observation and the instruction, while the SOR module complements it by modeling spatial dependencies among co-occurring objects. For instance, the SOR module identifies strong associations such as pillow–shelves or decoration–bathroom, which may not be highlighted in TOR but are crucial for understanding the local environment. These complementary attention patterns demonstrate that both modules collaboratively contribute to aligning visual objects with the semantic cues provided in the natural language instructions.

2) Visualization of the Navigation Trajectories: To elucidate the effectiveness of our proposed approach, we have rendered a comparative visualization of the navigation trajectories generated by both DUET and our method. As illustrated in Fig. 5b, both techniques can accurately reach navigation targets. However, DUET tends to involve the agent in excessive exploratory actions. This often results in repeated visits to the same places, hindering navigational efficiency. Conversely, our method substantially diminishes the agent's inclination to backtrack, facilitating the selection of more direct routes that expedite the completion of the navigation tasks. Furthermore, our analysis reveals that at the onset of navigation, our method opted for a proximal route, in contrast to DUET which embarked on a relatively longer path with a greater number of actions. This indicates that our method can effectively understand the relationships between objects encountered by the agent and the specified targets. Consequently, it identifies more efficient paths. This aids in completing navigational tasks more effectively.

V. Conclusion

In this study, we introduced Temporal-Spatial Object Relations Modules and a Turning Back Penalty (TBP) loss function that together enhance agent navigation. By learning the connections between various objects, the agent can more effectively complete navigation tasks. The application of the TBP loss function successfully prevents repetitive visits to the same location by the agent, thereby enhancing navigational efficiency. It is noteworthy that the object relationships we model might not be entirely accurate because of the limited datasets. Moving forward, our future endeavors will focus on devising more sophisticated object relationship modeling techniques, expanding dataset scales, and honing the precision and efficiency of navigational tasks.

References

[1] M. Nawaz, J. K.-T. Tang, K. Bibi, S. Xiao, H.-P. Ho, and W. Yuan, "Robust cognitive capability in autonomous driving using sensor fusion techniques: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 5, pp. 3228–3243, May 2024.

[2] C. Sun et al., "Toward ensuring safety for autonomous driving perception: Standardization progress, research advances, and perspectives," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 5, pp. 3286–3304, May 2024.

[3] P.-L. Guhur, M. Tapaswi, S. Chen, I. Laptev, and C. Schmid, "Airbert: In-domain pretraining for vision-and-language navigation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 1614–1623.

[4] P. Anderson et al., "Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3674–3683.

[5] W. Zhu et al., "Diagnosing vision-and-language navigation: What really matters," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Human Lang. Technol.*, 2022, pp. 5981–5993.

[6] Y. Qi et al., "REVERIE: Remote embodied visual referring expression in real indoor environments," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 9979–9988.

[7] W. Kang, J. Mun, S. Lee, and B. Roh, "Noise-aware learning from Web-crawled image-text data for image captioning," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2023, pp. 2930–2940.

[8] Y. Zhang, C.-H. Ho, and N. Vasconcelos, "Toward unsupervised realistic visual question answering," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2023, pp. 15567–15578.

[9] J. Qi, Y. Niu, J. Huang, and H. Zhang, "Two causal principles for improving visual dialog," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 10857–10866.

[10] Y. Qiao, C. Deng, and Q. Wu, "Referring expression comprehension: A survey of methods and datasets," *IEEE Trans. Multimedia*, vol. 23, pp. 4426–4440, 2021.

[11] Y. Hong, Q. Wu, Y. Qi, C. Rodriguez-Opazo, and S. Gould, "VLN BERT: A recurrent vision-and-language BERT for navigation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 1643–1653.

[12] D. Fried et al., "Speaker-follower models for vision-and-language navigation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 3318–3329.

[13] A. Moudgil, A. Majumdar, H. Agrawal, S. Lee, and D. Batra, "SOAT: A scene- and object-aware transformer for vision-and-language navigation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 7357–7367.

[14] H. Tan, L. Yu, and M. Bansal, "Learning to navigate unseen environments: Back translation with environmental dropout," in *Proc. Conf. North*, 2019, pp. 2610–2621.

[15] B. Lin, Y. Zhu, Z. Chen, X. Liang, J. Liu, and X. Liang, "ADAPT: Vision-language navigation with modality-aligned action prompts," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 15375–15385.

[16] 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 *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 16516–16526.

[17] X. Li, Z. Wang, J. Yang, Y. Wang, and S. Jiang, "KERM: Knowledge enhanced reasoning for vision-and-language navigation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 2583–2592.

[18] Z. Wang, X. Li, J. Yang, Y. Liu, and S. Jiang, "GridMM: Grid memory map for vision-and-language navigation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2023, pp. 15579–15590.

[19] H. Du, X. Yu, and L. Zheng, "Learning object relation graph and tentative policy for visual navigation," in *Eur. Conf. Comput. Vis.*, 2020, pp. 19–34.

[20] R. Dang, Z. Shi, L. Wang, Z. He, C. Liu, and Q. Chen, "Unbiased directed object attention graph for object navigation," in *Proc. 30th ACM Int. Conf. Multimedia*, Oct. 2022, pp. 3617–3627.

[21] R. Dang et al., "Search for or navigate to? Dual adaptive thinking for object navigation," in *Proc. IEEE Int. Conf. Comp. Vis. (ICCV)*, pp. 8250–8259, Oct. 2023.

[22] J. Chen, C. Gao, E. Meng, Q. Zhang, and S. Liu, "Reinforced structured state-evolution for vision-language navigation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 15429–15438.

[23] C.-W. Kuo and Z. Kira, "Beyond a pre-trained object detector: Cross-modal textual and visual context for image captioning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 17948–17958.

[24] R. Krishna et al., "Visual genome: Connecting language and vision using crowdsourced dense image annotations," *Int. J. Comput. Vis.*, vol. 123, no. 1, pp. 32–73, May 2017.

[25] R. E. Speer, J. Chin, and C. Havasi, "ConceptNet 5.5: An open multilingual graph of general knowledge," in *Proc. AAAI Conf. Artif. Intell.*, vol. 31, 2017, pp. 4444–4451.

[26] C. Gao, J. Chen, S. Liu, L. Wang, Q. Zhang, and Q. Wu, "Room-and-object aware knowledge reasoning for remote embodied referring expression," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 3063–3072.

[27] R. Schumann, W. Zhu, W. Feng, T.-J. Fu, S. Riezler, and W. Yang Wang, "VELMA: Verbalization embodiment of LLM agents for vision and language navigation in street view," 2023, *arXiv:2307.06082*.

[28] G. Zhou, Y. Hong, and Q. Wu, "NavGPT: Explicit reasoning in vision-and-language navigation with large language models," 2023, *arXiv:2305.16986*.

[29] D. Shah, B. Osiński, B. Ichter, and S. Levine, "LM-nav: Robotic navigation with large pre-trained models of language, vision, and action," in *Proc. 6th Conf. Robot Learn.*, Dec. 2022, pp. 492–504.

[30] H. Wang, W. Wang, W. Liang, S. C. H. Hoi, J. Shen, and L. V. Gool, "Active perception for visual-language navigation," *Int. J. Comput. Vis.*, vol. 131, no. 3, pp. 607–625, Mar. 2023.

[31] F. Zhu, X. Liang, Y. Zhu, Q. Yu, X. Chang, and X. Liang, "SOON: Scenario oriented object navigation with graph-based exploration," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 12684–12694.

[32] H. Chen, A. Suhr, D. Misra, N. Snavely, and Y. Artzi, "TOUCHDOWN: Natural language navigation and spatial reasoning in visual street environments," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 12530–12539.

[33] X. Wang et al., "Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 6622–6631.

[34] Y. Hong, C. Rodrí;guez-Opazo, Y. Qi, Q. Wu, and S. J. Gould, "Language and visual entity relationship graph for agent navigation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 7685–7696.

[35] H. Wang, W. Wang, W. Liang, C. Xiong, and J. Shen, "Structured scene memory for vision-language navigation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2021, pp. 8455–8464.

[36] Y. Qiao, Y. Qi, Y. Hong, Z. Yu, P. Wang, and Q. Wu, "HOP: History-and-order aware pretraining for vision-and-language navigation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 15397–15406.

[37] S. Chen, P.-L. Guhur, M. Tapaswi, C. Schmid, and I. Laptev, "Learning from unlabeled 3D environments for vision-and-language navigation," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 638–655.

[38] F. Zhu, Y. Zhu, X. Chang, and X. Liang, "Vision-language navigation with self-supervised auxiliary reasoning tasks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 10009–10019.

[39] H. Liu, "Cooperative multi-agent game based on reinforcement learning," *High-Confidence Comput.*, vol. 4, no. 1, Mar. 2024, Art. no. 100205. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2667295224000084

[40] C.-Y. Ma, Z. Wu, G. AlRegib, C. Xiong, and Z. Kira, "The regretful agent: Heuristic-aided navigation through progress estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 6725–6733.

[41] X. Wang, W. Xiong, H. Wang, and W. Y. Wang, "Look before you leap: Bridging model-free and model-based reinforcement learning for planned-ahead vision-and-language navigation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 38–55.

[42] W. Hao, C. Li, X. Li, L. Carin, and J. Gao, "Towards learning a generic agent for vision-and-language navigation via pre-training," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 13134–13143.

[43] X. Li et al., "Robust navigation with language pretraining and stochastic sampling," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 1494–1499.

[44] H. Tian, J. Meng, W.-S. Zheng, Y.-M. Li, J. Yan, and Y. Zhang, "Loc4Plan: Locating before planning for outdoor vision and language navigation," in *Proc. 32nd ACM Int. Conf. Multimedia*, Oct. 2024, pp. 4073–4081, doi: 10.1145/3664647.3681518.

[45] Y. Zhang and P. Kordjamshidi, "LOViS: Learning orientation and visual signals for vision and language navigation," in *Proc. COLING*, 2022, pp. 5745–5754. [Online]. Available: https://aclanthology.org/2022.coling-1.505

[46] S. Zhang, X. Song, Y. Bai, W. Li, Y. Chu, and S. Jiang, "Hierarchical object-to-zone graph for object navigation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 15110–15120.

[47] T. Zhu, S. Liu, B. Li, J. Liu, P. Liu, and F. Zheng, "Graph reasoning over explicit semantic relation," *High-Confidence Comput.*, vol. 4, no. 2, Jun. 2024, Art. no. 100190. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2667295223000880

[48] Y. Zhang and P. Kordjamshidi, "Explicit object relation alignment for vision and language navigation," in *Proc. 60th Annu. Meeting Assoc. Comput. Linguistics, Student Res. Workshop*, 2022, pp. 322–331.

[49] Y. Qi, Z. Pan, S. Zhang, A. van den Hengel, and Q. Wu, “Object-and-action aware model for visual language navigation,” in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 303–317.

[50] C. Huang, O. Mees, A. Zeng, and W. Burgard, “Visual language maps for robot navigation,” in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2023, pp. 10608–10615.

[51] S. Ross, G. Gordon, and D. Bagnell, “A reduction of imitation learning and structured prediction to no-regret online learning,” in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, 2011, pp. 627–635.

[52] C. Ma et al., “Self-monitoring navigation agent via auxiliary progress estimation,” in *Proc. 7th Int. Conf. Learn. Represent.*, 2019, pp. 5613–5630.

[53] P. Anderson et al., “Bottom-up and top-down attention for image captioning and visual question answering,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6077–6086.

[54] A. Chang et al., “Matterport3D: Learning from RGB-D data in indoor environments,” in *Proc. Int. Conf. 3D Vis. (3DV)*, Oct. 2017, pp. 667–676.

[55] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 1, Jun. 2019, pp. 4171–4186.

[56] H. Tan and M. Bansal, “LXMERT: Learning cross-modality encoder representations from transformers,” in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 5099–5110.

[57] L. H. Li et al., “Grounded language-image pre-training,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 10955–10965.

[58] M. Minderer et al., “Simple open-vocabulary object detection,” in *Proc. ECCV*, 2022, pp. 728–755.

[59] J. Krantz et al., “Beyond the nav-graph: Vision-and-language navigation in continuous environments,” in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2020, pp. 104–120.

[60] A. Pashevich, C. Schmid, and C. Sun, “Episodic transformer for vision-and-language navigation,” in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 15922–15932.

[61] J. Lu, D. Batra, D. Parikh, and S. Lee, “ViLBERT: Pre-training task-agnostic visiolinguistic representations for vision-and-language tasks,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 13–23.

[62] A. Dosovitskiy et al., “An image is worth 16×16 words: Transformers for image recognition at scale,” in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2020, pp. 611–631.

[63] S. Chen, P.-L. Guhur, C. Schmid, and I. Laptev, “History aware multimodal transformer for vision-and-language navigation,” in *Proc. NeurIPS*, 2021, pp. 5834–5847.

[64] X. Lin, G. Li, and Y. Yu, “Scene-intuitive agent for remote embodied visual grounding,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 7032–7041.

[65] R. Schumann and S. Riezler, “Analyzing generalization of vision and language navigation to unseen outdoor areas,” in *Proc. 60th Annu. Meeting Assoc. Comput. Linguistics*, May 2022, pp. 7519–7532. [Online]. Available: https://aclanthology.org/2022.acl-long.518/

[66] J. Li, A. Padmakumar, G. S. Sukhatme, and M. Bansal, “VLN-video: Utilizing driving videos for outdoor vision-and-language navigation,” in *Proc. AAAI Conf. Artif. Intell.*, 2024, vol. 38, no. 17, pp. 18517–18526.

[67] A. Graves, S. Fernández, and J. Schmidhuber, “Bidirectional lstm networks for improved phoneme classification and recognition,” in *Proc. Int. Conf. Artif. Neural Netw.*, 2005, pp. 799–804.

[68] O. Russakovsky et al., “ImageNet large scale visual recognition challenge,” *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[69] D. An et al., “BEVBert: Multimodal map pre-training for language-guided navigation,” Dec. 2022, *arXiv:2212.04385*.

[70] M. Li, Z. Wang, T. Tuytelaars, and M. Moens, “Layout-aware dreamer for embodied visual referring expression grounding,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 37, 2023, pp. 1386–1395.

[71] J. Xiang, X. Wang, and W. Y. Wang, “Learning to stop: A simple yet effective approach to urban vision-language navigation,” in *Proc. Findings Assoc. Comput. Linguistics, EMNLP*, Nov. 2020, pp. 699–707. [Online]. Available: https://aclanthology.org/2020.findings-emnlp.62/

Bowen Huang received the B.E. degree in computer science and technology from Shandong University, Qingdao, China, in 2021, where he is currently pursuing the Ph.D. degree with the Department of Computer Science and Technology. His research interests include computer vision and visual-language navigation.

Yanwei Zheng (Member, IEEE) received the B.S. degree from Shandong Jianzhu University in 1999, the M.S. degree from Shandong University in 2004, and the Ph.D. degree from Beihang University in January 2019, supervised by Prof. Zhang Xiong. He is currently an Associate Professor at the Institute of Intelligent Computing (IIC), School of Computer Science and Technology, Shandong University. His research interests include computer vision, visual navigation, and digital twins.

Dongchen Sui received the B.S. degree in computer science and technology from The University of Sydney, Australia, in 2021. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Technology, Shandong University, Qingdao, China. His research interests include computer vision and visual-language navigation.

Chuanlin Lan received the B.E. degree from the School of Electronic Information, Wuhan University, in 2020, and the Ph.D. degree from the Department of Electrical Engineering, City University of Hong Kong, in 2025. He is currently a Post-Doctoral Researcher with the School of Computer Science and Technology, Shandong University. His research interests include deep learning, computer vision, and embodied intelligence.

Xinpeng Zhao received the B.E. degree from the School of Computer Science and Technology, Qingdao University, in 2022. He is currently pursuing the M.E. degree with the School of Computer Science and Technology, Shandong University. His research interests include computer vision and multimodal learning.

Xiao Zhang is currently an Associate Professor with the School of Computer Science and Technology, Shandong University. He has published more than 20 papers in the prestigious refereed journals and conference proceedings, such as IEEE TRANSACTIONS ON MOBILE COMPUTING, UBICOMP, ACM Multimedia, IJCAI, AAAI, ACM CIKM, and IEEE ICDM. His research interests include data mining, multi-task learning, and federated learning.

Jingke Meng received the Ph.D. degree in computer science and technology from Sun Yat-sen University, Guangzhou, China, in 2020. She is currently an Associate Professor with the School of Computer Science and Engineering, Sun Yat-sen University. Her research interests include computer vision, multi-modal learning, and visual navigation.

Mengbai Xiao received the Ph.D. degree from the Department of Computer Science, George Mason University, in 2018. He is currently a Qilu Young Professor with the School of Computer Science and Technology, Shandong University. Before this, he was a Post-Doctoral Researcher with the Department of Computer Science and Engineering, The Ohio State University, for two years.

Yifei Zou (Member, IEEE) received the B.E. degree from the School of Computer Science, Wuhan University, in 2016, and the Ph.D. degree from the Department of Computer Science, The University of Hong Kong, in 2020. He is currently an Associate Professor with the School of Computer Science and Technology, Shandong University. His research interests include wireless networks, ad hoc networks, and distributed computing.

Dongxiao Yu (Senior Member, IEEE) received the B.S. degree from the School of Mathematics, Shandong University, in 2006, and the Ph.D. degree from the Department of Computer Science, The University of Hong Kong, in 2014. He is currently a Professor at the School of Computer Science and Technology, Shandong University. His research interests include wireless networks, distributed computing, and graph algorithms.