GridMM: Grid Memory Map for Vision-and-Language Navigation

Zihan Wang[1,2], Xiangyang Li[1,2], Jiahao Yang[1,2], Yeqi Liu[1,2], Shuqiang Jiang[1,2]

[1]Key Lab of Intelligent Information Processing Laboratory of the Chinese Academy of Sciences (CAS),
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190, China
[2]University of Chinese Academy of Sciences, Beijing, 100049, China

zihan.wang@vipl.ict.ac.cn, lixiangyang@ict.ac.cn,
{jiahao.yang, yeqi.liu}@vipl.ict.ac.cn, sqjiang@ict.ac.cn

Abstract

Vision-and-language navigation (VLN) enables the agent to navigate to a remote location following the natural language instruction in 3D environments. To represent the previously visited environment, most approaches for VLN implement memory using recurrent states, topological maps, or top-down semantic maps. In contrast to these approaches, we build the top-down egocentric and dynamically growing Grid Memory Map (i.e., GridMM) to structure the visited environment. From a global perspective, historical observations are projected into a unified grid map in a top-down view, which can better represent the spatial relations of the environment. From a local perspective, we further propose an instruction relevance aggregation method to capture fine-grained visual clues in each grid region. Extensive experiments are conducted on both the REVERIE, R2R, SOON datasets in the discrete environments, and the R2R-CE dataset in the continuous environments, showing the superiority of our proposed method. The source code is available at https://github.com/MrZihan/GridMM.

1. Introduction

Vision-and-language navigation (VLN) tasks [4, 33, 40] require an agent to understand natural language instructions and act according to the instructions. Two distinct VLN scenarios have been proposed, being navigation in discrete environments (*e.g.*, R2R [4], REVERIE [40], SOON [62]) and in continuous environments (*e.g.*, R2R-CE [32], RxR-CE [33]). The discrete environment in VLN is abstracted as the topology structure of interconnected navigable nodes. With the connectivity graph, the agent can move to an adjacent node on the graph by selecting a direction from navigable directions. Different from the discrete environments, VLN in continuous environments require the agent to move through low-level controls (*i.e.*, turn left 15 degrees, turn

Instruction: *Walk straight ahead and turn right to cross in front of the refrigerator. Walk past the wood table and chairs on your right and turn right. Stop in between the blue couch and the wood table.*

(a) The topological map

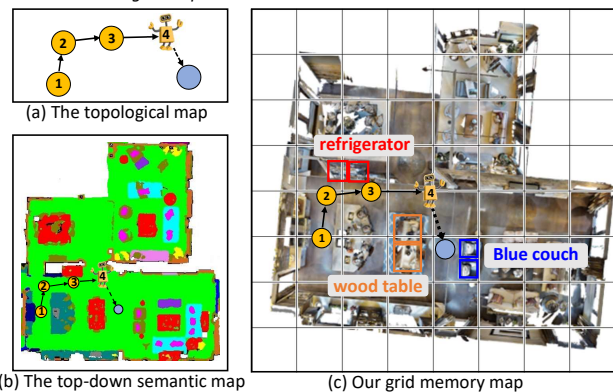(b) The top-down semantic map   (c) Our grid memory map

Figure 1. Illustration of different methods to represent the environment with maps for VLN.

right 15 degrees, or move forward 0.25 meters), which is closer to real-world robot navigation and more challenging.

Whether in discrete environments or continuous environments, historical information during navigation plays an important role in environment understanding and instruction grounding. In previous works [4, 18, 49, 55, 25], recurrent states are most commonly used as historical information for VLN, which encode historical observations and actions within a fixed-size state vector. However, such condensed states might be insufficient for capturing essential information in trajectory history. Therefore, Episodic transformer [39] and HAMT [12] propose to directly encode the trajectory history and actions as a sequence of previous observations instead of using recurrent states. Furthermore, in order to structure the visited environment and make global planning, a few recent approaches [10, 14, 34] structure the topological map, as shown in Fig. 1(a). However, these methods are difficult to represent the spatial relations among objects and scenes in historical observations, thus a lot of detailed information is lost. As shown in Fig. 1(b),

more recent works [27, 19, 11, 26] model the navigation environment using the top-down semantic map, which represents spatial relations more precisely. But the semantic concepts are extremely limited due to the pre-defined semantic labels. So the objects or scenes, which are not included in prior semantic labels, cannot be represented, such as the "refrigerator" in Fig. 1(b). Moreover, as illustrated in Fig. 1(b), the objects with diverse attributes such as "wood table" and "blue couch" cannot be fully expressed by the semantic map which misses object attributes.

In contrast to the above works [10, 14, 19, 26], we propose the Grid Memory Map (*i.e.*, GridMM), a visual representation structure for modeling global historical observations during navigation. Different from BEVBert [1], who applies local hybrid metric maps for short-term reasoning, our GridMM leverages both temporal and spatial information to depict the globally visited environment. Specifically, the grid map divides the visited environment into many equally large grid regions, and each grid region contains many fine-grained visual features. We dynamically construct a grid memory bank to update the grid map during navigation. At each step of navigation, the visual features from the pre-trained CLIP [43] model are saved into the memory bank, and all of them are categorized into the grid map regions based on their coordinates calculated via the depth information. To obtain the representation of each region, we design an instruction relevance aggregation method to capture the visual features most relevant to instructions and aggregate them into one holistic feature. With the help of $N \times N$ aggregated map features, the agent is able to accurately conduct the next action planning. A wealth of experiments illustrate the effectiveness of our GridMM compared with the previous methods.

In summary, we make the following contributions:

- We propose the Grid Memory Map for VLN to structure the global space-time relations of the visited environment and adopt instruction relevance aggregation to capture visual clues relevant to instructions.

- We comprehensively compare different maps representing the visited environment in VLN and analyze the characteristics of our proposed GridMM, which depicts more fine-grained information and gives some insights into future works in VLN.

- Extensive experiments are conducted to verify the effectiveness of our method in both discrete environments and continuous environments, which show that our method outperforms existing methods on many benchmark datasets.

2. Related work

Vision-and-Language Navigation (VLN). VLN [4, 55, 23, 53, 41, 14, 13] has received significant attention in recent years with the continual improvement. The VLN tasks include step-by-step instructions such as R2R [4] and RxR [33], navigation with dialog such as CVDN [52], and navigation for remote object grounding such as REVERIE [40] and SOON [62]. All tasks require the agent's ability to use time-dependent visual observations for decision-making. Restricted by the heavy computation of exploring the large action space in continuous environments, early works mainly focused on discrete environments. Among them, a recurrent unit is usually utilized to encode historical observations and actions within a fixed-size state vector [4, 18, 49, 55, 25]. Instead of relying on the recurrent states, HAMT [12] explicitly encodes the panoramic observation history to capture long-range dependency, and DUET [14] proposes to encode the topological map for efficient global planning. Inspired by the success of vision-and-language pre-training [48, 43], HOP [41, 42] utilizes well-designed proxy tasks for pre-training to enhance the interaction between vision and language modalities. ADAPT [38] employs action prompts to improve the cross-modal alignment ability. Based on data augmentation methods, some approaches enlarge training data of visual modality [28] and linguistic modality [18, 37, 17, 29] depending on existing VLN datasets. Moreover, AirBERT [20] and HM3D-AutoVLN [13] improve the performance by creating large-scale training dataset. KERM [36] utilizes a large knowledge base to depict navigation views for better generalization ability. In this work, we propose a dynamically growing grid memory map for structuring the visited environment and making long-term planning, which facilitates environment understanding and instruction grounding.

VLN in Continuous Environments (VLN-CE). VLN-CE [32] converts the topologically-defined VLN tasks such as R2R [4] into the continuous environment tasks, which is closer to real-world navigation. Different from the discrete environments, the agent in VLN-CE must navigate to the destination by selecting low-level action, similar to some visual navigation tasks [59, 58, 35, 60, 50, 51, 63]. Some approaches [19, 11] apply top-down semantic maps for environment understanding and use language-aligned waypoints supervision [27] for action prediction. Recently, Bridging [24] and Sim-2-Sim [31] for transferring pre-trained VLN agents to continuous environments have achieved considerable results. Compared with training agents from scratch in VLN-CE, this strategy can reduce the computational cost of pre-training and accelerate model convergence. In this work, we pre-train our model based on the proposed GridMM in discrete environments and then transfer the model to continuous environments. Experiments in both discrete environments and continuous environments illustrate the effectiveness of our method.

Maps for Navigation. The works on visual navigation [21, 8, 56] and other 3D indoor scene understand-
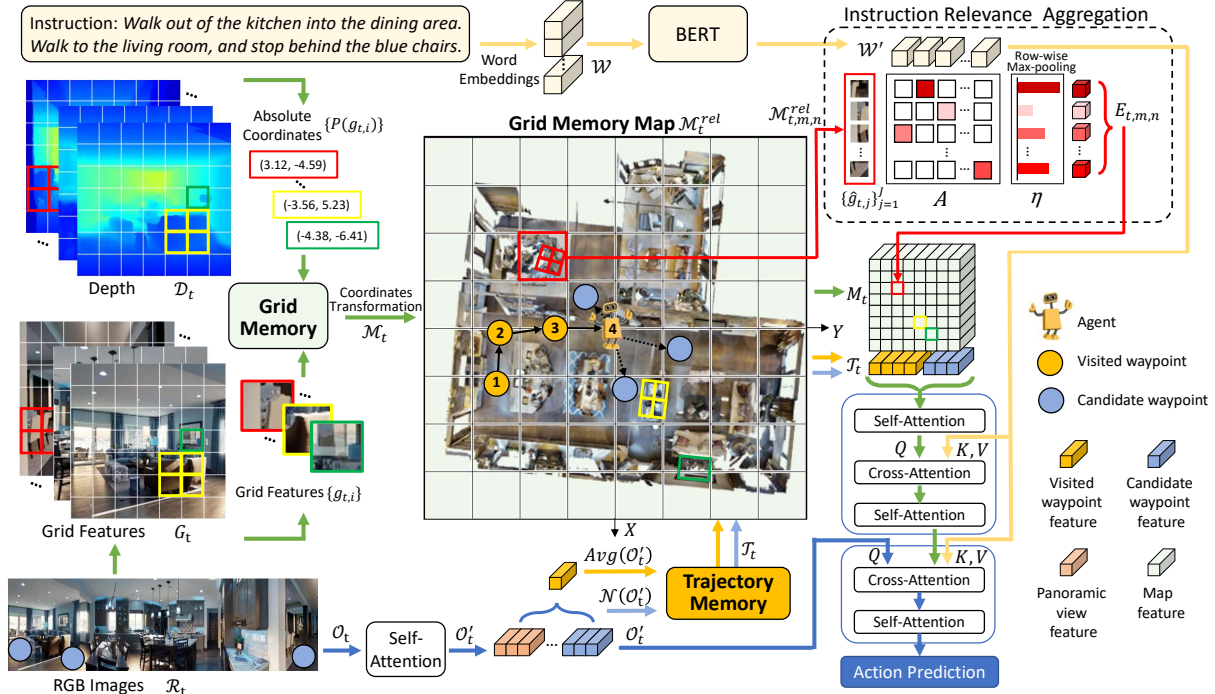
Figure 2. The overall pipeline. At step t, fine-grained grid features G_t are extracted from panoramic observations $\mathcal{R}_t$ and stored into grid memory $\mathcal{M}_t$ with their absolute coordinates (calculated via depth images $\mathcal{D}_t$ and coordinate of the agent). Waypoints of trajectory are represented via panoramic features $\mathcal{O}'_t$, and then stored into trajectory memory $\mathcal{T}_t$. An egocentric grid memory map can be constructed by projecting all features of $\mathcal{M}_t$ into a unified square map with $N \times N$ cells. Map features can be obtained by aggregating all features in each cell with an instruction relevance method. Panoramic view features, map features, and trajectory features are fed into a two-stage cross-modal encoder for action reasoning. For simplicity, layer normalization, feed-forward network, and residual structure are omitted from this figure. Best viewed in color.

ing tasks [22, 5, 6, 15] has a long tradition of constructing maps. Some works represent the map as topological structures for back-tracking to other locations [10] or supporting global action planning [14]. In addition, some approaches [19, 26] construct a top-down semantic map to more precisely represent spatial relations of the environment. Recently, BEVBert [1] introduced topo-metric maps from robotics into VLN, which uses topological maps for long-term planning and applies hybrid metric maps for short-term reasoning. Its metric map divides the local environment around the agent into 21×21 cells, and each cell represents a square region with a side length of $0.5m$. Moreover, the short-term visual observations within two steps are mapped into these cells. However, our GridMM is completely different in terms of: (**1**) BEVBert enriches the representations of the local observation with grid features. Our GridMM aims to perceive more space-time relationships with the dynamically growing grid map, which leverages both temporal and spatial information to depict the globally visited environment. (**2**) The grid-based metric map in BEVBert is only used for local action prediction. Our GridMM expands with the expansion of the visited environment, providing spatial enhanced representations for

both local and global action prediction. (**3**) The representations of the metric map for BEVBert are only the visual features. The representations of each cell in our GridMM are self-adapted to the instructions, which contain both visual and linguistic information.

3. Method

3.1. Navigation Setups

For VLN in discrete environments, the navigation connectivity graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ is provided in the Matterport3D simulator [7], where $\mathcal{V}$ denotes navigable nodes and $\mathcal{E}$ denotes edges. An agent is equipped with RGB and depth cameras, and a GPS sensor. Initialized at a starting node and given natural language instructions, the agent needs to explore the navigation connectivity graph $\mathcal{G}$ and reach the target node. $\mathcal{W} = \{w_l\}_{l=1}^L$ denote the word embeddings of the instruction with L words. At each time step t, the agent observes panoramic RGB images $\mathcal{R}_t = \{r_{t,k}\}_{k=1}^K$ and the depth images $\mathcal{D}_t = \{d_{t,k}\}_{k=1}^K$ of its current node $\mathcal{V}_t$, which contains K single view images. The agent is aware of a few navigable views $\mathcal{N}(\mathcal{R}_t) \in \mathcal{R}_t$ corresponding to its neighboring nodes and their coordinates.

VLN in continuous environments is established over Habitat [47], where the agent's position $\mathcal{P}_t$ can be any point in the open space. In each navigation step, we use a pre-trained waypoint predictor [24] to generate navigable waypoints in continuous environments, which assimilates the task with the VLN in discrete environments.

3.2. Grid Memory Mapping

As illustrated in Fig. 2, we present our grid memory mapping pipeline. At each navigation step t, we first store the fine-grained visual features and their corresponding coordinates in the grid memory. For the panoramic RGB images $\mathcal{R}_t = \{r_{t,k}\}_{k=1}^K$, we use a pre-trained CLIP-ViT-B/32 [43] model to extract grid features $G_t = \{g_{t,k} \in \mathbb{R}^{H \times W \times D}\}_{k=1}^K$, and the grid feature of row h column w is denoted as $g_{t,k,h,w} \in \mathbb{R}^D$. The corresponding depth images $\mathcal{D}_t$ are downsized to the same scale as $\mathcal{D}'_t = \{d'_{t,k} \in \mathbb{R}^{H \times W}\}_{k=1}^K$, and the depth value of row h column w is denoted as $d'_{t,k,h,w}$. For convenience, we denote all the subscripts (k, h, w) as i, where i ranges from 1 to I, and $I = K \cdot H \cdot W$. So $g_{t,k,h,w}$ is denoted as $\hat{g}_{t,i}$, and $d_{t,k,h,w}$ is denoted as $\hat{d}_{t,i}$. Similar to [3, 26], we can calculate the absolute coordinates $P(\hat{g}_{t,i})$ of $\hat{g}_{t,i}$:

$$
\begin{aligned}
P(\hat{g}_{t,i}) &= (x_{t,i} \, , \, y_{t,i}) \\
&= (\mathcal{X}_t + d_{t,i}^{line} \cdot cos\theta_{t,i} \, , \, \mathcal{Y}_t + d_{t,i}^{line} \cdot sin\theta_{t,i})
\end{aligned}
\tag{1}
$$

where $(\mathcal{X}_t, \mathcal{Y}_t)$ denotes the agent's current coordinate, $\theta_{t,i}$ denotes the heading angle between $\hat{g}_{t,i}$ and the current orientation of agent, $d_{t,i}^{line}$ denotes the euclidean distance between $\hat{g}_{t,i}$ and the agent, which can be calculated via $\hat{d}_{t,i}$ and $\theta_{t,i}$. We store all these grid features and their absolute coordinates in the grid memory:

$$
\mathcal{M}_t = \mathcal{M}_{t-1} \cup \{[\hat{g}_{t,i}, P(\hat{g}_{t,i})]\}_{i=1}^I
\tag{2}
$$

Then we propose a dynamic coordinate transformation method for constructing the grid memory map using visual features in grid memory $\mathcal{M}_t$. Intuitively, we can construct the maps as shown in Fig. 3(a). The visited environment is represented by projecting all historical observations $\hat{g}_{t,i}$ into unified maps based on their absolute coordinates $P(\hat{g}_{t,i})$. However, such maps have two drawbacks. First, it is not efficient enough to align the candidate observations and the instruction with the absolute coordinate. Second, it is difficult to determine the scale and extent of the map without prior information about the environment [61].

To address these deficiencies, we propose a new mapping method to construct the top-down egocentric and dynamically growing map, as illustrated in Fig. 3(b). At each step, we build a grid map in an egocentric view by projecting all features of the grid memory $\mathcal{M}_t$ into a new planar cartesian coordinate system with the agent's position as the coordinate origin and the agent's current direction as the positive direction of the y-axis. In this new coordinate system, for each grid feature $\hat{g}_{s,i}$ in $\mathcal{M}_t$ (where s ranges from 1 to t),
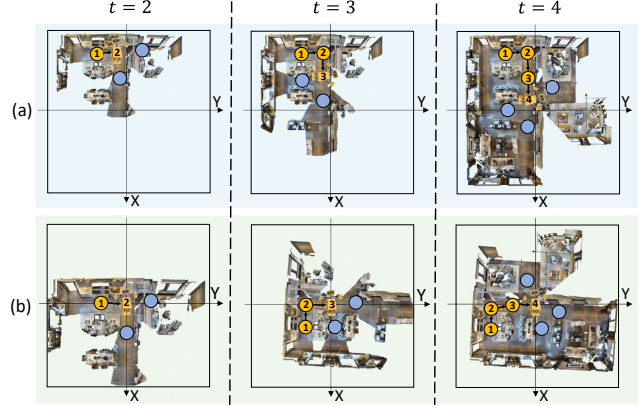

Figure 3. Maps in (a) use the absolute coordinate with a constant side length and coordinate origin. Maps in (b) use dynamically relative coordinates that the side length increases with the expansion of the visited environment, taking the position of the current agent as the coordinate origin and the direction of the current agent as the positive direction of the y-axis. Please zoom in for best view.

we can calculate the new relative coordinates $P_t^{rel}(\hat{g}_{s,i})$ in time step t:

$$
\begin{aligned}
P_t^{rel}(\hat{g}_{s,i}) &= (x_{s,i}^{rel} \, , \, y_{s,i}^{rel}) \\
&= (\, (x_{s,i} - \mathcal{X}_t) \cdot cos\Theta_t + (y_{s,i} - \mathcal{Y}_t) \cdot sin\Theta_t \, , \\
&\quad (y_{s,i} - \mathcal{Y}_t) \cdot cos\Theta_t - (x_{s,i} - \mathcal{X}_t) \cdot sin\Theta_t \,)
\end{aligned}
\tag{3}
$$

where Θ_t represents the heading angle between the new coordinate system and the old coordinate system.

Further, we construct the grid memory map (*i.e.*, GridMM) via the grid features and their new coordinates. At step t, the grid memory map takes L_t as the side length:

$$
\begin{aligned}
L_t = 2 \cdot max(\, &max(\{\{|x_{s,i}^{rel}|\}_{i=1}^I\}_{s=1}^t) \, , \\
&max(\{\{|y_{s,i}^{rel}|\}_{i=1}^I\}_{s=1}^t) \,)
\end{aligned}
\tag{4}
$$

So that the size of the GridMM increases with the expansion of the visited environment. The agent is always in the center of this map and the map is aligned with current panoramic observations in an egocentric view. Then the map is divided into $N \times N$ cells and all features of $\mathcal{M}_t$ are projected into these cells according to their new relative coordinates. Finally, we construct the grid memory map $\mathcal{M}_t^{rel}$ with $N \times N$ cells, and each cell contains multiple fine-grained visual features. After aggregating all visual features in each cell into one embedding vector, the map features $M_t \in \mathbb{R}^{N \times N \times D}$ is obtained. The detailed aggregation method is described in Sec. 3.3.2.

3.3. Model Architecture

3.3.1 Instruction and Observation Encoding

For instruction encoding, each word embedding in $\mathcal{W}$ is added with a position embedding and a token type embedding. All tokens are then fed into a multi-layer transformer

to obtain word representations, denoted as $\mathcal{W}' = \{w_l'\}_{l=1}^L$.

For view images $\mathcal{R}_t$ of the panoramic observation, we use the ViT-B/16 [16] pre-trained on ImageNet to extract visual features $\mathcal{R}_t'$. Then we represent their relative angles as $a_t = (sin\theta_t^a, cos\theta_t^a, sin\varphi_t^a, cos\varphi_t^a)$, where θ_t^a and φ_t^a are the relative heading and elevation angles to the agent's orientation. The candidate waypoints are represented as $\mathcal{N}(\mathcal{R}_t')$, and the line distance between waypoints and the current agent is denoted as b_t. Similarly, we represent the relative angles between the agent and the start waypoint as $c_t = (sin\theta_t^c, cos\theta_t^c, sin\varphi_t^c, cos\varphi_t^c)$. Then we concatenate the line distance $dist_{line}(\mathcal{V}_0, \mathcal{V}_t)$, navigation trajectory length $dist_{traj}(\mathcal{V}_0, \mathcal{V}_t)$, and action step $dist_{step}(\mathcal{V}_0, \mathcal{V}_t)$ between agent and the start waypoint to obtain $e_t = (dist_{line}(\mathcal{V}_0, \mathcal{V}_t), dist_{traj}(\mathcal{V}_0, \mathcal{V}_t), dist_{step}(\mathcal{V}_0, \mathcal{V}_t))$. Finally, the observation embeddings are as follows:

$$\mathcal{O}_t = LN(W_1^{\mathcal{O}}[\mathcal{R}_t'; \mathcal{N}(\mathcal{R}_t')]) + LN(W_2^{\mathcal{O}}[a_t; b_t; c_t; e_t]) \quad (5)$$

where the LN denotes layer normalization, $W_1^{\mathcal{O}}$ and $W_2^{\mathcal{O}}$ are learnable parameters. A special "stop" token $\mathcal{O}_{t,0}$ is added to $\mathcal{O}_t$ for the stop action. We use a two-layer transformer to model relations among observation embeddings and output $\mathcal{O}_t'$.

3.3.2 Grid Memory Encoding

As described in Sec. 3.2, we need to aggregate multiple grid features in each cell into one embedding vector. Due to the complexity of the navigation environment, a large number of grid features within each cell region are not all needed by the agent to complete navigation. The agent needs more critical and highly correlated information with current instruction to understand the environment. Therefore, we propose an instruction relevance method to aggregate features in each cell. Specifically, for grid features in each cell $\mathcal{M}_{t,m,n}^{rel} = \{\hat{g}_{t,j} \in \mathbb{R}^D\}_{j=1}^J$, where the corresponding co-ordinates $\{P^{rel}(\hat{g}_{t,j})\}_{j=1}^J$ are all within the cell of row m column n, the number of features in this cell is J. We evaluate the relevance of each grid feature to each token of navigation instruction by computing the relevance matrix A as:

$$A = (\mathcal{M}_{t,m,n}^{rel} W_1^A)(\mathcal{W}' W_2^A)^T \quad (6)$$

where W_1^A and W_2^A are learnable parameters. After that, we compute the row-wise max-pooling on A to evaluate the relevance of each grid feature to the instruction as:

$$\alpha_j = max(\{A_{j,l}\}_{l=1}^L) \quad (7)$$

At last, we aggregate the grid features within each cell into an embedding vector $E_{t,m,n}$:

$$\eta = softmax(\{\alpha_j\}_{j=1}^J) \quad (8)$$

$$E_{t,m,n} = \sum_{j=1}^J \eta_j(W^E \hat{g}_{t,j}) \quad (9)$$
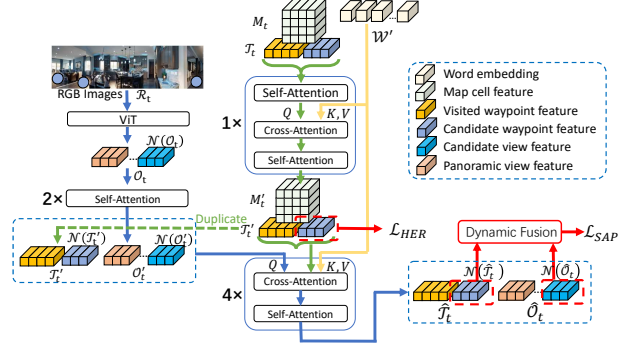

Figure 4. The detailed architecture for action prediction.

where W^E are learnable parameters. To represent the spatial relations, we introduce positional information into our grid memory map. Specifically, between each cell center and agent, we denote the line distance as q_t^M and represent relative heading angles as $h_t^M = (sin\Phi_t^M, cos\Phi_t^M)$. Then the map features can be obtained:

$$M_t = LN(E_t) + LN(W^M[q_t^M; h_t^M]) \quad (10)$$

where W^M are learnable parameters.

3.3.3 Navigation Trajectory Encoding

In order to implement global action planning, we further introduce the navigation trajectory into our GridMM. As shown in Sec. 3.3.1, at time step t, the agent receives panoramic features $\mathcal{O}_t'$ of waypoint $\mathcal{V}_t$. Then we can obtain visual representation $Avg(\mathcal{O}_t')$ of the current waypoint by average pooling of $\mathcal{O}_t'$. As the agent also partially observes candidate waypoints, we use the view image features $\mathcal{N}(\mathcal{O}_t')$ that contains these navigable waypoints as their visual representation. Between waypoints and current agent, we denote the line distances as $q^{\mathcal{T}}$, the relative heading angles as $h_t^{\mathcal{T}} = (sin\Phi_t^{\mathcal{T}}, cos\Phi_t^{\mathcal{T}})$, and the action step embeddings as $u^{\mathcal{T}}$. All historical waypoint features $\{Avg(\mathcal{O}_i')\}_{i=1}^{t-1}$, current waypoint feature $Avg(\mathcal{O}_t')$ and the candidate waypoint features $\mathcal{N}(\mathcal{O}_t')$ form the navigation trajectory:

$$\mathcal{T}_t = [\{LN(Avg(\mathcal{O}_i')) + LN(W_1^{\mathcal{T}}[q_i^{\mathcal{T}}; h_i^{\mathcal{T}}]) + u_i^{\mathcal{T}}\}_{i=1}^t;$$
$$LN(\mathcal{N}(\mathcal{O}_t')) + LN(W_2^{\mathcal{T}}[q_{\mathcal{N}}^{\mathcal{T}}; h_{\mathcal{N}}^{\mathcal{T}}]) + u_{\mathcal{N}}^{\mathcal{T}}] \quad (11)$$

where $W_1^{\mathcal{T}}$ and $W_2^{\mathcal{T}}$ are learnable parameters, a special "stop" token $\mathcal{T}_{t,0}$ is added to $\mathcal{T}_t$ for the stop action.

3.3.4 Cross-modal Reasoning

As illustrated in Fig. 2, we concatenate map features and navigation trajectory as $[M_t; \mathcal{T}_t]$, and then use a cross-modal transformer to fuse features from instruction $\mathcal{W}'$ and model space-time relations, forming the features $[M_t'; \mathcal{T}_t']$. We

specifically design a training loss $\mathcal{L}_{HER}$ (illustrated in Sec. 3.4) to supervise this module.

Subsequently, we use another cross-modal transformer with 4 layers to model vision-language relations and space-time relations. Specifically, each transformer layer consists of a cross-attention layer and a self-attention layer. For the cross-attention layer, we input panoramic observation and navigation trajectory $[\mathcal{O}'_t; \mathcal{T}'_t]$ as queries which attend over encoded instruction tokens, navigation trajectory and map features $[\mathcal{W}'; \mathcal{T}'_t; M'_t]$. And then, the self-attention layer takes encoded panoramic observation and navigation trajectory as input for action reasoning, where the output is denoted as $[\hat{\mathcal{O}}_t; \hat{\mathcal{T}}_t]$.

3.3.5 Action Prediction

We predict local navigation scores for the candidate views $\mathcal{N}(\hat{\mathcal{O}}_t)$ as below:

$$S_t^{\mathcal{O}} = FFN(\mathcal{N}(\hat{\mathcal{O}}_t)) \qquad (12)$$

and predict global navigation scores for the candidate navigable waypoints $\mathcal{N}(\hat{\mathcal{T}}_t)$ as below:

$$S_t^{\mathcal{T}} = FFN(\mathcal{N}(\hat{\mathcal{T}}_t)) \qquad (13)$$

where FFN denotes a two-layer feed-forward network. To be noted, $S_{t,0}^{\mathcal{O}}$ and $S_{t,0}^{\mathcal{T}}$ are the stop scores. Two separate FFNs are used to predict local action scores and global action scores, we gated fuse the scores following [14]:

$$S_t^{fusion} = \lambda_t S_t^{\mathcal{O}} + (1 - \lambda_t) S_t^{\mathcal{T}} \qquad (14)$$

where $\lambda_t = sigmoid(FFN([\hat{\mathcal{O}}_{t,0}; \hat{\mathcal{T}}_{t,0}]))$.

As illustrated in Fig. 2, $\mathcal{M}_t$ is the set of extracted features, and $\mathcal{M}_t^{rel}$ is the projected features with relative coordinates. $\mathcal{M}_{t,m,n}^{rel}$ is a subset of $\mathcal{M}_t^{rel}$ within a cell, and M_t is the obtained map features after aggregation. Meanwhile, the detailed architecture for action prediction is illustrated in Fig. 4. For loss functions, MLM and MVM are employed in the same ways as previous works [14] (which are omitted in Fig. 2 and Fig. 4). The SAP loss and HER loss are clearly described in Sec. 3.4. The candidate views are part of the agent's current panoramic observation as candidates for local action prediction. But the candidate waypoints are candidate locations in the global grid map for global action prediction. We use "Dynamic Fusion" to gated fuse these two action scores following DUET [14].

3.4. Pre-training and Fine-tuning

Pre-training. We utilize four tasks to pre-train our model.

1) Masked language modeling (MLM). We randomly mask out the words of the instruction with a probability of 15% and then predict the masked words $\mathcal{W}_{masked}$.

2) Masked view modeling (MVM). We randomly mask out view images with a probability of 15% and predict the semantic labels of masked view images. Similar to [14],

the target labels for view images are obtained by an image classification model [16] pre-trained on ImageNet.

3) Single-step action prediction (SAP). Given the ground truth action $\mathcal{A}_t$, the SAP loss is defined as follows:

$$\mathcal{L}_{SAP} = \sum_{t=1}^{T} CrossEntropy(S_t^{fusion}, \mathcal{A}_t) \qquad (15)$$

4) Historical environment reasoning (HER). The HER requires the agent to predict the next action only based on the map features and navigation trajectory, without panoramic observations:

$$S_t^{HER} = FFN(\mathcal{N}(\mathcal{T}'_t)) \qquad (16)$$

$$\mathcal{L}_{HER} = \sum_{t=1}^{T} CrossEntropy(S_t^{HER}, \mathcal{A}_t) \qquad (17)$$

Fine-tuning. For fine-tuning, we follow existing works [14, 24] to use Dagger [46] training techniques. Different from the pre-training process which uses the demonstration path, the supervision of fine-tuning is from a pseudo-interactive demonstrator which selects a navigable waypoint as the next target with the overall shortest distance from the current waypoint to the destination.

4. Experiment

4.1. Datasets and Evaluation Metrics

We evaluate our model on the REVERIE [40], R2R [4], SOON [62] datasets in discrete environments and R2R-CE [32] in continuous environments.

REVERIE contains high-level instructions which contain 21 words on average and the path length is between 4 and 7 steps. The predefined object bounding boxes are provided for each panorama, and the agent should select the correct object bounding box from candidates at the end of the navigation path.

R2R provides step-by-step instructions. The average length of instructions is 32 words and the average path length is 6 steps.

SOON also provides instructions that describe the target locations and target objects. The average length of instructions is 47 words, and the path length is between 2 and 21 steps. However, the object bounding boxes are not provided, the agent needs to predict the center location of the target object. Similar to the settings in [14], we use object detectors [2] to obtain candidate object boxes.

R2R-CE are collected based on the discrete Matterport3D environments [7], but use the Habitat simulator [44] to navigate in the continuous environments.

There are several standard metrics [4, 40] in VLN for evaluating the agent's performance, including Trajectory Length (TL), Navigation Error (NE), Success Rate (SR), SR

Methods	Val Unseen						Test Unseen					
	Navigation				Grounding		Navigation				Grounding	
	TL↓	OSR↑	SR↑	SPL↑	RGS↑	RGSPL↑	TL↓	OSR↑	SR↑	SPL↑	RGS↑	RGSPL↑
VLNBERT [25]	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 [20]	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 [41]	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 [12]	14.08	36.84	32.95	30.20	18.92	17.28	13.62	33.41	30.40	26.67	14.88	13.08
TD-STP [61]	-	39.48	34.88	27.32	21.16	16.56	-	40.26	35.89	27.51	19.88	15.40
DUET [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
BEVBert [1]	-	56.40	**51.78**	36.37	**34.71**	24.44	-	57.26	52.81	36.41	32.06	22.09
GridMM (Ours)	23.20	**57.48**	51.37	**36.47**	34.57	**24.56**	19.97	**59.55**	53.13	36.60	34.87	23.45

Table 1. Evaluation on the REVERIE dataset.

given the Oracle stop policy (OSR), Normalized inverse of the Path Length (SPL), Remote Grounding Success (RGS), and RGS penalized by Path Length (RGSPL).

4.2. Implementation Details

We adopt the pre-trained CLIP-ViT-B/32 [43] to extract grid features G_t on all datasets. We use the ViT-B/16 [16] pre-trained on ImageNet to extract panoramic view features $\mathcal{R}'_t$ on all datasets and extract object features on the REVERIE dataset as it provides bounding boxes. The BUTD object detector [2] is utilized on the SOON dataset to extract object bounding boxes. The number of layers for the language encoder, panorama encoder, map and trajectory encoder, and the cross-modal reasoning encoder are respectively set as 9, 2, 1, and 4 as shown in Fig. 2, all with a hidden size of 768. The parameters of all transformer layers are initialized with the pre-trained LXMERT [48].

4.3. Comparison to State-of-the-Art Methods

Table 1,2,3 compare our approach with the previous VLN methods on the REVERIE, R2R and SOON benchmarks. Table 4 compares our approach with the previous VLN-CE methods on the R2R-CE benchmark. Our approach achieves state-of-the-art performance on most metrics, demonstrating the effectiveness of the proposed approach. For the val unseen split of the REVERIE dataset in Table 1, our model outperforms the previous DUET [14] by 4.39% on SR and 2.74% on SPL. As shown in Table 2 and 3, it also shows performance gains on the R2R and SOON dataset compared to DUET. In particular, our approach significantly outperforms all previous methods on the R2R-CE dataset in Table 4, demonstrating the effectiveness of our GridMM for VLN-CE.

4.4. Ablation Study

We compare the performance of different maps representing the visited environments on the val unseen split of the R2R-CE dataset.

Methods	Val Unseen				Test Unseen			
	TL↓	NE↓	SR↑	SPL↑	TL↓	NE↓	SR↑	SPL↑
VLNBERT [25]	12.01	3.93	63	57	12.35	4.09	63	57
AirBERT [20]	11.78	4.01	62	56	12.41	4.13	62	57
SEvol [9]	12.26	3.99	62	57	13.40	4.13	62	57
HOP [41]	12.27	3.80	64	57	12.68	3.83	64	59
HAMT [12]	11.46	2.29	66	61	12.27	3.93	65	60
TD-STP [61]	-	3.22	70	63	-	3.73	67	61
DUET [14]	13.94	3.31	72	60	14.73	3.65	69	59
BEVBert [1]	14.55	**2.81**	75	64	15.87	**3.13**	73	**62**
GridMM (Ours)	13.27	2.83	**75**	**64**	14.43	3.35	**73**	**62**

Table 2. Evaluation on the R2R dataset.

Split	Method	TL↓	OSR↑	SR↑	SPL↑	RGSPL↑
Val Unseen	GBE [62]	28.96	28.54	19.52	13.34	1.16
	DUET [14]	36.20	50.91	36.28	22.58	3.75
	GridMM (Ours)	38.92	**53.39**	**37.46**	**24.81**	**3.91**
Test Unseen	GBE [62]	27.88	21.45	12.90	9.23	0.45
	DUET [14]	41.83	43.00	33.44	**21.42**	**4.17**
	GridMM (Ours)	46.20	**48.02**	**36.27**	21.25	4.15

Table 3. Evaluation on the SOON dataset.

1) Grid memory map vs. other maps. As shown in Table 5, we compare the effects of three different maps on the R2R-CE dataset. For row 2, we followed the same model structure as [14]. For row 3, we take the top-down semantic map as a substitute for grid features. Specifically, we followed CM² [19] to obtain an egocentric top-down semantic map, and use a convolution layer to extract semantic features in each cell instead of grid features. Row 4 uses a pre-trained object detection model VinVL [57] to detect multiple objects and extract their features as substitutes for grid features. More detailed experimental setups can be found in the supplementary materials.

In Table 5, all results with maps (rows 2-5) are better than the baseline method (row 1), which fully demonstrates the necessity of constructing maps representing the environments for VLN. Furthermore, our GridMM is better than DUET (topological map), as GridMM contains more fine-grained information. The method with a top-down semantic map (row 3) is beneficial to navigation, but it is still inferior to row 4, row 5, and even row 2 with the topological map.

Methods	Val Seen					Val Unseen					Test Unseen				
	TL↓	NE↓	OSR↑	SR↑	SPL↑	TL↓	NE↓	OSR↑	SR↑	SPL↑	TL↓	NE↓	OSR↑	SR↑	SPL↑
VLN-CE* [32]	9.26	7.12	46	37	35	8.64	7.37	40	32	30	8.85	7.91	36	28	25
AG-CMTP [10]	-	6.60	56.2	35.9	30.5	-	7.9	39.2	23.1	19.1	-	-	-	-	-
R2R-CMTP [10]	-	7.10	45.4	36.1	31.2	-	7.9	38.0	26.4	22.7	-	-	-	-	-
WPN [30]	8.54	5.48	53	46	43	7.62	6.31	40	36	34	8.02	6.65	37	32	30
LAW* [45]	9.34	6.35	49	40	37	8.89	6.83	44	35	31	-	-	-	-	-
CM²* [19]	12.05	6.10	50.7	42.9	34.8	11.54	7.02	41.5	34.3	27.6	13.9	7.7	39	31	24
CM²-GT* [19]	12.60	4.81	58.3	52.8	41.8	10.68	6.23	41.3	37.0	30.6	-	-	-	-	-
WS-MGridMM* [11]	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
Sim-2-Sim [31]	11.18	4.67	61	52	44	10.69	6.07	52	43	36	11.43	6.17	52	44	37
ERG† [54]	11.8	5.04	61	46	42	9.96	6.20	48	39	35	-	-	-	-	-
CMA† [24]	11.47	5.20	61	51	45	10.90	6.20	52	41	36	11.85	6.30	49	38	33
VLNBERT† [24]	12.50	5.02	59	50	44	12.23	5.74	53	44	39	13.31	5.89	51	42	36
DUET† (Ours) [14]	12.62	**4.13**	67	57	49	13.04	5.26	58	47	39	13.13	5.82	50	42	36
GridMM† (Ours)	12.69	4.21	**69**	**59**	**51**	13.36	**5.11**	**61**	**49**	**41**	13.31	**5.64**	**56**	**46**	**39**

Table 4. Evaluation on the R2R-CE dataset. The methods marked with * use a forward-facing camera with a 90-degree HFOV instead of panoramic images. The methods marked with † use the same waypoint predictor [24] for a fair comparison. Especially, we transfer the pre-trained DUET model to R2R-CE.

Mapping methods	TL↓	NE↓	OSR↑	SR↑	SPL↑
No Map	14.61	5.64	57.24	45.19	37.82
DUET (topological map)	13.04	5.26	57.91	47.02	38.86
Top-down semantic map	13.78	5.33	57.46	46.36	38.41
Map with object features	13.15	5.39	59.12	47.61	40.13
Our GridMM	13.36	**5.11**	**60.90**	**49.05**	**40.99**

Table 5. Comparison among different maps on the val unseen split of the R2R-CE dataset. Row 1 is our baseline method that uses our proposed model but without grid features. Row 3 takes top-down semantic maps as substitutes for grid features. Row 4 takes object features extracted from a detection model [57] as substitutes for grid features. Row 5 is our proposed GridMM.

GridMM	Ego.	Traj.	Instr.	TL↓	NE↓	OSR↑	SR↑	SPL↑
	✓	✓		14.61	5.64	57.24	45.19	37.82
✓		✓	✓	13.24	5.23	59.11	48.72	40.14
✓	✓		✓	13.14	5.24	58.35	47.42	39.41
✓	✓	✓		13.22	5.39	59.75	48.63	39.83
✓	✓	✓	✓	13.36	**5.11**	**60.90**	**49.05**	**40.99**

Table 6. Ablation study results on the val unseen split of R2R-CE dataset. "GridMM" denotes using grid memory map. "Ego." denotes using the egocentric view shown in Fig. 3(b) instead of the map in Fig. 3(a). "Traj." denotes using trajectory memory and features. "Instr." denotes using instruction relevance aggregation method instead of average pooling grid features in each cell.

Map scale	TL↓	NE↓	OSR↑	SR↑	SPL↑
8×8	13.42	5.23	58.58	47.07	39.49
14×14	13.36	5.11	**60.90**	49.05	40.99
20×20	12.59	**4.95**	57.86	**49.86**	**42.52**

Table 7. The effect of different map scales on the val unseen split of the R2R-CE dataset.

The reason is that map features extracted from the semantic map have a large gap with panoramic visual features.

Results in Table 5 indicate that GridMM is superior to the topological and semantic maps.

2) Grid features vs. object features. By comparing the results of row 4 and row 5 in Table 5, we can find out that the grid map using grid features works better than using object features. This is mainly because of the following reasons: (i) Object features from object detection model [57] are not enough to represent all visual information, such as house structure and background. (ii) Grid features from CLIP [43] have larger semantic space and better generalization ability. Different from previous methods [9] [54] of obtaining environment representation based on objects, grid features are of great importance for representing environments.

3) Is it necessary that map in an egocentric view? As shown in Sec. 3.2 and Fig. 3, we discussed two coordinate systems for our grid memory map, *i.e.*, absolute coordinates and dynamically relative coordinates. Row 2 in Table 6 shows the results of the absolute coordinate system, where the results are obtained by removing the process of coordinate transformation (*i.e.*, depicted in Equation 3) but the side length L_t of the map increases with the expansion of the visited environment (*i.e.*, depicted in Equation 4). For the settings in row 2, q_t^M and h_t^M (*i.e.*, depicted in Sec. 3.3.2) are replaced with the line distance and heading angle between each cell center and the start waypoint. The experimental results show that the egocentric relative coordinate system works better than the absolute coordinate system. It is mainly because maps with the absolute coordinate are not efficient enough to align the candidate observations and the instruction.

4) The effect of navigation trajectory information. As illustrated in Table 6, row 3 is inferior to row 5. The results verify the necessity of navigation trajectory, which helps with instruction grounding. The hypothesis is that the navigation trajectory can provide information for grounding the next step to *"cross in front of the refrigerator"* or to *"walk past the wood table and chairs on your right"*, as illustrated in Fig. 1 (c).

5) The effect of instruction relevance aggregation method. As shown in Table 6, row 5 with instruction relevance aggregation has better performance than row 4. Row 4 simply aggregates features in each map cell via average pooling, which makes it difficult to dig out critical visual cues. Our aggregation method evaluates the relevance of each grid feature to navigation instruction and uses the attention mechanism to filter out irrelevant features and capture critical clues.

6) The effect of map scale. As shown in Table 7, we evaluate the scale of our GridMM. We observe an upward trend in navigation performance as the map scale increases. This is mainly because a map with a larger scale can accommodate more environmental details and represent spatial relations more precisely. However, increasing the map scale leads to heavy computational cost but the gains are slight. So we choose a relatively balanced scale (*i.e.*, 14×14).
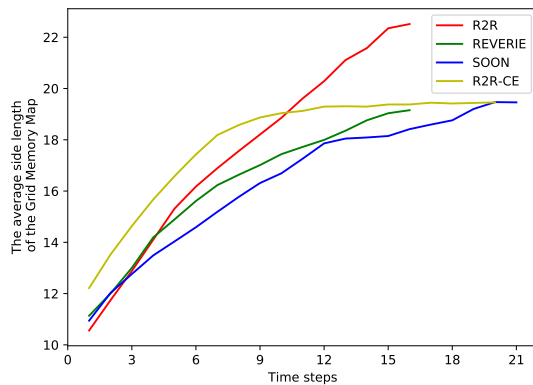
4.5. Statistical Analyses

Figure 5. The side length of the Grid Memory Map increases with the time steps. The x-axis indicates the time steps, and the y-axis is the average side length (meters) of the Grid Memory Map.

The side length of the GridMM. As illustrated in Fig. 5, the side length of GridMM increases with the expansion of the visited environment. For all datasets, the side length gradually increases from about 10 meters to about 20 meters during navigation. Obviously, the fixed-size map is difficult to adapt to the visited environment that constantly expands, thus our GridMM with a dynamically relative coordinate
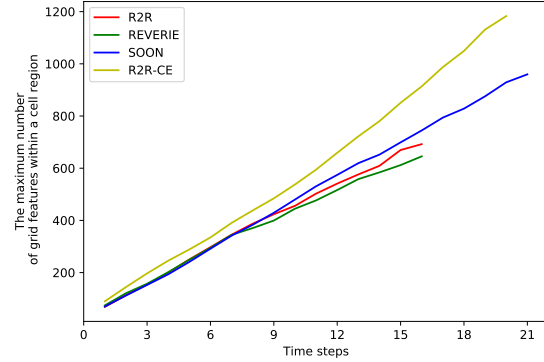

Figure 6. The maximum number of grid features within a cell region increases with the time steps. The x-axis indicates the navigation time steps, and the y-axis is the maximum number of grid features within a cell region.

system works better. Compared with other datasets, R2R has a larger map size at the end of navigation. It shows that the agent can explore new unvisited environments more on the R2R dataset.

The number of grid features within each cell region. As illustrated in Fig. 6, the maximum number of grid features within a cell region exceeds 600 at the end of navigation on all datasets. A large number of grid features within a cell region contain noise and are redundant. The average pooling of so many features is not efficient enough, resulting in critical cues being overwhelmed by noise. In contrast, the instruction relevance aggregation method works better than the average pooling, which filters out irrelevant features and captures critical clues.

5. Conclusion

In this paper, we propose a top-down egocentric and dynamically growing Grid Memory Map (*i.e.*, GridMM) to structure the visited environment for VLN. Moreover, an instruction relevance aggregation module is proposed to capture fine-grained visual clues relevant to instructions. We comprehensively analyze the effectiveness of our model and compare it with other methods. Our GridMM provides both global space-time perception and local detailed clues, thus enabling more accurate navigation results. However, there are still some limitations to our approach, regarding how to handle multi-floor environments remains open. In the future, we will continuously explore how to better represent the indoor environment for VLN and Embodied AI.

References

[1] Dong An, Yuankai Qi, Yangguang Li, Yan Huang, Liang Wang, Tieniu Tan, and Jing Shao. Bevbert: Topo-metric map pre-training for language-guided navigation. *arXiv preprint arXiv:2212.04385*, 2022.

[2] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *CVPR*, pages 6077–6086, 2018.

[3] Peter Anderson, Ayush Shrivastava, Joanne Truong, Arjun Majumdar, Devi Parikh, Dhruv Batra, and Stefan Lee. Sim-to-real transfer for vision-and-language navigation. In *Conference on Robot Learning (CoRL)*, 2020.

[4] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *CVPR*, pages 3674–3683, 2018.

[5] Edward Beeching, Jilles Dibangoye, Olivier Simonin, and Christian Wolf. Egomap: Projective mapping and structured egocentric memory for deep rl. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 525–540. Springer, 2020.

[6] Vincent Cartillier, Zhile Ren, Neha Jain, Stefan Lee, Irfan Essa, and Dhruv Batra. Semantic mapnet: Building allocentric semantic maps and representations from egocentric views. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 964–972, 2021.

[7] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. In *3DV*, pages 667–676, 2017.

[8] Devendra Singh Chaplot, Dhiraj Prakashchand Gandhi, Abhinav Gupta, and Russ R Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. *Advances in Neural Information Processing Systems*, 33:4247–4258, 2020.

[9] Jinyu Chen, Chen Gao, Erli Meng, Qiong Zhang, and Si Liu. Reinforced structured state-evolution for vision-language navigation. In *CVPR*, pages 15450–15459, June 2022.

[10] Kevin Chen, Junshen K. Chen, Jo Chuang, Marynel Vázquez, and Silvio Savarese. Topological planning with transformers for vision-and-language navigation. In *CVPR*, 2021.

[11] Peihao Chen, Dongyu Ji, Kunyang Lin, Runhao Zeng, Thomas H Li, Mingkui Tan, and Chuang Gan. Weakly-supervised multi-granularity map learning for vision-and-language navigation. In *NeurIPS*, 2022.

[12] Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. History aware multimodal transformer for vision-and-language navigation. In *NeurIPS*, volume 34, pages 5834–5847, 2021.

[13] Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. Learning from unlabeled 3d environments for vision-and-language navigation. In *ECCV*, pages 638–655, 2022.

[14] Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. Think global, act local: Dual-scale graph transformer for vision-and-language navigation. In *CVPR*, pages 16537–16547, 2022.

[15] Samyak Datta, Sameer Dharur, Vincent Cartillier, Ruta Desai, Mukul Khanna, Dhruv Batra, and Devi Parikh. Episodic memory question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19119–19128, 2022.

[16] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2020.

[17] Zi-Yi Dou and Nanyun Peng. Foam: A follower-aware speaker model for vision-and-language navigation. In *NAACL*, 2022.

[18] Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. Speaker-follower models for vision-and-language navigation. In *NeurIPS*, volume 31, 2018.

[19] Georgios Georgakis, Karl Schmeckpeper, Karan Wanchoo, Soham Dan, Eleni Miltsakaki, Dan Roth, and Kostas Daniilidis. Cross-modal map learning for vision and language navigation. In *CVPR*, 2022.

[20] Pierre-Louis Guhur, Makarand Tapaswi, Shizhe Chen, Ivan Laptev, and Cordelia Schmid. Airbert: In-domain pretraining for vision-and-language navigation. In *CVPR*, pages 1634–1643, 2021.

[21] Saurabh Gupta, James Davidson, Sergey Levine, Rahul Sukthankar, and Jitendra Malik. Cognitive mapping and planning for visual navigation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2616–2625, 2017.

[22] Joao F Henriques and Andrea Vedaldi. Mapnet: An allocentric spatial memory for mapping environments. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8476–8484, 2018.

[23] Yicong Hong, Cristian Rodriguez, Yuankai Qi, Qi Wu, and Stephen Gould. Language and visual entity relationship graph for agent navigation. In *NeurIPS*, volume 33, pages 7685–7696, 2020.

[24] Yicong Hong, Zun Wang, Qi Wu, and Stephen Gould. Bridging the gap between learning in discrete and continuous environments for vision-and-language navigation. In *CVPR*, June 2022.

[25] Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. Vln bert: A recurrent vision-and-language bert for navigation. In *CVPR*, pages 1643–1653, 2021.

[26] Chenguang Huang, Oier Mees, Andy Zeng, and Wolfram Burgard. Visual language maps for robot navigation. In *ICRA*, London, UK, 2023.

[27] Muhammad Zubair Irshad, Niluthpol Chowdhury Mithun, Zachary Seymour, Han-Pang Chiu, Supun Samarasekera, and Rakesh Kumar. Sasra: Semantically-aware spatio-temporal reasoning agent for vision-and-language navigation in continuous environments. *arXiv preprint arXiv:2108.11945*, 2021.

[28] Mohit Bansal Jialu Li, Hao Tan. Envedit: Environment editing for vision-and-language navigation. In *CVPR*, 2022.

[29] Aishwarya Kamath, Peter Anderson, Su Wang, Jing Yu Koh, Alexander Ku, Austin Waters, Yinfei Yang, Jason Baldridge, and Zarana Parekh. A new path: Scaling vision-and-language navigation with synthetic instructions and imitation learning. *arXiv preprint arXiv:2210.03112*, 2022.

[30] Jacob Krantz, Aaron Gokaslan, Dhruv Batra, Stefan Lee, and Oleksandr Maksymets. Waypoint models for instruction guided navigation in continuous environment. In *ICCV*, 2021.

[31] Jacob Krantz and Stefan Lee. Sim-2-sim transfer for vision-and-language navigation in continuous environments. In *ECCV*, 2022.

[32] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *ECCV*, 2020.

[33] Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *EMNLP*, pages 4392–4412, 2020.

[34] Mingxiao Li, Zehao Wang, Tinne Tuytelaars, and Marie-Francine Moens. Layout-aware dreamer for embodied referring expression grounding. In *AAAI*, 2023.

[35] Weijie Li, Xinhang Song, Yubing Bai, Sixian Zhang, and Shuqiang Jiang. ION: instance-level object navigation. In *ACM MM*, pages 4343–4352, 2021.

[36] Xiangyang Li, Zihan Wang, Jiahao Yang, Yaowei Wang, and Shuqiang Jiang. KERM: Knowledge enhanced reasoning for vision-and-language navigation. In *CVPR*, pages 2583–2592, 2023.

[37] Xiwen Liang, Fengda Zhu, Lingling Li, Hang Xu, and Xiaodan Liang. Visual-language navigation pretraining via prompt-based environmental self-exploration. In *ACL*, pages 4837–4851, 2022.

[38] Bingqian Lin, Yi Zhu, Zicong Chen, Xiwen Liang, Jianzhuang Liu, and Xiaodan Liang. Adapt: Vision-language navigation with modality-aligned action prompts. In *CVPR*, pages 15396–15406, 2022.

[39] Alexander Pashevich, Cordelia Schmid, and Chen Sun. Episodic transformer for vision-and-language navigation. In *ICCV*, 2021.

[40] Yuankai Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. Reverie: Remote embodied visual referring expression in real indoor environments. In *CVPR*, pages 9982–9991, 2020.

[41] Yanyuan Qiao, Yuankai Qi, Yicong Hong, Zheng Yu, Peng Wang, and Qi Wu. HOP: History-and-order aware pre-training for vision-and-language navigation. In *CVPR*, pages 15418–15427, 2022.

[42] Yanyuan Qiao, Yuankai Qi, Yicong Hong, Zheng Yu, Peng Wang, and Qi Wu. Hop+: History-enhanced and order-aware pre-training for vision-and-language navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[43] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, pages 8748–8763, 2021.

[44] Santhosh K Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alex Clegg, John Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, et al. Habitat-matterport 3d dataset (hm3d): 1000 large-scale 3d environments for embodied ai. *arXiv preprint arXiv:2109.08238*, 2021.

[45] Sonia Raychaudhuri, Saim Wani, Shivansh Patel, Unnat Jain, and Angel Chang. Language-aligned waypoint (law) supervision for vision-and-language navigation in continuous environments. In *EMNLP*, 2021.

[46] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

[47] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *ICCV*, pages 9339–9347, 2019.

[48] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. In *EMNLP*, pages 5103–5114, 2019.

[49] Hao Tan, Licheng Yu, and Mohit Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. In *NAACL*, pages 2610–2621, 2019.

[50] Tianqi Tang, Heming Du, Xin Yu, and Yi Yang. Monocular camera-based point-goal navigation by learning depth channel and cross-modality pyramid fusion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 5422–5430, 2022.

[51] Tianqi Tang, Xin Yu, Xuanyi Dong, and Yi Yang. Auto-navigator: Decoupled neural architecture search for visual navigation. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 3743–3752, 2021.

[52] Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. Vision-and-dialog navigation. In *PMLR*, 2020.

[53] Hanqing Wang, Wenguan Wang, Wei Liang, Caiming Xiong, and Jianbing Shen. Structured scene memory for vision-language navigation. In *CVPR*, pages 8455–8464, 2021.

[54] Ting Wang, Zongkai Wu, Feiyu Yao, and Donglin Wang. Graph based environment representation for vision-and-language navigation in continuous environments. *arXiv preprint arXiv:2301.04352*, 2023.

[55] Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *CVPR*, pages 6629–6638, 2019.

[56] Saim Wani, Shivansh Patel, Unnat Jain, Angel Chang, and Manolis Savva. Multion: Benchmarking semantic map memory using multi-object navigation. *Advances in Neural Information Processing Systems*, 33:9700–9712, 2020.

[57] Pengchuan Zhang, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. Vinvl: Revisiting visual representations in vision-language models. In *CVPR*, pages 5579–5588, 2021.

[58] Sixian Zhang, Weijie Li, Xinhang Song, Yubing Bai, and Shuqiang Jiang. Generative meta-adversarial network for unseen object navigation. In *ECCV*, volume 13699, pages 301–320.

[59] Sixian Zhang, Xinhang Song, Yubing Bai, Weijie Li, Yakui Chu, and Shuqiang Jiang. Hierarchical object-to-zone graph for object navigation. In *ICCV*, pages 15110–15120, 2021.

[60] Sixian Zhang, Xinhang Song, Weijie Li, Yubing Bai, Xinyao Yu, and Shuqiang Jiang. Layout-based causal inference for object navigation. In *CVPR*, pages 10792–10802, 2023.

[61] Yusheng Zhao, Jinyu Chen, Chen Gao, Wenguan Wang, Lirong Yang, Haibing Ren, Huaxia Xia, and Si Liu. Target-driven structured transformer planner for vision-language navigation. In *ACM MM*, pages 4194–4203, 2022.

[62] Fengda Zhu, Xiwen Liang, Yi Zhu, Qizhi Yu, Xiaojun Chang, and Xiaodan Liang. Soon: Scenario oriented object navigation with graph-based exploration. In *CVPR*, pages 12689–12699, 2021.

[63] Fengda Zhu, Linchao Zhu, and Yi Yang. Sim-real joint reinforcement transfer for 3d indoor navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11388–11397, 2019.