Adaptive Zone-aware Hierarchical Planner for Vision-Language Navigation

Chen Gao[1], Xingyu Peng[1], Mi Yan[3], He Wang[3], Lirong Yang[4]
Haibing Ren[4], Hongsheng Li[5], Si Liu[1,2]*
[1]Institute of Artificial Intelligence, Beihang University [2]Hangzhou Innovation Institute, Beihang University
[3]CFCS and School of EECS, Peking University [4]Meituan [5]The Chinese University of Hong Kong

Abstract

The task of Vision-Language Navigation (VLN) is for an embodied agent to reach the global goal according to the instruction. Essentially, during navigation, a series of sub-goals need to be adaptively set and achieved, which is naturally a hierarchical navigation process. However, previous methods leverage a single-step planning scheme, i.e., directly performing navigation action at each step, which is unsuitable for such a hierarchical navigation process. In this paper, we propose an Adaptive Zone-aware Hierarchical Planner (AZHP) to explicitly divides the navigation process into two heterogeneous phases, i.e., sub-goal setting via zone partition/selection (high-level action) and sub-goal executing (low-level action), for hierarchical planning. Specifically, AZHP asynchronously performs two levels of action via the designed State-Switcher Module (SSM). For high-level action, we devise a Scene-aware adaptive Zone Partition (SZP) method to adaptively divide the whole navigation area into different zones on-the-fly. Then the Goal-oriented Zone Selection (GZS) method is proposed to select a proper zone for the current sub-goal. For low-level action, the agent conducts navigation-decision multi-steps in the selected zone. Moreover, we design a Hierarchical RL (HRL) strategy and auxiliary losses with curriculum learning to train the AZHP framework, which provides effective supervision signals for each stage. Extensive experiments demonstrate the superiority of our proposed method, which achieves state-of-the-art performance on three VLN benchmarks (REVERIE, SOON, R2R).

1. Introduction

In recent years, Embodied-AI (E-AI) research has attracted a surge of interest within the computer vision, natural language processing and robotics communities since its interdisciplinary nature. The long-term goal of E-AI research is to build intelligent agents that can interact with humans to complete assigned tasks. In this paper, we focus on the Vision-Language Navigation (VLN) task, one of the most fundamental E-AI topics, where embodied agents need to navigate in a photorealistic 3D environment (generally unseen) according to the natural language instruction.

The instruction given to the agent is mainly two types, *i.e.*, step-by-step instruction (*e.g.* R2R) and goal-oriented instruction (*e.g.* REVERIE and SOON). The latter is more practical for the home assistant robot since people usually do not provide fine-grained commands, but also more challenging. Firstly, the goal-oriented instruction contains po-

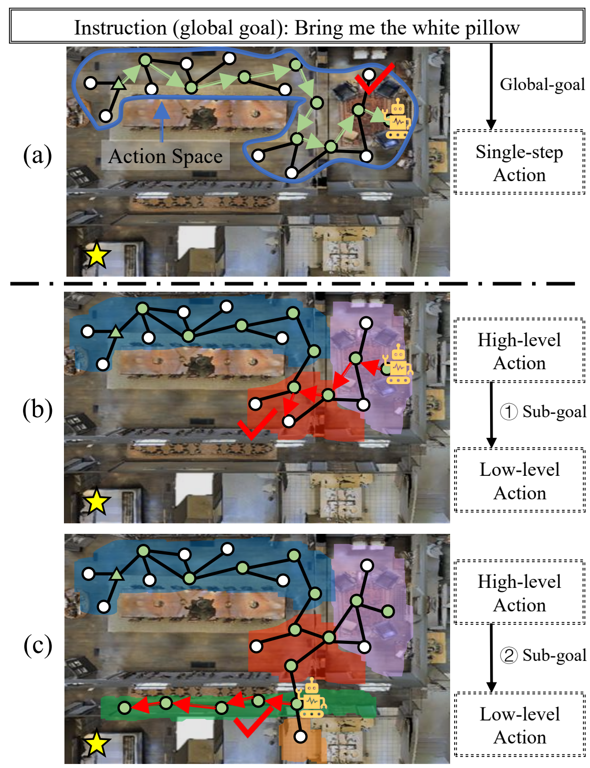

Figure 1. Given a goal-oriented/semantic-level instruction, (a) previous methods essentially adopt a singel-step navigation paradigm, *i.e.*, directly taking an action from the action space according to the global goal at each step; (b)(c) we instead propose a hierarchical navigation paradigm, containing high- and low-level actions to adaptively set and achieve a series of sub-goals.

*Corresponding author

tential *hierarchical information*. As shown in Figure 1, the global goal is "bring me the white pillow", which indicates the agent needs to complete several potential sub-goals, *e.g*., leaving the current room, finding the bedroom, locating the white pillow. Thus it is essentially a hierarchical navigation process, where the high-level process is *sub-goal setting* and the low-level process is *sub-goal executing*. Secondly, sub-goal means reaching a sub-target in a sub-region, which requires the agent to divide the scene into several zones and choose the proper zone for the current sub-goal. Importantly, the sub-goal depends not only on the instruction, and an appropriate sub-goal needs to be set based on the agent's current state, which means the agent needs to conduct zone partition and selection adaptively during navigation. For example, in Figure 1(b), when the agent is in the living room, the sub-goal is set as "finding the exit in the exit area (red zone)". Thirdly, it is non-trivial to learn such a hierarchical navigation policy, especially given that there are no expert demonstrations for teaching the high-level process.

However, the dominant paradigm of current state-of-the-art VLN methods is essentially a *singel-step planning* paradigm as shown in Figure 1(a). It directly takes one step of navigation action at each time, according to the action space and the global goal. Such a paradigm does not explicitly model the hierarchical planning nature of the VLN task, largely limiting the long-horizon decision ability.

To address the issues, we propose an Adaptive Zone-aware Hierarchical Planner (AZHP) based on our main idea, *i.e*., building a novel hierarchical planning framework for the VLN task. Firstly, AZHP models the navigation process as a hierarchical action-making process containing high-level and low-level actions. During navigation, the high-level action aims to set sub-goals, and the low-level action aims to complete the sub-goals accordingly. Specifically, the high-level action divides the whole scene into different zones and selects a proper zone for navigation based on the current state, *e.g*., the green zone (hallway) in Figure 1(c). Then the low-level action is applied to execute specific navigation decision multi-steps in the selected zone until reaching the sub-target. Secondly, for the high-level action, we propose a Scene-aware adaptive Zone Partition (SZP) method to adaptively divide the global action map into several zones on-the-fly, according to the position and observations of each viewpoint. Note that the action map is a maintained topological map that records the historical trajectory and observations. Also, we design a Goal-oriented Zone Selection (GZS) method to select a specific zone according to the instruction and zone attributes. Besides, a State-Switcher Module (SSM) is placed to decide whether the current sub-goal is achieved and switch to the next sub-goal, supporting the asynchronous scheme. Thirdly, since there is no direct supervision signal for high-level action training, we propose a Hierarchical Reinforcement Learning (HRL) strategy to provide cooperative rewards. Besides, we design auxiliary losses with a curriculum learning strategy to improve the learning robustness further.

In summary, we make the following contributions. (i) We propose AZHP, which conducts a hierarchical navigation paradigm via setting two-level actions, to solve the long-horizon planning VLN task. To the best of our knowledge, AZHP is the pioneering work investigating hierarchical planning strategy for the VLN task. (ii) We devise SZP and GZS for high-level action, where SZP adaptively divides scenes into several zones on-the-fly, and GZS selects the corresponding zone for a specific sub-goal. Also, SSM is designed to support asynchronous switching between high/low-level actions. (iii) To construct and learn the hierarchical planning policy, we design an HRL strategy and auxiliary losses with a curriculum learning manner. Superior performance on three datasets demonstrates the method's effectiveness. Code is available at: `https://github.com/chengaopro/AZHP`.

2. Related Work

2.1. Vision-Language Navigation

With the development of standard datasets [3, 5, 11, 22, 28, 43, 61], the Vision-Language Navigation (VLN) task [3, 16, 54] is getting increasingly popular in recent years.

Early VLN solutions utilise RNN to reserve the navigation history [3, 14, 31, 36, 47, 49, 53]. However, these methods have limited ability to capture long-range dependency as the path length grows. To fulfil such a long-term memory in navigation, transformer-based models [8, 17, 18, 20, 30, 37, 41, 57, 59] are devised for the VLN task. VLN-BERT [20] inserts a recurrent state in the transformer to record navigation history. In HAMT [8], all observations and actions in history are directly encoded by transformers as different types of tokens. Meanwhile, some works [2, 7, 10, 13, 19, 51, 61] aim to build navigation topological maps on-the-fly to provide more complete scenario representations. DUET [10] dynamically assembles the topological map during navigation and combines the local-global decision results. [51] conducts reasoning and designed policy on the global action map. Another line of works [2, 9, 10, 24–26, 36, 51, 52, 55, 58] focus on the navigation policy learning. In EGP [13], the agent can jump to a remote viewpoint along the shortest path recorded in topological maps. Pathdreamer [25] designs a model to generate features for future viewpoints and reduces exploring costs.

However, none of the work focuses on investigating the hierarchical nature of VLN task. They essentially lie in a single-step navigation paradigm that performs one-level actions step-by-step to achieve the global goal.

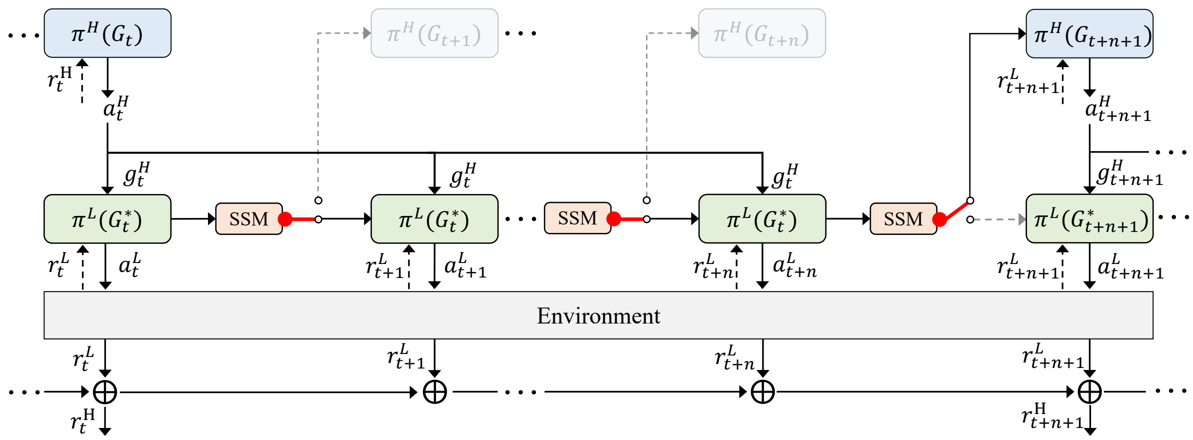

Figure 2. The overall architecture of the designed hierarchical planning scheme, which contains a high-level policy $\pi^H(\cdot)$ and low-level policy $\pi^L(\cdot)$. Specifically, $\pi^H(\cdot)$ sets a sub-goal g_t^H via taking action a_t^H, *i.e.*, zone partition/selection. Further, $\pi^L(\cdot)$ conducts multistep navigation actions a_t^L within the zone. Besides, SSM aims to decide when to switch the state for the next sub-goal.

2.2. Hierarchical Reinforcement Learning

Hierarchical Reinforcement Learning (HRL) is a popular framework [45] for long-horizon tasks due to its efficient exploration [38, 39] and interpretability. In this framework, the action is decomposed into two hierarchical levels, and the agent outputs the high/low-level actions sequentially. Currently, there are two common kinds of HRL, *i.e.*, the options framework [4, 23, 48] and the goal-conditional framework [29, 38, 40]. The former chooses an option from a fixed-size option set as the high-level action and selects the corresponding low-level policy, while the latter outputs a continuous embedding as the high-level action which serves as the input to the low-level policy. Our work falls into the option framework while differs from them by adaptively adding options during navigation to better handle the growing action space. Recently, HRL is also explored in navigation tasks [21, 27, 29] with continuous space. Compared to them, we further demonstrate that the high-level policy, which explicitly divides action space into zones and chooses zone-level sub-goals, can better improve performance.

3. Method

Problem Formulation. In the VLN task, an embodied agent is required to navigate to the target location [3, 22] or even localise the remote target object [43, 61], with the hint of a natural language instruction and observed environment along the path. In the beginning, the agent is spawned at a random location in a previously unseen environment, where the navigable area is described as an undirected graph G with an adjacency matrix E. Note that the agent can only observe the explored part of G. At each location (viewpoint), the agent receives a panoramic view of the surrounding environment, which is represented as a set of local views $\{o_i \in \mathbb{R}^{1\times D_h} | i = 1, \cdots, 36\}$. Each local view contains the corresponding heading $angle_h$ and elevation $angle_e$.

3.1. Hierarchical Planning Overview

The VLN task is intrinsically hierarchical, which consists of a high-level process (*i.e.* sub-goal setting) and a low-level process (*i.e.* sub-goal executing). Note that sub-goal means reaching a sub-target in a sub-region. Therefore, we propose an Adaptive Zone-aware Hierarchical Planner (AZHP) to model such a hierarchical planning process explicitly. Specifically, AZHP is composed of two policy networks as shown in Figure 2, where a high-level policy $\pi^H(\cdot)$ learns to set sub-goal g^H and a low-level policy $\pi^L(\cdot)$ learns to achieve g^H accordingly. During navigation, at time step t, we maintain a topological graph G_t following [10] to record the historical trajectory and current observations.

Specifically, at time step t, the high-level action a_t^H, *i.e.*, zone partition/selection, is taken based on the learned high-level policy network $a_t^H \leftarrow \pi^H(a_t^H | G_t; \boldsymbol{\theta}^H)$, where $\boldsymbol{\theta}^H$ is the network parameter. After zone partition and selection, we obtain a selected zone G_t^*, which is a sub-graph of G_t. Then the low-level action a_t^L, *i.e.*, navigation decision, is conducted in multi-steps. a_t^L is obtained via the low-level policy: $a_t^L \leftarrow \pi^L(a_t^L | G_t^*; \boldsymbol{\theta}^L)$, which means the agent just navigates in G_t^*. In addition, we propose a State-Switcher Module (SSM) to learn whether g^H is achieved and whether switching to the next sub-goal during navigation.

3.2. Hierarchical Reinforcement Learning

Since there is no expert demonstration for training the high-level policy $\pi^H(\cdot)$, we propose a hierarchical reinforcement learning (HRL) solution (shown in Figure 2) and auxiliary losses (introduced in Sec 3.5).

Firstly, we design a reward function for the low-level policy $\pi^L(\cdot)$. During navigation, at step t, the agent executes an action a_t^L. Then the reward r_t^L is defined as: (i) when the distance dis_t to the target location changes, $r_t^L = -(dis_t - dis_{t-1})$; (ii) when the agent stops at the

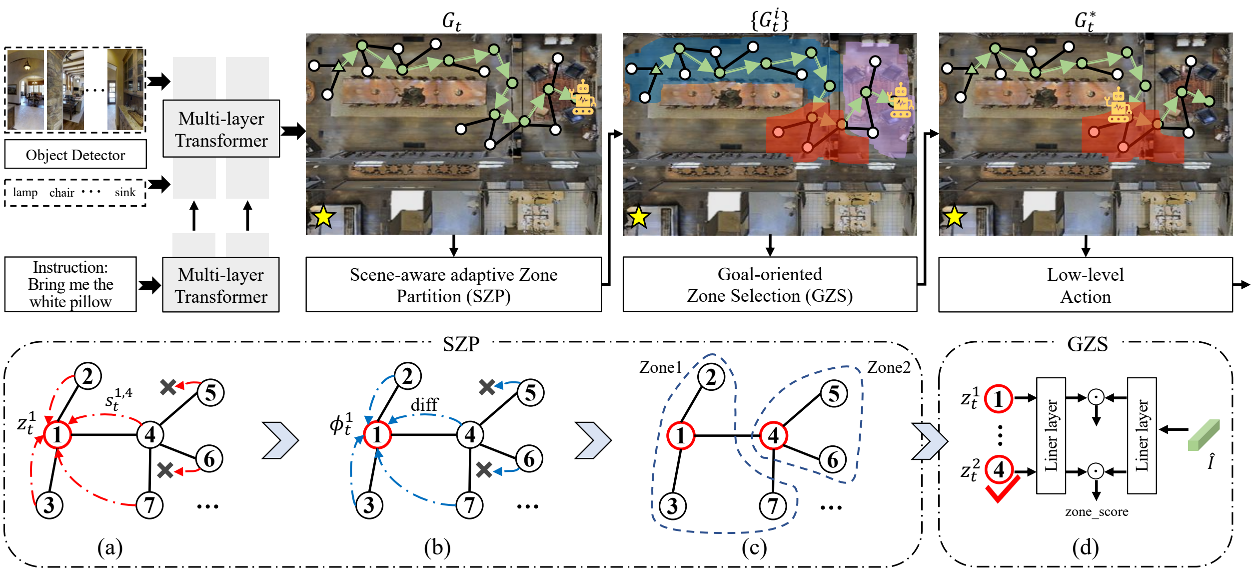

Figure 3. Illustration of the pipeline of AZHP (above) and SZP/GZS (below). Best viewed in colour.

target location, r_t^L is set to 10, otherwise -10; (iii) passing the target location without stopping gets a -10 reward. Then the reward for high-level policy is obtained via accumulation $r_t^H = \sum_{t=start}^{end} r_t^L$, where $[start, end]$ is the time interval of the current sub-goal.

Then we adopt the Temporal Difference (TD) algorithm to optimise both policy networks based on r_t^H and r_t^L. For simplicity, we only describe the process of high-level policy optimisation. Technically, we utilise an MLP network to estimate the state-value functions $v^H(G_t; \boldsymbol{W}^H)$, where $\boldsymbol{W}^H$ is a learnable parameter. Note that the state-value function is used to evaluate how good the current state is. Then we calculate the TD target y_t^H and TD error δ_t^H via:

$$y_t^H = r_t^H + \gamma \cdot v^H(G_{t+1}; \boldsymbol{W}^H), \tag{1}$$

$$\delta_t^H = v^H(G_t; \boldsymbol{W}^H) - y_t^H, \tag{2}$$

where γ is the factor of discounted return. Finally, $\pi^H(\cdot)$ and $v^H(\cdot)$ can be optimised by gradient descent:

$$\boldsymbol{\theta}^H \leftarrow \boldsymbol{\theta}^H - \beta \cdot \delta_t^H \cdot \frac{\partial}{\partial \boldsymbol{\theta}^H} \ln(\pi^H(a_t^H | G_t; \boldsymbol{\theta}^H)), \tag{3}$$

$$\boldsymbol{W}^H \leftarrow \boldsymbol{W}^H - \alpha \cdot \delta_t^H \cdot \frac{\partial}{\partial \boldsymbol{W}^H} \ln(v^H(G_t; \boldsymbol{W}^H)), \tag{4}$$

where α, β are hyperparameters.

3.3. High-level Policy

To set sub-region for sub-goal adaptively, we propose a high-level policy network, which contains two parts, *i.e.*, Scene-aware adaptive Zone Partition (SZP) and Goal-oriented Zone Selection (GZS).

Scene-aware adaptive Zone Partition (SZP). As shown in Figure 3, at time step t, we apply an object detector on the current view images to extract objects' visual features. Then we follow [10] to adopt a cross-modal transformer to integrate current view/object visual features and language features into the topological graph $G_t = (H_t, E_t)$. G_t contains N_t^v nodes/viewpoints, and $H_t \in \mathbb{R}^{N_t^v \times D_h}$, $E_t \in \mathbb{R}^{N_t^v \times N_t^v}$ represent the node feature matrix and weighted adjacency matrix respectively. Note that E_t is initialised via the distance between corresponding viewpoints. Therefore, the goal of SZP is to perform zone partition over G_t, obtaining a set of sub-graphs $\{G_t^i\}_{i=1}^{N_t^z}$, where N_t^z denotes the zone number. Here, we set N_t^z adaptively during navigation, *i.e.*, the larger G_t is, the more zones are divided:

$$N_t^z = \lceil visit\_len_t \times ratio \rceil, \tag{5}$$

where $visit\_len_t$ is the current trajectory length, and $ratio \in (0, 1)$ is the hyperparameter.

Firstly, we treat all nodes as zone centres and aim to produce the corresponding zone features $Z_t = \{z_t^i\}_{i=1}^{N_t^v}$ for each of them as shown in Figure 3(a). Considering that z_t^i should represent the state of its surrounding, we integrate the neighbour features in an adaptive manner. Specifically, we calculate the relation score $s_t^{i,j}$ of node feature $h_t^j \in \mathbb{R}^{1 \times D_h}$ regarding to node feature $h_t^i \in \mathbb{R}^{1 \times D_h}$ via:

$$s_t^{i,j} = \begin{cases} \sigma([h_t^i W_H, h_t^j] W_S) & e_t^{i,j} < THR_d \\ -\inf & \text{else} \end{cases}, \tag{6}$$

where $\sigma(\cdot)$ is activate function, $[,]$ is concatenation, $W_H \in \mathbb{R}^{D_h \times D_h}$, $W_S \in \mathbb{R}^{2D_h \times 1}$ are learnable parameters. Note that $e_t^{i,j}$ represents the distance between nodes i and j. Since remote nodes are not considered neighbours, we set a distance threshold THR_d. Hence we obtain a relation score matrix $S_t \in \mathbb{R}^{N_t^v \times N_t^v}$. For standardisation, we perform the softmax function on each line of S_t as $S_t \leftarrow \text{softmax}(S_t)$. Then we get zone features via $Z_t = S_t H_t$.

Secondly, as shown in Figure 3(b), we grade zone feature z_t^i via calculating the representive score ϕ_t^i of node i:

$$\phi_t^i = \sigma(z_t^i W_1 + \sum(z_t^i W_2 - z_t^j W_3)), e_t^{i,j} < THR_d \quad (7)$$

where $W_1, W_2, W_3 \in \mathbb{R}^{D_h \times 1}$ are learnable parameters, and $(z_t^i W_2 - z_t^j W_3)$ indicates the difference clues between two zones. As we expect G_t to be divided into N_t^z zones, we select top N_t^z zone centres according to ϕ_t, where we denote the index of zone centres as $\hat{i}_t$. Thus the probability distribution of nodes belonging to zones is $P_t = \text{softmax}_{\hat{i}_t}(S_t) \in \mathbb{R}^{N_t^v \times N_t^z}$. Then other nodes are assigned to the selected zone centre according to P_t and distance, as shown in Figure 3(c). Therefore, we split G_t into N_t^z zones $\{G_t^i\}_{i=1}^{N_t^z}$, where $G_t^i = (H_t^i, E_t^i)$, and H_t^i, E_t^i denote node features and adjancy matrix of i-th zone respectively.

Goal-oriented Zone Selection (GZS). This part aims to select a zone G_t^* from $\{G_t^i\}_{i=1}^{N_t^z}$ for further low-level actions. Firstly, the instruction is encoded as embeddings $I \in \mathbb{R}^{L \times D_h}$ by language encoder [10, 46], where L is the length. Note that we denote the produced sentence-level feature of instruction as $\hat{I} \in \mathbb{R}^{1 \times D_h}$. Basically, we hope the selected zone for sub-goal is orientated to the instruction. Secondly, as shown in Figure 3(d), we calculate a $zone\_score_i$ for each zone via inner production function:

$$zone\_score_i = \sigma(\hat{I} W_I) \cdot \sigma(z_t^i W_Z)^{\mathrm{T}}, \quad (8)$$

where $W_I, W_Z \in \mathbb{R}^{D_h \times D_w}$ are learnable parameters. Thus we select zone $G_t^* = (H_t^*, E_t^*)$ with the highest score as the navigation area for the current sub-goal, where N_t^* is nodes number, and H_t^*, E_t^* are nodes features and adjancy matrix. After the selection is completed, the agent moves to the selected zone centre along the shortest path and performs subsequent low-level actions within the zone.

3.4. Low-level Policy

Navigation Decision. At time step t, the low-level policy network $\pi^L(\cdot)$ produces the low-level action a_t^L (*i.e.* navigation decision). Specifically, only viewpoints in the current zone $G_t^* = (H_t^*, E_t^*)$ can be taken as a navigation action. We adopt multi-layers cross-attention to evaluate each node feature h_t^{*i} in G_t^* and pick the viewpoint with highest score as a_t^L to navigate to, which is simply formulated as:

$$a_t^L = \arg\max_{i \in G_t^*} \text{CrossAttention}(h_t^{*i}, I). \quad (9)$$

State-Switcher Module (SSM). To determine whether to switch the current state to the next sub-goal, *i.e.*, applying another group of high- and low-level actions, we propose an SSM. Concretely, we evaluate the state at time step t by:

$$state\_score_t = \text{sigmoid}(\sigma(\hat{I} W_I') \cdot \sigma(h_t^c W_S)), \quad (10)$$

where $h_i^c \in \mathbb{R}^{1 \times D_h}$ is the feature of current viewpoint, and $W_I', W_S \in \mathbb{R}^{D_h \times D_w}$ are learnable parameters. Then, we set a threshold $THR_S \in [0, 1]$ for $state\_score_t$ to decide whether to switch to the next sub-goal, *i.e.*, re-divide the whole graph G and re-select G^* to navigate. If the agent keeps the current sub-goal, it will continue to navigate in the current zone via low-level actions and add newly observed viewpoints to the current zone G_t^* for updating.

3.5. Training Objectives

The training losses consist of two parts: auxiliary losses for high-level policy and action losses for low-level policy.

Auxiliary Losses. There are no expert demonstrations for the high-level action, *i.e.*, zone partition and selection. Thus we train the high-level policy network with both hierarchical RL (mentioned in Sec 3.2) and auxiliary losses, *i.e.*, zone partition loss L_{zp} and zone selection loss L_{zs}.

For L_{zp}, we divide G_t in a heuristic way and apply the results as zone partition labels. Concretely, we obtain the indexes $\bar{i}_t$ of N_t^z zone centers via the formulation:

$$\bar{i}_t = \{\text{round}(1 + (k - 0.5) \times step_t) | k = 1, \cdots, N_t^z\}, \quad (11)$$

where round$(\cdot)$ is rounding function, $step_t = (visit\_len_t - 1)/N_t^z$, $\bar{i}_t^k$ represents the index of k-th zone center. Such a process makes the centres evenly distributed. Then the zone label $\bar{z}_t^j$ of node j is calculated by nearest neighbor search: $\bar{z}_t^j = \arg\min_{i \in \bar{i}_t} e_t^{i,j}$. Besides, the probability distribution of nodes belonging to zones is P_t, where $P_t(j, i)$ is the possibility of node j belonging to zone i. L_{zp} is calculated via:

$$L_{zp} = \sum_{t=1}^{T} \sum_{j=1}^{N_t^v} -\log P_t(j, \bar{z}_t^j). \quad (12)$$

For L_{zs}, at time step t, we take $\bar{i}_t^*$ as the label, where $\bar{i}_t^*$ is the index of the zone that GT viewpoint lies in:

$$L_{zs} = \sum_{t=1}^{T} -\log[\text{softmax}(zone\_score)]_{\bar{i}_t^*}. \quad (13)$$

Curriculum Learning Strategy. Since the proposed L_{zp} and L_{zs} are not based on GT labels and only provide heuristic supervision, thus we apply them only at the beginning of the training phase to improve the initial learning robustness. After that, we utilise HRL to train the network, aiming to obtain a more flexible high-level policy.

Action Losses. Following [10, 20], we compute the navigation action loss in the low-level process via:

$$L_{nav} = -\sum_{t=1}^{T} [\log p(\bar{a}_t) + \log p(\bar{a}_t^{\pi})], \quad (14)$$

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↑
Random	11.99	8.92	2.74	1.91	1.97	-	10.76	11.93	1.76	1.01	0.96	-	10.34	8.88	2.30	1.44	1.18	-
Human	-	-	-	-	-	-	-	-	-	-	-	-	21.18	86.83	81.51	53.66	77.84	51.44
Seq2Seq [3]	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
SMNA [35]	**7.54**	43.29	41.25	39.61	30.07	28.98	**9.07**	11.28	8.15	6.44	4.54	3.61	**9.23**	8.39	5.80	4.53	3.10	2.39
RCM [53]	10.70	29.44	23.33	21.82	16.23	15.36	11.98	14.23	9.29	6.97	4.89	3.89	10.60	11.68	7.84	6.67	3.67	3.14
FAST-Mat [43]	16.35	55.17	50.53	45.50	31.97	29.66	45.28	28.20	14.40	7.19	7.84	4.67	39.05	30.63	19.88	11.61	11.28	6.08
CKR [15]	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	-
ORIST [42]	10.73	49.12	45.19	42.21	29.87	27.77	10.90	25.02	16.84	15.14	8.52	7.58	11.38	29.20	22.19	18.97	10.68	9.28
VLNBERT [20]	13.44	53.90	51.79	47.96	38.23	35.61	16.78	35.02	30.67	24.90	18.77	15.27	15.86	32.91	29.61	23.99	16.50	13.51
AirBERT [17]	15.16	48.98	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
SIA [34]	13.63	65.85	61.91	57.08	45.96	42.65	41.53	44.67	31.53	16.28	22.41	11.56	48.61	44.56	30.80	14.85	19.02	9.20
HAMT [8]	-	-	-	-	-	25.18	14.08	36.84	32.95	30.20	18.92	17.28	13.62	33.41	30.40	26.67	14.88	13.08
DUET [10]	13.86	73.86	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
HOP [44]	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
AZHP (ours)	13.95	**75.12**	**74.14**	**67.22**	**59.80**	**54.20**	22.32	**53.65**	**48.31**	**36.63**	**34.00**	**25.79**	21.84	55.31	51.57	35.85	**32.25**	**22.44**

Table 1. Comparisons on REVERIE dataset. Our AZHP significantly boosts the navigation performance, especially on val-unseen set.

where $\bar{a}_t$ is the teacher action, and $\bar{a}_t^{\pi}$ is the heuristic action label, *i.e.*, shortest path from the current node to the target. Besides, we adopt $L_{og} = -\log p(\bar{o})$ as object grounding loss, where $\bar{o}$ is the ground-truth object. Therefore, the total loss (except for HRL) is obtained as:

$$L = \lambda_1 L_{zp} + \lambda_2 L_{zs} + \lambda_3 L_{nav} + L_{og}, \quad (15)$$

where $\lambda_1, \lambda_2, \lambda_3$ are balance factors.

4. Experiments

4.1. Datasets and Metrics

We conduct extensive experiments on three datasets: REVERIE [43], SOON [61] and R2R [3].

REVERIE contains $21,702$ instructions, with an average length of 18. In each panorama, predefined object bounding boxes are provided. Apart from reaching the correct target, the agent also needs to select the correct object at the end. REVERIE contains 85 scenes, where 59 for training and val-seen, 10 for val-unseen and 16 for test-unseen.

SOON also requires the agent to choose the correct object at the end of the navigation path. However, predefined object bounding boxes are not provided, thus we use a detector to obtain bounding boxes for objects. Besides, the instruction contains a more detailed description of the goal.

R2R has $10,800$ panoramic views in 90 scenes, with $7,189$ paths sampled from the navigation graphs. Each path is equipped with several navigation instructions. The dataset is split into training, val-seen, val-unseen, and test-unseen.

Evaluation Metrics. Following prior works, we report the standard metrics: success rate (SR); trajectory length (TL); the success rate weighted by trajectory length (SPL); navigation error (NE); oracle success rate (OSR). Navigation is marked as successful only if the navigation error is below 3m. The oracle success means that the trajectory passes through successful areas. Additionally, we adopt RGS and RGSPL to evaluate object grounding for REVERIE and SOON. Concretely, RGS (remote grounding success) is the success rate of finding the correct object. RGSPL represents the RGS weighted by trajectory length.

4.2. Implementation Details

The training process mainly includes two phases: pre-training and fine-tuning. For the pre-training phase, we follow [10, 20] to adopt four proxy training tasks. Additionally, we also train our high-level policy network with the proposed auxiliary losses. Since samples in the pre-training phase are cut out from the GT trajectories, SZP, GZS and SSM are trained simultaneously on each single step. Note that our transformer layers are initialised via the pre-trained LXMERT [46]. The batch size is 64, and the learning rate is set to $5e-5$ with a weight decay of 0.01. We pre-train our model for $100,000$ iterations. The model is trained via an AdamW optimiser with a learning rate of $1e-5$ and a batch size of 8. The loss weights are $\lambda_1 = 0.1, \lambda_2 = 2.5, \lambda_3 = 2$ for balancing each loss term to the same order of magnitude. Besides, $ratio$ and THR_D are set to 0.3 and 6 respectively. The hidden state dimension D_h is 768 and D_w is 128. The pre-training and fine-tuning phases cost about 12 hours on a single NVIDIA A100 GPU, respectively.

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

REVERIE. As shown in Table 1, our proposed AZHP significantly improves the navigation performance. On the val-unseen split, compared to the previous state-of-the-art method DUET [10], AZHP improves OSR and SR by 2.58% and 1.33%. SPL is increased from 33.73% to 36.63%, RGS raises from 32.15% to 34.00%, and RGSPL raises from 23.03% to 25.79%. On the test split, AZHP brings RGS and RGSPL to a promising performance.

SOON. As shown in Table 2, our AZHP beats all previous methods by a huge margin on main metrics. For example, AZHP raises OSR by 5.28% and SR by 4.43%, respectively. Though TL of AZHP is larger, our method still gains an improvement of SPL by 4.00%, which indicates that AZHP explores the environment in a more efficient way, *i.e.*, significantly improves the navigation performance with less

Methods	Val-Unseen				
	TL↓	OSR↑	SR↑	SPL↑	RGSPL↑
GBE [61]	**28.96**	28.54	19.52	13.34	1.16
DUET [10]	36.20	50.91	36.28	22.58	3.75
AZHP (ours)	39.33	**56.19**	**40.71**	**26.58**	**5.53**

Table 2. Comparisons on SOON dataset.

Methods	Val-Unseen				Test-Unseen			
	TL↓	NE↓	SR↑	SPL↑	TL↓	NE↓	SR↑	SPL↑
Random	9.77	9.23	16	-	9.89	9.79	13	12
Human	-	-	-	-	11.85	1.61	86	76
Speak-Follow [14]	-	6.62	35	-	14.82	6.62	35	28
RCM+SIL [53]	11.46	6.09	43	-	11.97	6.12	43	38
SM [35]	-	5.52	45	32	18.04	5.67	48	35
Regretful [36]	-	5.32	50	41	13.69	5.69	48	40
EGP [13]	-	5.34	52	41	-	-	-	-
EnvDrop [47]	10.70	5.22	52	48	11.66	5.23	51	47
FedCLIP-ViL [60]	-	4.80	56	50	-	-	-	-
PREVALENT [18]	10.19	4.71	58	53	10.51	5.30	54	51
AuxRN [62]	-	5.28	55	50	-	5.15	55	51
RelGraph [19]	**9.99**	4.73	57	53	**10.29**	4.75	55	52
AP [52]	19.90	4.40	55	40	21.00	4.77	56	37
ORIST [42]	10.90	4.72	57	51	11.31	5.10	57	52
NvEM [1]	11.83	4.27	60	55	12.98	4.37	58	54
SSM [51]	20.70	4.32	62	45	20.40	4.57	61	46
SSM+CCC [50]	-	-	-	-	-	4.30	62	49
CSAP [56]	12.59	3.72	65	59	13.30	4.06	62	57
VLNBERT [20]	12.01	3.93	63	57	12.35	4.09	63	57
REM [12]	12.44	3.89	64	60	13.11	3.87	65	59
ADAPT [32]	12.21	3.77	64	58	12.99	3.79	65	59
MTVM [33]	-	3.73	66	59	-	3.85	65	59
SEvol [6]	12.26	3.99	62	57	13.40	4.13	62	57
HAMT [8]	11.46	**2.29**	66	61	12.27	3.93	65	60
DUET [10]	13.94	3.31	72	60	14.73	3.65	69	59
HOP [44]	12.27	3.80	64	57	12.68	3.83	64	59
AZHP (ours)	14.05	3.15	**72**	**61**	14.95	**3.52**	**71**	**60**

Table 3. Comparisons on R2R dataset.

cost growth on exploration. Additionally, RGSPL is also lifted up from 3.75% to 5.53%. SOON contains more challenging data than other datasets, indicating that AZHP is more capable of coping with complex scenes. Note that the results of the test split can not be evaluated on the official competition website currently.

R2R. AZHP also achieves competitive performance on the R2R dataset compared to previous methods, which is shown in Table 3. Though the SPL is the same with HAMT [8], our SR improves 6% on both val-unseen and test, which is a large margin. Besides, compared to DUET [10], AZHP also refines the navigation performance. For example, on the test split, NE is dropped from 3.65m to 3.52m. SR also rises from 69% to 71%, and SPL is lifted from 59% to 60%.

4.4. Ablation Study

High-level Action. Ablations are conducted on REVERIE. In Table 4, '#1' introduces high-level action compared to our Base-Net, where SSM, SZP and GZS are replaced by the heuristic methods. When the proposed modules are discarded, SSM is replaced by a fixed-step switching strategy, SZP is replaced by the heuristic partitioning strategy.

SSM. With the SSM added in '#2', SR is raised up from 44.90% to 46.95% obviously on val-unseen as shown in Table 4. This demonstrates that SSM successfully evaluates the current navigation state, and thus high-level action is performed more effectively. Besides, '#2' also surpasses Base-Net confirming the validity of the high-level action.

SZP. Compared with '#2', '#3' gains 2.55% and 1.27% absolute increment of SR and SPL on val-unseen, which confirms the effectiveness of the proposed SZP. With a more reasonable partition over the navigation graph, the improvement in the unseen environment is quite obvious.

GZS. With GZS added in '#4', compared to '#3', SPL is lifted from 34.13% to 36.63%, which shows that GZS can pick the appropriate zone under the guide of instruction and increase the chance for successful navigation. Besides, with GZS, the effectiveness of SZP and SSM is further reflected.

Zone Number. As shown in Table 5a, zone partition and selection refine navigation performance by narrowing the action space. However, an excessive number of zones may hurt the performance (*e.g.* $ratio = 0.8$) since the representations of zones may be blurred, which brings additional difficulties for selection.

Distance Threshold in SZP. Results in Table 5b reveal how THR_d affects the performance, where higher THR_d indicates a more sparse graph. When $THR_d = 6$, the performance is greatly improved on val-unseen. Such gain confirms that sparse graph representation provides more generalizable information. However, there is a trade-off since an excessively sparse graph (*e.g.* $THR_d = 8$) otherwise loses crucial information, reducing performance.

4.5. Qualitative Analysis

To better analyse the effectiveness of SZP, we visualise two samples in Figure 4 and Figure 5, where viewpoints are shown on the top-down view of the scene. We colour viewpoints according to the zone partition and use arrows to demonstrate the trajectory.

In Figure 4(a), the agent starts at a bathroom and searches the nearby study room, finding nothing relevant. Then it returns to the hallway and labels all points as one single zone. When walking along the hallway (Figure 4(b)), it dynamically separates the zone into two parts: points along the hallway and points near the bathroom. It can be seen that our model learns to adaptively adjust zone partition as it explores, considering geometric distance, functionality and correspondence to the instruction.

As shown in Figure 5, this example is an open scene with two sets of sofas in a large living room. At first, our model divides both of them into a single zone (green in Figure 5(a)). With more detailed observation, our model learns to splits two sets of sofas into different zones (green and blue in Figure 5(b)) and successfully finds the target statue at the end of this large living room. This sample demon-

Name	Low-level Action	High-level Action	SSM	SZP	GZS	Val-Seen					Val-Unseen				
						TL↓	OSR↑	SR↑	SPL↑	RGS↑	TL↓	OSR↑	SR↑	SPL↑	RGS↑
Base-Net	✓					16.51	**78.57**	**75.61**	66.96	60.15	26.85	50.67	43.62	28.20	28.66
#1	✓	✓				14.61	78.07	75.19	**67.57**	**61.77**	24.67	51.15	44.90	31.63	29.93
#2	✓	✓	✓			14.86	76.81	75.33	67.55	61.70	25.10	53.96	46.95	32.86	31.41
#3	✓	✓	✓	✓		14.22	74.07	72.80	65.14	59.31	24.69	**56.43**	**49.50**	34.13	33.48
#4	✓	✓	✓	✓	✓	**13.95**	75.12	74.14	67.22	59.80	**22.32**	53.65	48.31	**36.63**	**34.00**

Table 4. **Ablations**. The performance is gradually improved with the continuous addition of proposed methods, especially on val-unseen.

ratio	Val-Seen				Val-Unseen			
	TL↓	SR↑	SPL↑	RGS↑	TL↓	SR↑	SPL↑	RGS↑
0	16.51	**75.61**	66.96	**60.15**	26.85	43.62	28.20	28.66
0.3	13.95	74.14	**67.22**	59.80	**22.32**	48.31	**36.63**	**34.00**
0.6	**13.70**	73.86	66.86	60.08	23.51	**50.33**	35.54	33.91
0.8	13.93	72.38	65.22	57.41	24.18	48.57	34.65	32.38

(a) **Zone Number**: We change the zone number via adjusting $ratio$. Too many or few zones hurt the performance, where $ratio = 0.3$ is the best.

THR_d	Val-Seen				Val-Unseen			
	TL↓	SR↑	SPL↑	RGS↑	TL↓	SR↑	SPL↑	RGS↑
0	**13.41**	**76.46**	**70.47**	**63.25**	24.06	46.15	32.32	29.93
3	14.86	75.33	67.55	61.70	25.10	47.95	33.86	31.41
6	13.95	74.14	67.22	59.80	**22.32**	**48.31**	**36.63**	**34.00**
8	13.77	74.35	67.58	60.58	23.89	47.69	33.69	31.55

(b) **Graph Sparsity**: When $THR_d = 0$, the graph in SZP is extremely dense, thus lacks generalisation. Setting $THR_d = 6$ obtains a balanced performance.

Table 5. **Ablations.** We conduct ablation studies on the key components of our model to further analyse the insights.

(a)

(b)

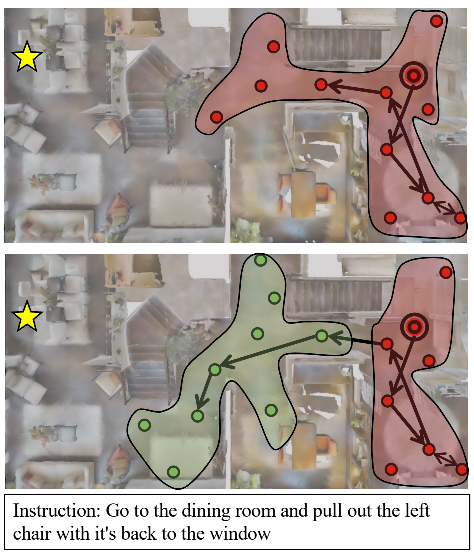

Figure 4. SZP Visualisation on val-unseen. Best viewed in colour.

(a)

(b)

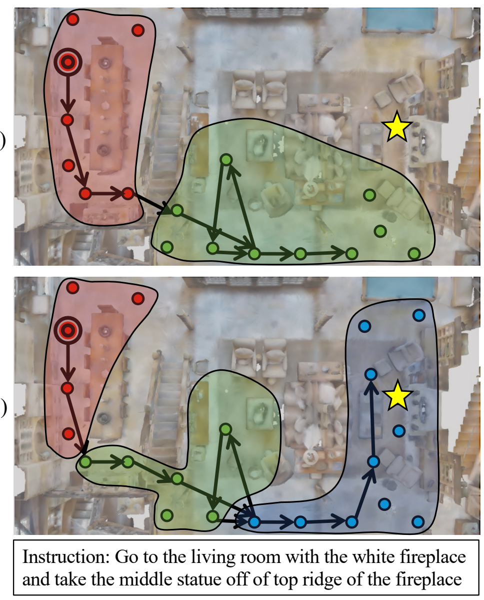

Figure 5. SZP Visualisation on val-unseen.

strates the flexibility of our SZP algorithm, which can dynamically adjust the granularity of the partition according to the specific environment and instruction.

5. Conclusion

In this paper, we propose AZHP for the VLN task, the pioneer investigation of the hierarchical navigation policy. The proposed AZHP divides the navigation process into hierarchical high-level and low-level actions. For the high-level, we devise SZP to adaptively divide the topological map into different zones online and GZS to select the zone corresponding to the sub-goal. Additionally, SSM is designed to fulfil asynchronous switching between high/low-level actions. To promote the learning process of such a hierarchical policy, we design an HRL strategy and auxiliary losses, which are performed in a curriculum learning manner. AZHP achieves state-of-the-art performance on REVERIE, SOON and R2R datasets. We believe this work will bring new insights to the VLN task and benefit following related works. Also, the code and limitation discussion are provided in the supplementary materials, and the code will be public to facilitate future research.

Acknowledgement. This research is supported by National Key R&D Program of China (2022ZD0115502), National Natural Science Foundation of China (NO. 62122010), CAAI-Huawei MindSpore Open Fund, Zhejiang Provincial Natural Science Foundation of China under Grant No.LDT23F02022F02, Key Research and Development Program of Zhejiang Province under Grant 2022C01082.

References

[1] Dong An, Yuankai Qi, Yan Huang, Qi Wu, Liang Wang, and Tieniu Tan. Neighbor-view enhanced model for vision and language navigation. In *ACM MM*, 2021. 7

[2] Peter Anderson, Ayush Shrivastava, Devi Parikh, Dhruv Batra, and Stefan Lee. Chasing ghosts: Instruction following as bayesian state tracking. In *NeurIPS*, 2019. 2

[3] 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*, 2018. 2, 3, 6

[4] Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. In *AAAI*, 2017. 3

[5] Angel X. Chang, Angela Dai, T. Funkhouser, Maciej Halber, M. Nießner, M. Savva, Shuran Song, A. Zeng, and Y. Zhang. Matterport3d: Learning from rgb-d data in indoor environments. In *3DV*, 2017. 2

[6] Jinyu Chen, Chen Gao, Erli Meng, Qiong Zhang, and Si Liu. Reinforced structured state-evolution for vision-language navigation. In *CVPR*, 2022. 7

[7] Kevin Chen, Junshen K Chen, Jo Chuang, Marynel Vázquez, and Silvio Savarese. Topological planning with transformers for vision-and-language navigation. In *CVPR*, 2021. 2

[8] Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. History aware multimodal transformer for vision-and-language navigation. In *NeurIPS*, 2021. 2, 6, 7

[9] Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. Learning from unlabeled 3d environments for vision-and-language navigation. In *ECCV*, 2022. 2

[10] 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*, 2022. 2, 3, 4, 5, 6, 7

[11] Wenhao Cheng, Xingping Dong, Salman Khan, and Jianbing Shen. Learning disentanglement with decoupled labels for vision-language navigation. In *ECCV*, 2022. 2

[12] Liu Chong, Zhu Fengda, Chang Xiaojun, Liang Xiaodan, Ge Zongyuan, and Shen Yi-Dong. Vision-language navigation with random environmental mixup. In *ICCV*, 2021. 7

[13] Zhiwei Deng, Karthik Narasimhan, and Olga Russakovsky. Evolving graphical planner: Contextual global planning for vision-and-language navigation. In *NeurIPS*, 2020. 2, 7

[14] 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*, 2018. 2, 7

[15] Chen Gao, Jinyu Chen, Si Liu, Luting Wang, Qiong Zhang, and Qi Wu. Room-and-object aware knowledge reasoning for remote embodied referring expression. In *CVPR*, 2021. 6

[16] Jing Gu, Eliana Stefani, Qi Wu, Jesse Thomason, and Xin Eric Wang. Vision-and-language navigation: A survey of tasks, methods, and future directions. In *ACL*, 2022. 2

[17] Pierre-Louis Guhur, Makarand Tapaswi, Shizhe Chen, Ivan Laptev, and Cordelia Schmid. Airbert: In-domain pretraining for vision-and-language navigation. In *ICCV*, 2021. 2, 6

[18] Weituo Hao, Chunyuan Li, Xiujun Li, Lawrence Carin, and Jianfeng Gao. Towards learning a generic agent for vision-and-language navigation via pre-training. In *CVPR*, 2020. 2, 7

[19] Yicong Hong, Cristian Rodriguez, Yuankai Qi, Qi Wu, and Stephen Gould. Language and visual entity relationship graph for agent navigation. In *NeurIPS*, 2020. 2, 7

[20] Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. Vln bert: A recurrent vision-and-language bert for navigation. In *CVPR*, 2021. 2, 5, 6, 7

[21] Muhammad Zubair Irshad, Chih-Yao Ma, and Zsolt Kira. Hierarchical cross-modal agent for robotics vision-and-language navigation. In *ICRA*, 2021. 3

[22] Vihan Jain, Gabriel Magalhaes, Alexander Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. Stay on the path: Instruction fidelity in vision-and-language navigation. In *ACL*, 2019. 2, 3

[23] Leslie Pack Kaelbling. Hierarchical learning in stochastic domains: Preliminary results. In *ICML*, 1993. 3

[24] Liyiming Ke, Xiujun Li, Yonatan Bisk, Ari Holtzman, Zhe Gan, Jingjing Liu, Jianfeng Gao, Yejin Choi, and Siddhartha Srinivasa. Tactical rewind: Self-correction via backtracking in vision-and-language navigation. In *CVPR*, 2019. 2

[25] Jing Yu Koh, Honglak Lee, Yinfei Yang, Jason Baldridge, and Peter Anderson. Pathdreamer: A world model for indoor navigation. In *ICCV*, 2021. 2

[26] Jacob Krantz, Aaron Gokaslan, Dhruv Batra, Stefan Lee, and Oleksandr Maksymets. Waypoint models for instruction-guided navigation in continuous environments. In *ICCV*, 2021. 2

[27] 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. 3

[28] 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*, 2020. 2

[29] Chengshu Li, Fei Xia, Roberto Martin-Martin, and Silvio Savarese. Hrl4in: Hierarchical reinforcement learning for interactive navigation with mobile manipulators. In *CoRL*, 2020. 3

[30] Jialu Li, Hao Tan, and Mohit Bansal. Envedit: Environment editing for vision-and-language navigation. In *CVPR*, 2022. 2

[31] Xiwen Liang, Fengda Zhu, Yi Zhu, Bingqian Lin, Bing Wang, and Xiaodan Liang. Contrastive instruction-trajectory learning for vision-language navigation. In *AAAI*, 2022. 2

[32] Bingqian Lin, Yi Zhu, Zicong Chen, Xiwen Liang, Jianzhuang Liu, and Xiaodan Liang. Adapt: Vision-language navigation with modality-aligned action prompts. In *CVPR*, 2022. 7

[33] Chuang Lin, Yi Jiang, Jianfei Cai, Lizhen Qu, Gholamreza Haffari, and Zehuan Yuan. Multimodal transformer with variable-length memory for vision-and-language navigation. In *ECCV*, 2022. 7

[34] Xiangru Lin, Guanbin Li, and Yizhou Yu. Scene-intuitive agent for remote embodied visual grounding. In *CVPR*, 2021. 6

[35] Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. Self-monitoring navigation agent via auxiliary progress estimation. In *ICLR*, 2019. 6, 7

[36] Chih-Yao Ma, Zuxuan Wu, Ghassan AlRegib, Caiming Xiong, and Zsolt Kira. The regretful agent: Heuristic-aided navigation through progress estimation. In *CVPR*, 2019. 2, 7

[37] Arjun Majumdar, Ayush Shrivastava, Stefan Lee, Peter Anderson, Devi Parikh, and Dhruv Batra. Improving vision-and-language navigation with image-text pairs from the web. In *ECCV*, 2020. 2

[38] Ofir Nachum, Shixiang Shane Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. *NeurIPS*, 2018. 3

[39] Ofir Nachum, Haoran Tang, Xingyu Lu, Shixiang Gu, Honglak Lee, and Sergey Levine. Why does hierarchy (sometimes) work so well in reinforcement learning? *arXiv preprint arXiv:1909.10618*, 2019. 3

[40] Gerhard Neumann, Wolfgang Maass, and Jan Peters. Learning complex motions by sequencing simpler motion templates. In *ICML*, 2009. 3

[41] Alexander Pashevich, Cordelia Schmid, and Chen Sun. Episodic transformer for vision-and-language navigation. In *ICCV*, 2021. 2

[42] Yuankai Qi, Zizheng Pan, Yicong Hong, Ming-Hsuan Yang, Anton van den Hengel, and Qi Wu. The road to know-where: An object-and-room informed sequential bert for indoor vision-language navigation. In *ICCV*, 2021. 6, 7

[43] 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*, 2020. 2, 3, 6

[44] 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*, 2022. 6, 7

[45] Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 1999. 3

[46] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. In *EMNLP-IJCNLP*, 2019. 5, 6

[47] Hao Tan, Licheng Yu, and Mohit Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. In *NAACL*, 2019. 2, 7

[48] Chen Tessler, Shahar Givony, Tom Zahavy, Daniel Mankowitz, and Shie Mannor. A deep hierarchical approach to lifelong learning in minecraft. In *AAAI*, 2017. 3

[49] Arun Balajee Vasudevan, Dengxin Dai, and Luc Van Gool. Talk2nav: Long-range vision-and-language navigation with dual attention and spatial memory. *ICJV*, 2021. 2

[50] Hanqing Wang, Wei Liang, Jianbing Shen, Luc Van Gool, and Wenguan Wang. Counterfactual cycle-consistent learning for instruction following and generation in vision-language navigation. In *CVPR*, 2022. 7

[51] Hanqing Wang, Wenguan Wang, Wei Liang, Caiming Xiong, and Jianbing Shen. Structured scene memory for vision-language navigation. In *CVPR*, 2021. 2, 7

[52] Hanqing Wang, Wenguan Wang, Tianmin Shu, Wei Liang, and Jianbing Shen. Active visual information gathering for vision-language navigation. In *ECCV*, 2020. 2, 7

[53] 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*, 2019. 2, 6, 7

[54] Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. Vision-language navigation policy learning and adaptation. *TPAMI*, 2020. 2

[55] Xin Wang, Wenhan Xiong, Hongmin Wang, and William Yang Wang. Look before you leap: Bridging model-free and model-based reinforcement learning for planned-ahead vision-and-language navigation. In *ECCV*, 2018. 2

[56] Siying Wu, Xueyang Fu, Feng Wu, and Zheng-Jun Zha. Cross-modal semantic alignment pre-training for vision-and-language navigation. In *ACM MM*, 2022. 7

[57] Li Xiujun, Li Chunyuan, Xia Qiaolin, Bisk Yonatan, Asli Celikyilmaz, Gao Jianfeng, A. Smith Noah, and Yejin Choi. Robust navigation with language pretraining and stochastic sampling. In *EMNLP-IJCNLP*, 2019. 2

[58] Jiwen Zhang, Jianqing Fan, Jiajie Peng, et al. Curriculum learning for vision-and-language navigation. In *NeurIPS*, 2021. 2

[59] 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*, 2022. 2

[60] Kaiwen Zhou and Xin Eric Wang. Fedvln: Privacy-preserving federated vision-and-language navigation. In *ECCV*, 2022. 7

[61] Fengda Zhu, Xiwen Liang, Yi Zhu, Qizhi Yu, Xiaojun Chang, and Xiaodan Liang. Soon: Scenario oriented object navigation with graph-based exploration. In *CVPR*, 2021. 2, 3, 6, 7

[62] Fengda Zhu, Yi Zhu, Xiaojun Chang, and Xiaodan Liang. Vision-language navigation with self-supervised auxiliary reasoning tasks. In *CVPR*, 2020. 7