Meta-Explore: Exploratory Hierarchical Vision-and-Language Navigation Using Scene Object Spectrum Grounding

Minyoung Hwang[1], Jaeyeon Jeong[1], Minsoo Kim[3], Yoonseon Oh[2], Songhwai Oh[1]
[1]Electrical and Computer Engineering and ASRI, Seoul National University
[2]Department of Electronic Engineering, Hanyang University
[3]Interdisciplinary Major in Artificial Intelligence, Seoul National University
{minyoung.hwang, jaeyeon.jeong}@rllab.snu.ac.kr, goldbird5@snu.ac.kr, yoh21@hanyang.ac.kr, songhwai@snu.ac.kr

Abstract

The main challenge in vision-and-language navigation (VLN) is how to understand natural-language instructions in an unseen environment. The main limitation of conventional VLN algorithms is that if an action is mistaken, the agent fails to follow the instructions or explores unnecessary regions, leading the agent to an irrecoverable path. To tackle this problem, we propose Meta-Explore, a hierarchical navigation method deploying an exploitation policy to correct misled recent actions. We show that an exploitation policy, which moves the agent toward a well-chosen local goal among unvisited but observable states, outperforms a method which moves the agent to a previously visited state. We also highlight the demand for imagining regretful explorations with semantically meaningful clues. The key to our approach is understanding the object placements around the agent in spectral-domain. Specifically, we present a novel visual representation, called scene object spectrum (SOS), which performs category-wise 2D Fourier transform of detected objects. Combining exploitation policy and SOS features, the agent can correct its path by choosing a promising local goal. We evaluate our method in three VLN benchmarks: R2R, SOON, and REVERIE. Meta-Explore outperforms other baselines and shows significant generalization performance. In addition, local goal search using the proposed spectral-domain SOS features significantly improves the success rate by 17.1% and SPL by 20.6% against the state-of-the-art method of the SOON benchmark. Project page: https://rllab-snu.github.io/projects/Meta-Explore/doc.html

1. Introduction

Visual navigation in indoor environments has been studied widely and shown that an agent can navigate in unexplored environments [1]. By recognizing the visual context and constructing a map, an agent can explore the environment and solve tasks such as moving towards a goal or following a desired trajectory. With the increasing development in human language understanding, vision-and-language navigation (VLN) [2] has enabled robots to communicate with humans using natural languages. The high degree of freedom in natural language instructions allows VLN to expand to various tasks, including (1) following fine-grained step-by-step instructions [2–13] and (2) reaching a target location described by goal-oriented language instructions [14–20].

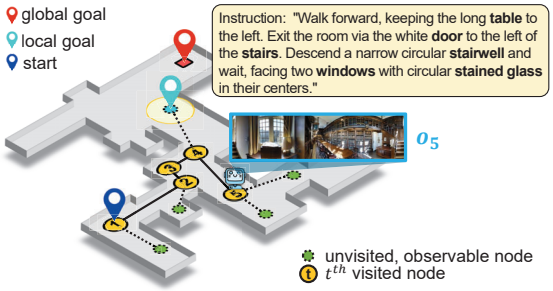

Figure 1. **Hierarchical Exploration.** At each episode, a natural-language instruction is given to the agent to navigate to a goal location. The agent explores the environment and constructs a topological map by recording visited nodes ● and next step reachable nodes ✱. Each node consists of the position of the agent and visual features. o_t denotes the observation at time t. The agent chooses an unvisited local goal to solve the regretful exploration problem.

A challenging issue in VLN is the case when an action is mistaken with respect to the given language instruction [21–26]. For instance, if the agent is asked to turn right at the end of the hallway but turns left, the agent may end up in irrecoverable paths. Several existing studies solve this issue via hierarchical exploration, where the high-level planner decides when to explore and the low-level planner chooses what actions to take. If the high-level planner chooses to explore, the agent searches unexplored regions, and if it chooses to exploit, the agent executes the best action based on the previous exploration. Prior work [21–23] returns the agent to the last successful state and resumes exploration.

This work was in part supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2019-0-01190, [SW Star Lab] Robot Learning: Efficient, Safe, and Socially-Acceptable Machine Learning, 80%, and No.2022-0-00907, Development of AI Bots Collaboration Platform and Self-Organizing AI, 20%). (Corresponding authors: Yoonseon Oh and Songhwai Oh.)

However, such methods take a heuristic approach because the agent only backtracks to a recently visited location. The agent does not take advantage of the constructed map and instead naively uses its recent trajectory for backtracking. Another recent work [26] suggests graph-based exploitation, which uses a topological map to expand the action space in global planning. Still, this method assumes that the agent can directly jump to a previously visited node. Since this method can perform a jump action at every timestep, there is no trigger that explicitly decides when to explore and when to exploit. Therefore, we address the importance of time scheduling for exploration-exploitation and efficient global planning using a topological map to avoid reexploring visited regions.

We expand the notion of hierarchical exploration by proposing Meta-Explore, which not only allows the high-level planner to choose when to correct misled local movements but also finds an unvisited state inferred to be close to the global goal. We illustrate the overview of hierarchical exploration in Figure 1. Instead of backtracking, we present an exploitation method called local goal search. We show that it is more efficient to plan a path to a local goal, which is the most promising node from the unvisited but reachable nodes. We illustrate the difference between conventional backtracking and local goal search in Figure 2. Based on our method, we show that exploration and exploitation are not independent and can complement each other: (1) to overtake regretful explorations, the agent can perform exploitation and (2) the agent can utilize the constructed topological map for local goal search. We also highlight the demand for imagining regretful explorations with semantically meaningful clues. Most VLN tasks require a level of understanding objects nearby the agent, but previous studies simply encode observed panoramic or object images [2, 3, 16–18, 21–35]. In this paper, we present a novel semantic representation of the scene called *scene object spectrum* (SOS), which is a matrix containing the arrangements and frequencies of objects from the visual observation at each location. Using SOS features, we can sufficiently estimate the context of the environment. We show that the proposed spectral-domain SOS features manifest better linguistic interpretability than conventional spatial-domain visual features. Combining exploitation policy and SOS features, we design a navigation score that measures the alignment between a given language instruction and a corrected trajectory toward a local goal. The agent compares local goal candidates and selects a near-optimal candidate with the highest navigation score from corrected trajectories. This involves high-level reasoning related to the landmarks (e.g., bedroom and kitchen) and objects (e.g., table and window) that appear in the instructions.

The main contributions of this paper are as follows:

- We propose a hierarchical navigation method called Meta-Explore, deploying an exploitation policy to correct misled recent actions. The agent searches for an appropriate local goal instead of reversing the recent action sequence.
- In the exploitation mode, the agent uses a novel scene representation called *scene object spectrum* (SOS), which contains the spectral information of the object placements in the scene. SOS features provide semantically meaningful clues to choose a near-optimal local goal and help the agent to solve the regretful exploration problem.
- We evaluate our method on three VLN benchmarks: R2R [2], SOON [16], and REVERIE [17]. The experimental results show that the proposed method, Meta-Explore, improves the success rate and SPL in test splits of R2R, SOON and val split of REVERIE. The proposed method shows better generalization results compared to all baselines.

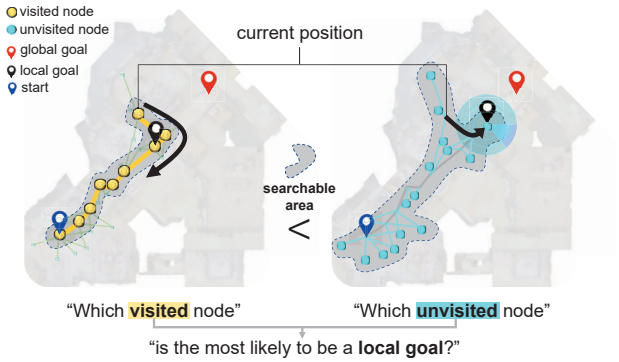

Figure 2. **Local Goal Search for Exploitation.** The local goal is likely to be chosen as the closest node to the global goal. Existing methods only backtrack to a visited node (left). We expand the searchable area by including unvisited but reachable nodes (right).

2. Related Work

2.1. Vision-and-Language Navigation

In VLN, an agent encodes the natural language instructions and follows the instructions, which can be either (1) a fine-grained step-by-step instruction the agent can follow [2–4], (2) a description of the target object and location [16, 17], or (3) additional guidance given to the agent [18, 27]. These tasks require the agent to recognize its current location using some words in the natural-language instructions. Prior work [2, 28–31] show that an agent can align visual features to language instructions via neural networks and use the multimodal output embeddings to generate a suitable action at each timestep. Most VLN methods utilize cross-modal attention, either with recurrent neural networks [2, 28] or with transformer-based architectures [29–31]. For sequential action prediction, Hong *et al.* [32] further use recurrent units inside transformer architectures, while Pashevich *et al.* [33] and Chen *et al.* [34] use additional transformers to embed past observations and actions.

2.2. Exploration-Exploitation

In an unseen environment, the agent must maximize the return without knowing the true value functions. One of the solutions to this problem is to switch back and forth between exploration and exploitation [36]. In the exploration mode, the agent gathers more information about the environment. On the other hand, the agent uses information collected during exploration and chooses the best action for exploitation. Ecoffet *et al.* [37] reduced the exploration step by archiving the states and exploring again from the successful states. Pislar *et al.* [38] addressed the various scheduling policies and demonstrated their method on Atari games. Recent work [39, 40] successfully demonstrates the effectiveness of hierarchical exploration in image-goal navigation.

Like commonly used greedy navigation policies, VLN tasks also deal with the problem of maximizing the chance to reach the goal without knowing the ground truth map. Several VLN methods employ the concept of exploitation to tackle this problem. Ke *et al.* [35] look forward to several possible future trajectories and decide whether to backtrack or not and where to backtrack. Others [21–23] estimate the progress to tell whether the agent becomes lost and make the agent backtrack to a previously visited location to restart exploration. However, previous studies do not take into account what should be done in the exploitation mode. In order to handle this problem, we propose a hierarchical navigation method which determines the scheduling between exploration and exploitation.

2.3. Visual Representations

Popular visual encoding methods via ResNet [41] and ViT [42] can be trained to learn rotation-invariant visual features. Both methods learn to extract visual features with high information gain for global and local spatial information. The high complexity of the features leads to low interpretability of the scene and therefore requires the agent to use additional neural networks or complex processing to utilize them. On the other hand, traditional visual representation methods such as Fourier transform use spectral analysis, which is highly interpretable and computationally efficient. One drawback of the traditional methods is that they fail to maximize the information gain. Nonetheless, an appropriate use of essential information can be helpful for high-level decision making and enables more straightforward interpretation and prediction of the visual features. One traditional navigation method, Sturz *et al.* [43] used Fourier transform to generate rotation-invariant visual features. However, no research has transformed the spectral information of the detected objects to represent high-level semantics from visual observations. Focusing on the fact that 2D Fourier transform can extract morphological properties of images [44], we can find out the shape or structure of detected objects through 2D Fourier transform. In this paper, we decompose the object mask into binary masks by object categories and perform a 2D Fourier transform on each binary mask.

3. Method

3.1. Problem Formulation

We deal with VLN in discrete environments, where the environment is given as an undirected graph $G_e = \{V, E\}$. V denotes a set of N navigable nodes, $\{V_i\}_{i=1}^{N}$, and E is the adjacency matrix describing connectivity among the nodes in V. We denote the observation at node V_i as O_i. The agent uses a panoramic RGB image observation o_t and current node v_t, which are collected at time t. The agent either moves to a neighboring node or executes a `stop` action. a_t denotes the action at time t. The objectives of VLN are categorized as follows: (1) to follow language instructions [2] and (2) to find a target object described by language instructions in a fixed time T [16, 17]. We present a general hierarchical exploration method that can be applied to both tasks. We also enhance the navigation policy by extracting cross-domain visual representations from the environments, i.e., spatial-domain and spectral-domain representations. To balance the information loss and interpretability of the visual feature, we adopt multi-channel fast Fourier transform (FFT) to encode semantic masks of the detected objects into category-wise spectral-domain features.

3.2. Meta-Explore

We design a learnable hierarchical exploration method for VLN called Meta-Explore, which decides (1) when to *explore* or *exploit* and (2) a new imagined local goal to seek during exploitation. The overall network architecture of the proposed Meta-Explore is shown in Figure 3. Given a language instruction L, the agent navigates in the environment until it finds the target described in L. Meta-Explore consists of a mode selector and two navigation modules corresponding to two modes: exploration and exploitation. At each timestep, the mode selector chooses to explore or exploit. At $t = 0$, the mode is initialized to exploration. In the *exploration mode*, the agent outputs an action toward a neighboring node to move the agent toward the goal. When the mode selector recognizes that the agent is not following the instruction successfully, the mode is switched to exploitation. In the *exploitation mode*, the agent seeks a new *local goal* with the highest correspondence against the language instructions from the previously unvisited candidate nodes using spectral-domain visual features. The agent moves toward the local goal by planning a path. After the agent arrives at the local goal, the mode is reset to exploration. The explore-exploit switching decision occurs through the mode selector by estimating the probability to explore. The agent repeats this explore-exploit behavior until it determines that the target is found and decides to stop.

3.2.1 Mode Selector

At time t, the agent observes visual features about the current node v_t and several reachable nodes. We call the nodes reachable at the current timestep as candidate nodes. n_{cand}

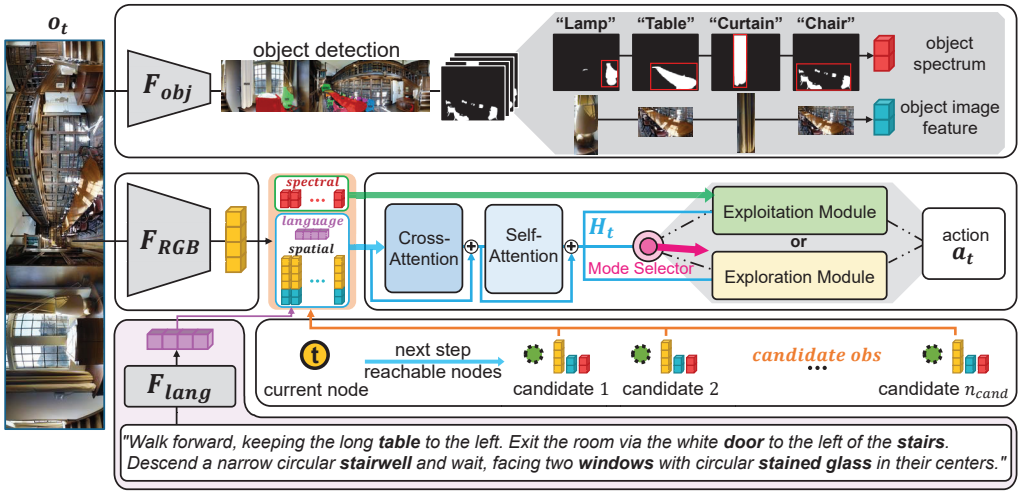

Figure 3. **Network Architecture.** Three types of visual features: panoramic (yellow), object image (aquamarine), and object spectrum (red) are encoded. The color in each parenthesis denotes the color describing the corresponding feature. The cross-modal transformer encodes language and spatial visual features as hidden state H_t. A mode selector gives *explore* or *exploit* command to the agent by predicting the explore probability $P_{explore}$. The selected navigation module outputs an action a_t from the possible n_{cand} candidate nodes.

denotes the number of candidate nodes. We use a cross-modal transformer with n_L layers to relate visual observations to language instructions. The cross-modal transformer takes the visual features of nodes in the constructed topological map at time t, G_t, and outputs cross-modal embedding H_t to encode visual observations with L. We concatenate location encoding and history encoding [24] to the visual features as node features to consider the relative pose from v_t and the last visited timestep of each node, respectively. Each word is encoded via a pretrained language encoder [45], which is used for general vision-language tasks.

The cross-modal transformer consists of cross-attention layer $\text{L2V\_Attn}(\hat{W}, \hat{V}) = \text{Softmax}(\hat{W}\Theta_q^\dagger(\hat{V}\Theta_k^\dagger)^T/\sqrt{d})\hat{V}\Theta_v^\dagger$ and self-attention layer $\text{SelfAttn}(X) = \text{Softmax}\ ((X\Theta_q(X\Theta_k)^T + A\Theta_e + b_e)/\sqrt{d})X\Theta_v$, where $\hat{W}$, $\hat{V}$, X, and A denote word, visual, node representations and adjacency matrix of G_t, respectively. The (query, key, value) weight matrices of self-attention and cross-attention layers are denoted as $(\Theta_q, \Theta_k, \Theta_v)$ and $(\Theta_q^\dagger, \Theta_k^\dagger, \Theta_v^\dagger)$, respectively. The final cross-modal embedding at time t after passing through n_L transformer layers is denoted as H_t. To encourage the monotonic increasing relationship between language and visual attentions at each timestep, we define a correlation loss $L_{corr} = \sum_{t=1}^{T} ||\text{L2V\_Attn} - I_{n_x}||_1$ for training the cross-modal transformer, where n_x denotes the dimension of the H_t and I_{n_x} denotes an identity matrix of size $n_x \times n_x$.

As illustrated in Figure 4, the mode selector estimates the probability to explore $P_{explore}$ given the cross-modal hidden state H_t. We denote the mode selector as S_{mode} and use a two-layer feed-forward neural network. Given H_t, S_{mode} outputs the exploration probability as $P_{explore} = 1 - S_{mode}(H_t)$. If $P_{explore} \geq 0.5$, the exploration policy outputs a probability distribution for reachable nodes at the next step. At time $t+1$, the agent moves to the node with the highest probability. If $P_{explore} < 0.5$, the agent determines that the current trajectory is regretful, so the agent should traverse to find a local goal, which is the most likely to be the closest node to the global goal. The exploitation policy mainly utilizes object-level features to search for the local goal with high-level reasoning. After the local goal is chosen, the path planning module outputs an action following the shortest path to the local goal.

To train the mode selector, we require additional demonstration data other than the ground truth trajectory, such that it switches between exploration and exploitation. We generate the demonstration data from the ground truth trajectories, with additional detours. For the detours, we stochastically select candidate nodes other than the ground truth paths and add the trajectory that returns to the current viewpoint. The imitation learning loss for training the mode selector is defined as $L_{mode} = \sum_{t=1}^{T} \mathbb{1}(m_t = \text{gt}_t)$, where m_t is the mode of the agent, 0 for exploitation and 1 for exploration. gt_t is 1 if the current node is in the shortest ground truth trajectory and $\text{gt}_t = 0$, otherwise.

3.2.2 Exploration Module

In the exploration mode, the agent follows the following sequential operations: topological map construction, self-monitoring, and an exploration policy. To improve the exploration, we adopt self-monitoring [21] to predict the current progress of exploration to enhance the exploration policy itself. Prior work [21, 22] has shown that auxiliary loss using self-monitoring can regularize the exploration policy.

Topological Map Construction. The agent constructs graph G_t by classifying nodes into two types: (1) visited nodes and (2) unvisited but observable nodes. At current

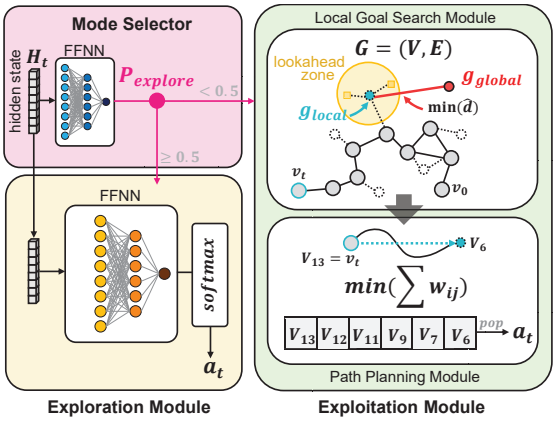

Figure 4. **Navigation Modules.** Mode selector estimates $P_{explore}$, i.e., the probability to explore, and chooses between exploration and exploitation modules. The selected navigation module outputs the next action a_t.

time t, the agent at node $v_t \in \{V_i\}_{i=1}^{N}$ observes $N(v_t)$ neighbor nodes as next step candidates at time $t+1$. The visited nodes consist of visual features of their own and the neighboring nodes from panoramic RGB observations. The unvisited nodes can be observed only if they are connected to at least one visited node. The topological map records the positions and visual features of observed nodes at each timestep. By knowing the positions of nodes in G_t, the agent can plan the shortest path trajectory between two nodes.

Self-Monitoring. We use a progress monitor to estimate the current navigation progress at each episode. Self-monitoring via estimating current progress helps the agent choose the next action that can increase the progress. The estimated progress $\hat{p}_t = F_{progress}(H_t)$ is the output of a feed-forward neural network, given H_t as input. We measure the ground truth progress p_t as the ratio between the current distance to the goal and the shortest path length of the episode subtracted from 1, described as $1 - \frac{d_{geo}(v_t, v_{goal})}{d_{geo}(v_0, v_{goal})}$, where $d_{geo}(a, b)$ is the geodesic distance between a and b. v_0, v_t, and v_{goal} denote initial, current, and goal positions, respectively. We add progress loss $L_{progress} = \sum_{t=1}^{T}(\hat{p}_t - p_t)^2$ to train the progress monitor while training the exploration policy.

Exploration Policy. The exploration policy $F_{explore}$ estimates the probability of moving to the candidate nodes at the next step. The agent chooses the action a_t at time t based on the estimated probability distribution among candidate nodes, described as $a_t = \arg\max_{V_i}(F_{explore}([H_t]_i))$. $F_{explore}$ is implemented via a two-layer feed-forward network with the cross-modal hidden state H_t given as input. The output of $F_{explore}$ becomes a probability distribution over possible actions. To only consider unvisited nodes, we mask out the output for visited nodes. For training, we sample the next action from the probability distribution instead of choosing a node with the highest probability. We describe the training details in Section 3.3.

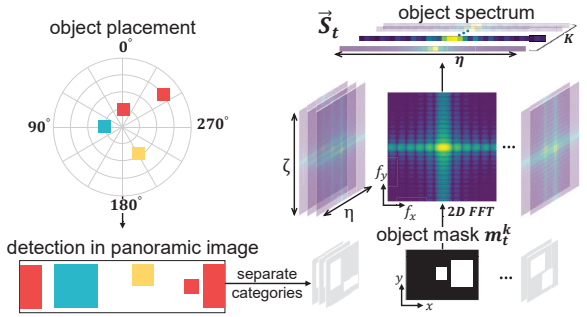

Figure 5. **Scene Object Spectrum (SOS).** The agent calculates *scene object spectrum* (SOS) features for efficient exploitation. SOS features incorporate semantic information observed in a single panoramic image by performing category-wise 2D FFT.

3.2.3 Exploitation Module

In the exploitation mode, the agent requires high-level reasoning with identifiable environmental clues to imagine regretful exploration cases. To find clues in an object-level manner, we present a novel visual representation by capturing object information in the spectral-domain. The novel representation is more easily predictable than spatial features such as RGB image embeddings. The agent can take advantage of the predictability by expanding the searchable area to find a local goal. We choose the local goal as the closest node to the global goal in the feature space.

Spectral-Domain Visual Representations. Common navigation policies can lead the agent toward the node with the highest similarity to the target. However, even with a good learned policy, the agent can act in a novice manner in unseen environments. In this paper, we seek extra information from the environment for generalizable high-level reasoning to resolve the issue. As illustrated in Figure 5, *scene object spectrum* (SOS) incorporates semantic information observed in a single panoramic image by generating a semantic mask for each object category and applying Fourier transform to each semantic mask. The semantic mask for object class k at time t is calculated as a binary mask $[m_t^k]_{ij}$ that detects the object at pixel $(i,\ j)$. Suppose there are a total of K object categories. When multiple objects are detected for one object category, the binary mask appears as a union of the bounding boxes of the detected objects. We define $\mathbf{FFT}$ as a channel-wise 2D fast Fourier transform that receives K binary semantic masks and outputs K spectral-domain features, where K is the number of object classes. Then, SOS feature $\vec{S}_t = [s_t^1, ..., s_t^K]^T$ can be defined as $s_t^k = \log|\mathbf{FFT}(m_t^k)|$. For simplicity, we perform mean pooling on the vertical spectral axis and normalize the output. The final SOS feature has shape (K, η), where η is the maximum horizontal frequency.

Local Goal Search Using Semantic Clues. We argue that returning to a previously visited node does not guarantee the agent escapes from the local optima. Instead of backtracking

to a previously visited node, the agent searches for a local goal to move towards. If the agent plans a path and moves towards the local goal, the agent does not need to repeat unnecessary actions in visited regions after the exploitation ends. Additionally, searching for a local goal takes full advantage of the topological map by utilizing the connections among the observed nodes. To expand the searchable area further, we let the agent choose the local goal from previously unvisited and unchosen candidate nodes.

To choose a local goal, we first score the corrected trajectories to measure the alignment with the language instruction L. We use SOS features as semantic environmental clues to estimate the navigation score S_{nav} of the corrected trajectory, which is the shortest path trajectory from the initial node to the local goal in the constructed topological map. To simplify, we convert the language instruction into a list of objects $W^o = [w_1^o, ..., w_B^o]$ consisting of $B[\leq K]$ object categories (e.g., desk, cabinet, and microwave). We approximate the corresponding reference SOS features as $[\hat{\delta}(w_1^o), ..., \hat{\delta}(w_B^o)]$ where the i^{th} row of $\hat{\delta}(w_k^o)$ is defined as $[\hat{\delta}(w_k^o)]_i = \mathbb{1}(k=i)\lambda(\delta(w_k^o))\,\mathrm{sinc}(\frac{j}{2} - \frac{\eta}{4})$. $\lambda(\delta(w_k^o))$ denotes the average width of detected bounding boxes of object w_k^o in the environment. A detailed approximation process is explained in the supplementary material. To simulate a corrected trajectory $\mathcal{T}' = (v_1', ..., v_{t'}')$, we calculate the SOS features $[\vec{S}_1', ..., \vec{S}_{t'}']$ corresponding to the nodes in $\mathcal{T}'$. We measure the similarity between two object spectrum features via the cosine similarity of the flattened vectors. Finally, the navigation score S_{nav} of $\mathcal{T}'$ is computed as:

$$S_{nav}(\mathcal{T}') = \frac{\sum_{i=1}^{B}\sum_{j=1}^{t'}(\frac{\hat{\delta}(w_i^o)}{|\hat{\delta}(w_i^o)|}\cdot\frac{\vec{S}_j'}{|\vec{S}_j'|})((\hat{\delta}(w_i^o)-\hat{\delta}(\overline{w}^o))\cdot(\vec{S}_j'-\overline{\vec{S}}'))}{\sqrt{\frac{t'}{B}\cdot\sum_{i=1}^{B}(\hat{\delta}(w_i^o)-\hat{\delta}(\overline{w}^o))^2\sum_{j=1}^{t'}(\vec{S}_j'-\overline{\vec{S}}')^2}}, \tag{1}$$

where $\hat{\delta}(\overline{w}^o)$ and $\overline{\vec{S}}'$ denote the average values of SOS features $\hat{\delta}(w_i^o)$ and $\vec{S}_j'$, respectively. This equation can also be interpreted as a pseudo correlation-coefficient function between object list W^o and trajectory $\mathcal{T}'$. The exploitation policy selects the node with the highest navigation score as the local goal from the previously unvisited candidates.

Figure 6 illustrates a simple scenario of entering a room. Suppose $W^o = [\text{sculpture}, \text{door}, \text{bed}]$ and the agent has to compare two trajectories $\mathcal{T}_1 = (v_1, v_2, A)$ and $\mathcal{T}_1 = (v_1, v_2, B)$. Each similarity matrix in Figure 6 has the (t, j) element as the similarity between the SOS feature of V_t and $\hat{\delta}(w_j^o)$, which is calculated as $\hat{\delta}(w_j^o)\cdot\vec{S}_t'$. Notably, the similarity matrix shows monotonic alignment and the navigation score is higher when the next action is chosen correctly.

3.3. Training Details

We use [24] for pretraining the visual encoder with panoramic RGB observations. We use the DAgger algorithm [46] to pretrain the navigation policy and the mode selector. To prevent overfitting, we iteratively perform

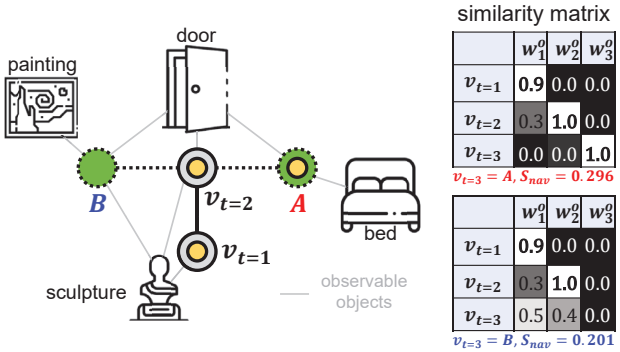

Figure 6. **Toy Example.** Monotonic alignment between language instruction and visual observation is desirable. Yellow dots in the nodes describe the ground truth trajectory. Based on the node at $t = 3$, the similarity matrix can show either monotonic or non-monotonic alignment between object tokens and SOS features. The green circles describe the possible candidates A, B for next action.

teacher forcing and student forcing to choose the action from the exploration policy. Imitation learning loss is calculated as $L_{IL} = \sum_{t=1}^{T} -\log p(a_t^*|a_t)$ and object grounding loss is calculated as $L_{OG} = -\log p(\text{obj}^*|\text{obj}_{pred})$, where obj* denotes the ground truth and obj$_{pred}$ denotes the predicted object location. The total loss function is defined as $L_{total} = L_{mode} + L_{progress} + L_{corr} + L_{IL} + L_{OG}$. We further finetune the agent via A2C [47]. The exploration policy selects the action a_t with probability p_t^a. Reinforcement learning loss is defined as $L_{RL} = -\sum_t a_t^s \log(p_t^a) A_t - \lambda \sum_t a_t^* \log(p_t^a)$. To train the mode selector, progress monitor, and exploration policy in an end-to-end manner, we use the total loss function as $L_{fine} = L_{mode} + L_{progress} + L_{RL}$. The exploitation policy searches the path toward the local goal from the constructed navigation graph. Thus, the exploitation policy is not learned.

4. Navigation Experiments

4.1. Experiment Settings

We evaluate our method on three VLN benchmarks, Room-to-Room(R2R) [2], SOON [16], and REVERIE [17].

R2R evaluates the visually-grounded natural navigation performance of the agent. The agent must navigate to the predefined goal point given image observations and language instructions in an unseen environment.

SOON is also a goal-oriented VLN benchmark. Natural language instructions in SOON have an average length of 47 words. The agent should locate the target location and detect the location of an object to find the target object.

REVERIE is a goal-oriented VLN benchmark that provides natural language instruction about target locations and objects. In REVERIE, the agent is given an instruction referring to a remote object with an average length of 21 words. With this instruction and a panoramic observation from the environment, the agent should navigate to the location the instruction describes and find the correct object

bounding box among the predefined object bounding boxes.

4.2. Evaluation Metrics

4.2.1 Navigation performance

We evaluate algorithms using the trajectory length (TL), success rate (SR), and success weighted by inverse path length (SPL) [48], and oracle success rate (OSR) for the navigation performance comparison. An episode is recorded as a success if the agent takes a `stop` action within 3 m of the target location. TL is the average path length in meters. SR is denoted as the number of successes divided by the total number of episodes, M. SPL is calculated as $\frac{1}{M}\sum_{i=1}^{M} S_i \frac{l_i}{\max(p_i,l_i)}$, where S_i denotes the success as a binary value. p_i and l_i denote the shortest path and actual path lengths for the i^{th} episode. OSR uses the oracle stop policy instead of the stop policy of the agent.

4.2.2 Object grounding performance

We also evaluate the object grounding performance of the agent by the success rate of finding the target object (FSR) and the target finding success weighted by inverse path length (FSPL)[1] [16, 17]. FSPL is calculated as $\text{FSPL} = \frac{1}{N}\sum_{i=1}^{N} S_i^{nav} S_i^{loc} \cdot l_i^{nav}/\max(l_i^{nav}, l_i^{gt})$, where S_i^{nav} is whether the agent navigates to the target, S_i^{loc} is whether the agent finds a target object bounding box, and l_i^{nav} and l_i^{gt} are the navigation trajectory length and ground truth trajectory length, respectively.

4.3. Baselines and Implementation Details

We compare our method with several other baselines as follows. For each task, we compare our method with a number of baselines that use various types of memory (recurrent, sequential, and topological map). For methods implemented with a hierarchical navigation framework, we compare the specific exploitation methods: homing, jump, and local goal search. Homing makes the agent backtrack, and jump makes the agent jump to a previously visited node. The hyperparameters and detailed model architecture of Meta-Explore are described in the supplementary material.

4.4. Comparison with Navigation Baselines

We compare our method with navigation baselines[2]. We focus on the success rate and SPL. Rendered results and detailed analyses with other evaluation metrics are provided in the supplementary material.

R2R. Table 1 compares the proposed Meta-Explore with baselines for the R2R navigation task. We categorize the baseline methods based on the type of constructed memory and the type of exploitation. Our method outperforms other exploration-only baselines over all types of validation and test splits in success rate and SPL. Compared with hierarchical baselines SMNA [21], Regretful-Agent [22], FAST [35], and SSM [26], Meta-Explore improves success rate and SPL by at least 16.4% and 8.9%, respectively. The main difference is that Meta-Explore constructs a topological map during exploration and uses the map for local goal search in exploitation. On the contrary, homing exploitation policies in SMNA, Regretful-Agent, and FAST only rely on the current trajectory, instead of taking advantage of the constructed memory. Jump exploitation in SSM uses a topological map to search a successful previous node, but it makes an unrealistic assumption that the agent can directly jump to a previously visited distant node and unfairly saves time. In our approach, we plan a path to the local goal based on the topological map. The experiment results reveal that even if we design a hierarchical navigation framework, exploration and exploitation are not entirely separate but they can complement each other.

SOON, REVERIE. Table 2 compares Meta-Explore with baselines in the SOON navigation task. While the proposed method does not improve performance in val seen split, Meta-Explore outperforms other baselines in the test unseen split of SOON for success rate by 17.1% and SPL by 20.6%. The result implies that for the goal-oriented VLN task, high performance in train or val seen splits can be the overfitted result. Because the agent can be easily overfitted to the training data, making a generalizable model or providing a deterministic error-correction module for inference is essential. Meta-Explore chooses the latter approach by correcting the trajectory via exploitation in regretful cases. The evaluation results in the REVERIE navigation task are described in the supplementary material. Meta-Explore shows improvement in the val split of REVERIE for success rate and SPL, but the improvement in the test split is lower than the results in R2R and SOON. We found 252 meaningless object categories (e.g., verbs, adjectives, and prepositions) and 418 replaceable object categories (e.g., typographical errors and synonyms) in the REVERIE[3] dataset. Because our exploitation method utilizes object-based parsing of the given instruction to match with the detected object categories, the effectiveness of the proposed method is lessened due to inaccuracies and inconsistencies in the dataset. We expect to have higher performance if the mistakes in the dataset are fixed.

4.5. Local Goal Search using SOS Features

To discuss the significance of modeling exploitation policy, we conduct specific experiments about choosing the local goal for R2R and SOON. We evaluate our method using different types of local goal search, as shown in Table 3 and 4. Oracle denotes a method which selects a local goal using the ground truth trajectory. The performance of the oracle provides the achievable performance for each dataset. The results imply that local goal search using either

[1] Identical with its original term, Remote Grounding Success (RGS).

[2] † indicates reproduced results.

[3] 10.7% and 41.2% of a total of 46,476 words in the bounding box dataset correspond to meaningless and replaceable object categories, respectively.

Methods	Memory	Exploit	Val Seen				Val Unseen				Test Unseen			
			SR↑	SPL↑	TL↓	NE↓	SR↑	SPL↑	TL↓	NE↓	SR↑	SPL↑	TL↓	NE↓
Random	-	-	16	-	9.58	9.45	16	-	9.77	9.23	13	12	9.89	9.79
Human	-	-	-	-	-	-	-	-	-	-	11.85	1.61	86	76
Seq2Seq [2]	Rec	✗	6.0	39	11.33	-	22	-	**8.39**	7.84	20	18	**8.13**	7.85
VLN⟳BERT [32]	Rec	✗	72	68	**11.13**	2.90	63	57	12.01	3.93	63	57	12.35	4.09
SMNA† [21]	Rec	**homing**	69	63	11.69	3.31	47	41	12.61	5.48	61	56	-	4.48
Regretful-Agent [22]	Rec	**homing**	69	63	-	3.23	50	41	-	5.32	48	40	-	5.69
FAST (short) [35]	Rec	**homing**	-	-	-	-	56	43	21.17	4.97	54	41	22.08	5.14
FAST (long) [35]	Rec	**homing**	70	04	188.06	3.13	63	02	224.42	4.03	61	03	196.53	4.29
HAMT-e2e [34]	Seq	✗	76	72	11.15	2.51	66	61	11.46	**2.29**	65	60	12.27	3.93
DUET [24]	Top. Map	✗	79	73	12.32	2.28	**72**	60	13.94	3.31	69	59	14.73	3.65
SSM [26]	Top. Map	**jump**	71	62	14.7	3.10	62	45	20.7	4.32	61	46	20.4	4.57
Meta-Explore (Ours)	Top. Map	**local goal**	**81**	**75**	11.95	**2.11**	**72**	**62**	13.09	3.22	**71**	**61**	14.25	**3.57**

Table 1. Comparison and evaluation results of the baselines and our model in the R2R Navigation Task.
Gray shaded rows describe hierarchical navigation baselines. Three memory types: Rec(recurrent), Seq(sequential), and Top. Map(topological map)

Methods	Memory	Exploit	Val Seen Instruction				Val Seen House				Test Unseen House			
			SR↑	SPL↑	OSR↑	FSPL↑	SR↑	SPL↑	OSR↑	FSPL↑	SR↑	SPL↑	OSR↑	FSPL↑
Human	-	-	-	-	-	-	-	-	-	-	90.4	59.2	91.4	51.1
Random	Rec	✗	0.0	1.5	0.1	1.4	0.1	0.0	0.4	0.9	2.1	0.4	2.7	0.0
Speaker-Follower [28]	Rec	✗	97.9	97.7	97.8	24.5	61.2	60.4	69.4	**9.1**	7.0	6.1	9.8	0.6
RCM [49]	Rec	✗	84.0	82.6	89.1	10.9	62.4	60.9	72.7	7.8	7.4	6.2	12.4	0.7
AuxRN [23]	Rec	✗	98.4	97.4	**98.7**	13.7	68.8	**67.3**	**78.5**	8.3	8.1	6.7	11.0	0.5
GBE w/o GE	Top. Map	✗	89.5	88.3	91.8	24.2	62.5	60.8	73.0	6.7	11.4	8.7	18.8	0.8
GBE [16]	Top. Map	✗	98.4	97.9	98.6	**44.2**	**76.3**	62.5	64.1	7.3	11.9	10.2	19.5	1.4
GBE†	Top. Map	✗	-	-	-	-	19.5	13.3	28.5	1.2	12.9	9.2	21.5	0.5
DUET [24]	Top. Map	✗	94.0	91.6	90.0	31.1	36.3	22.6	50.9	3.8	33.4	21.4	43.0	**4.2**
Meta-Explore (Ours)	Top. Map	**local goal**	**100.0**	**99.1**	96.0	33.9	44.7	34.8	52.7	8.9	**39.1**	**25.8**	**48.7**	4.0

Table 2. Comparison and evaluation results of the baselines and our model in the SOON Navigation Task.

spatial or spectral visual representations is more effective than random local goal search. The results show that local goal search using spectral visual representations, i.e., SOS features, lead the agent to desirable nodes the most. We also compare local goal search with homing and the difference between the performance of the two methods is most noticeable in the test split of the SOON navigation task. As shown in Table 4, choosing the local goal with only spatial-domain features, the navigation performance does not improve compared to homing. On the contrary, spectral-domain local goal search shows significant improvement against homing by 10.4% in success rate, 34.5% on SPL, and 27.4% on FSPL. The results imply that using spectral-domain SOS features helps high-level decision making, thereby enhancing the navigation performance. To further show the effectiveness of SOS features, we provide sample local goal search scenarios in the supplementary material.

Local Goal	Val Seen					Val Unseen				
	SR↑	SPL↑	OSR↑	TL↓	NE↓	SR↑	SPL↑	OSR↑	TL↓	NE↓
Oracle	81.88	74.12	87.46	13.06	1.93	75.95	62.53	84.16	14.00	2.71
Random	79.33	72.67	85.31	13.19	2.22	70.97	59.45	80.16	14.92	3.34
Homing	80.22	73.63	85.60	12.51	2.14	71.65	60.60	80.33	13.91	3.26
Spatial	79.63	73.14	85.60	12.99	2.22	71.56	60.01	80.33	14.90	3.27
Spectral	**80.61**	**75.15**	**85.80**	**11.95**	**2.11**	**71.78**	**61.68**	**80.76**	**13.09**	**3.22**

Table 3. Comparison of Exploitation Policies. (R2R)

Local Goal	Val Seen House				Test Unseen House			
	SR↑	SPL↑	OSR↑	FSPL↑	SR↑	SPL↑	OSR↑	FSPL↑
Oracle	54.42	37.96	63.72	11.01	48.38	28.45	62.98	4.74
Random	24.78	11.97	34.96	3.08	24.19	7.41	35.84	1.29
Homing	42.04	27.72	48.23	**10.18**	35.40	19.18	**51.62**	3.14
Spatial	32.30	11.60	39.38	1.90	26.11	10.58	39.23	1.43
Spectral	**44.69**	**34.84**	**52.65**	8.89	**39.09**	**25.80**	48.67	**4.01**

Table 4. Comparison of Exploitation Policies. (SOON)

4.6. Ablation Study

We conduct an ablation study to compare the proposed method against language-triggered hierarchical exploration. Results in the supplementary material show that among the three representation domains, spatial, spectral, and language, the spectral-domain features enhance navigation performance the most. Additionally, to implicate further applications of Meta-Explore in continuous environments, we evaluate our method on the photo-realistic Habitat [50] simulator to solve image-goal navigation and vision-and-language navigation tasks. Implementation details and results are included in the supplementary material. Results show that our method outperforms baselines in both tasks.

5. Conclusion

We have proposed Meta-Explore, a hierarchical navigation method for VLN, by correcting mistaken short-term actions via efficient exploitation. In the exploitation mode, the agent is directed to a local goal which is inferred to be the closest to the target. A topological map constructed during exploration helps the agent to search and plan the shortest path toward the local goal. To further search beyond the frontier of the map, we present a novel visual representation called *scene object spectrum* (SOS), which compactly encodes the arrangements and frequencies of nearby objects. Meta-Explore achieves the highest generalization performance for test splits of R2R, SOON, and val split of REVERIE navigation tasks by showing less overfitting and high success rates. We plan to apply Meta-Explore for VLN tasks in continuous environments in our future work.

References

[1] Fengda Zhu, Yi Zhu, Vincent Lee, Xiaodan Liang, and Xiaojun Chang. Deep learning for embodied vision navigation: A survey. *arXiv preprint arXiv:2108.04097*, 2021. 1

[2] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3674–3683, 2018. 1, 2, 3, 6, 8

[3] Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12538–12547, 2019. 1, 2

[4] Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-Across-Room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *Conference on Empirical Methods for Natural Language Processing*, 2020. 1, 2

[5] 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 *Transactions of the Association for Computational Linguistics*, pages 1862–1872, Florence, Italy, July 2019. Association for Computational Linguistics. 1

[6] An Yan, Xin Eric Wang, Jiangtao Feng, Lei Li, and William Yang Wang. Cross-lingual vision-language navigation. *arXiv preprint arXiv:1910.11301*, 2019. 1

[7] Keji He, Yan Huang, Qi Wu, Jianhua Yang, Dong An, Shuanglin Sima, and Liang Wang. Landmark-rxr: Solving vision-and-language navigation with fine-grained alignment supervision. In *Proceedings of the International Conference on Neural Information Processing Systems*, volume 34, pages 652–663. Curran Associates, Inc., 2021. 1

[8] Jacob Krantz, Erik Wijmans, Arjun Majundar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision and language navigation in continuous environments. In *Proceedings of the European Conference on Computer Vision*, 2020. 1

[9] Piotr Mirowski, Andras Banki-Horvath, Keith Anderson, Denis Teplyashin, Karl Moritz Hermann, Mateusz Malinowski, Matthew Koichi Grimes, Karen Simonyan, Koray Kavukcuoglu, Andrew Zisserman, et al. The streetlearn environment and dataset. *arXiv preprint arXiv:1903.01292*, 2019. 1

[10] Karl Moritz Hermann, Mateusz Malinowski, Piotr Mirowski, Andras Banki-Horvath, Keith Anderson, and Raia Hadsell. Learning to follow directions in street view. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11773–11781, Apr. 2020. 1

[11] Arun Balajee Vasudevan, Dengxin Dai, and Luc Van Gool. Talk2nav: Long-range vision-and-language navigation with dual attention and spatial memory. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, volume 129, pages 246–266. Springer, 2021. 1

[12] Dipendra Misra, Andrew Bennett, Valts Blukis, Eyvind Niklasson, Max Shatkhin, and Yoav Artzi. Mapping instructions to actions in 3D environments with visual goal prediction. In *Proceedings of the Conference on Empirical Methods for Natural Language Processing*, pages 2667–2678, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. 1

[13] Ta-Chung Chi, Minmin Shen, Mihail Eric, Seokhwan Kim, and Dilek Hakkani-tur. Just ask: An interactive learning framework for vision and language navigation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2459–2466, 2020. 1

[14] Yi Wu, Yuxin Wu, Georgia Gkioxari, and Yuandong Tian. Building generalizable agents with a realistic and rich 3d environment. *arXiv preprint arXiv:1801.02209*, 2018. 1

[15] Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1–10, 2018. 1

[16] Fengda Zhu, Xiwen Liang, Yi Zhu, Qizhi Yu, Xiaojun Chang, and Xiaodan Liang. Soon: Scenario oriented object navigation with graph-based exploration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12689–12699, 2021. 1, 2, 3, 6, 7, 8

[17] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9982–9991, 2020. 1, 2, 3, 6, 7

[18] Khanh Nguyen and Hal Daumé III. Help, anna! visual navigation with natural multimodal assistance via retrospective curiosity-encouraging imitation learning. In *Proceedings of the Conference on Empirical Methods for Natural Language Processing*, November 2019. 1, 2

[19] Khanh Nguyen, Debadeepta Dey, Chris Brockett, and Bill Dolan. Vision-based navigation with language-based assistance via imitation learning with indirect intervention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2019. 1

[20] Alane Suhr, Claudia Yan, Jack Schluger, Stanley Yu, Hadi Khader, Marwa Mouallem, Iris Zhang, and Yoav Artzi. Executing instructions in situated collaborative interactions. In *Proceedings of the Conference on Empirical Methods for Natural Language Processing*, pages 2119–2130, 2019. 1

[21] 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 *Proceedings of the International Conference on Learning Representations*, 2019. 1, 2, 3, 4, 7, 8

[22] Chih-Yao Ma, Zuxuan Wu, Ghassan AlRegib, Caiming Xiong, and Zsolt Kira. The regretful agent: Heuristic-aided navigation

through progress estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6732–6740, 2019. 1, 2, 3, 4, 7, 8

[23] Fengda Zhu, Yi Zhu, Xiaojun Chang, and Xiaodan Liang. Vision-language navigation with self-supervised auxiliary reasoning tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10012–10022, 2020. 1, 2, 3, 8

[24] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16537–16547, 2022. 1, 2, 4, 6, 8

[25] Zhiwei Deng, Karthik Narasimhan, and Olga Russakovsky. Evolving graphical planner: Contextual global planning for vision-and-language navigation. In *Proceedings of the International Conference on Neural Information Processing Systems*, volume 33, pages 20660–20672, 2020. 1, 2

[26] Hanqing Wang, Wenguan Wang, Wei Liang, Caiming Xiong, and Jianbing Shen. Structured scene memory for vision-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8455–8464, June 2021. 1, 2, 7, 8

[27] Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. Vision-and-dialog navigation. In *Proceedings of the Conference on Robot Learning*, pages 394–406, 2020. 2

[28] 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 *Proceedings of the International Conference on Neural Information Processing Systems*, 2018. 2, 8

[29] Xiujun Li, Chunyuan Li, Qiaolin Xia, Yonatan Bisk, Asli Celikyilmaz, Jianfeng Gao, Noah A Smith, and Yejin Choi. Robust navigation with language pretraining and stochastic sampling. In *Proceedings of the Conference on Empirical Methods for Natural Language Processing*, pages 1494–1499, 2019. 2

[30] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Proceedings of the International Conference on Neural Information Processing Systems*, 2019. 2

[31] Pierre-Louis Guhur, Makarand Tapaswi, Shizhe Chen, Ivan Laptev, and Cordelia Schmid. Airbert: In-domain pretraining for vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1634–1643, 2021. 2

[32] Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. Vln◌bert: A recurrent vision-and-language bert for navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1643–1653, 2021. 2, 8

[33] Alexander Pashevich, Cordelia Schmid, and Chen Sun. Episodic transformer for vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15942–15952, 2021. 2

[34] Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. History aware multimodal transformer for vision-and-language navigation. In *Proceedings of the International Conference on Neural Information Processing Systems*, pages 5834–5847, 2021. 2, 8

[35] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6741–6749, 2019. 2, 3, 7, 8

[36] James G March. Exploration and exploitation in organizational learning. *Organization science*, 2(1):71–87, 1991. 3

[37] Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O Stanley, and Jeff Clune. First return, then explore. *Nature*, 590(7847):580–586, 2021. 3

[38] Miruna Pislar, David Szepesvari, Georg Ostrovski, Diana L Borsa, and Tom Schaul. When should agents explore? In *Proceedings of the International Conference on Learning Representations*, 2022. 3

[39] Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. In *Proceedings of the International Conference on Learning Representations*, 2020. 3

[40] Meera Hahn, Devendra Singh Chaplot, Shubham Tulsiani, Mustafa Mukadam, James M Rehg, and Abhinav Gupta. No rl, no simulation: Learning to navigate without navigating. In *Proceedings of the International Conference on Neural Information Processing Systems*, volume 34, pages 26661–26673. Curran Associates, Inc., 2021. 3

[41] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 3

[42] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proceedings of the International Conference on Learning Representations*, 2021. 3

[43] Wolfgang Stürzl and Hanspeter A Mallot. Efficient visual homing based on fourier transformed panoramic images. *Robotics and Autonomous Systems*, 54(4):300–313, 2006. 3

[44] Jean Serra. *Mathematical Morphology*, pages 1–16. Springer International Publishing, Cham, 2020. 3

[45] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. In *Proceedings of the Conference on Empirical Methods for Natural Language Processing*, pages 5100–5111, 2019. 4

[46] Stéphane Ross, Geoffrey J Gordon, and J Andrew Bagnell. No-regret reductions for imitation learning and structured prediction. *arXiv preprint arXiv:1011.0686*, 2010. 6

[47] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *Proceedings of the International Conference on Machine Learning*, pages 1928–1937, 2016. 6

[48] Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, et al. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018. 7

[49] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6629–6638, 2019. 8

[50] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training home assistants to rearrange their habitat. In *Proceedings of the International Conference on Neural Information Processing Systems*, 2021. 8