Hierarchical Object-to-Zone Graph for Object Navigation

Sixian Zhang[1,2], Xinhang Song[1,2]*, Yubing Bai[1,2], Weijie Li[1,2], Yakui Chu[4], Shuqiang Jiang[1,2,3]
[1]Key Lab of Intelligent Information Processing Laboratory of the Chinese Academy of Sciences (CAS),
Institute of Computing Technology, Beijing [2]University of Chinese Academy of Sciences, Beijing
[3]Institute of Intelligent Computing Technology, Suzhou, CAS [4]Huawei Application Innovate Laboratory, Beijing
{sixian.zhang, xinhang.song, yubing.bai, weijie.li}@vipl.ict.ac.cn
chuyakui@huawei.com; sqjiang@ict.ac.cn

Abstract

The goal of object navigation is to reach the expected objects according to visual information in the unseen environments. Previous works usually implement deep models to train an agent to predict actions in real-time. However, in the unseen environment, when the target object is not in egocentric view, the agent may not be able to make wise decisions due to the lack of guidance. In this paper, we propose a hierarchical object-to-zone (HOZ) graph to guide the agent in a coarse-to-fine manner, and an online-learning mechanism is also proposed to update HOZ according to the real-time observation in new environments. In particular, the HOZ graph is composed of scene nodes, zone nodes and object nodes. With the pre-learned HOZ graph, the real-time observation and the target goal, the agent can constantly plan an optimal path from zone to zone. In the estimated path, the next potential zone is regarded as sub-goal, which is also fed into the deep reinforcement learning model for action prediction. Our methods are evaluated on the AI2-Thor simulator. In addition to widely used evaluation metrics SR and SPL, we also propose a new evaluation metric of SAE that focuses on the effective action rate. Experimental results demonstrate the effectiveness and efficiency of our proposed method. The code is available at https://github.com/sx-zhang/HOZ.git.

1. Introduction

Visual navigation task requires the agent to reach a specified goal. Conventional methods usually require spatial layout information, such as maps of the environments, which can be easily obtained in seen environments while unavailable in unseen environments. Therefore, how to efficiently navigate to the target in unseen environments is typically challenging.

*Corresponding author.

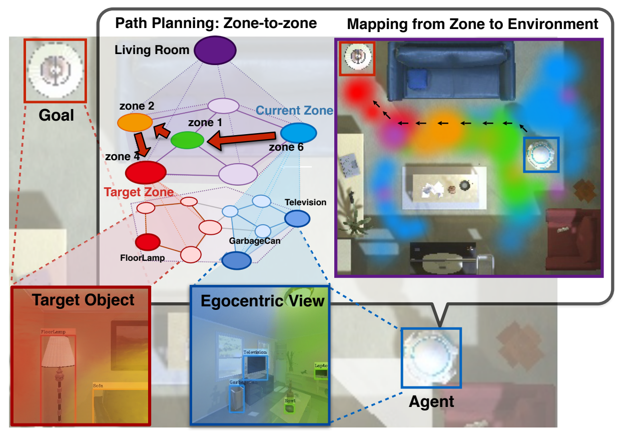

Figure 1. Overview of object navigation with HOZ graph. At the beginning, agent locates at the Current Zone ($zone_6$, blue) and the goal *FloorLamp* belongs to Target Zone ($zone_4$, red). The HOZ graph plans a real-time optimal path ($zone_6 - zone_1 - zone_2 - zone_4$). Then agent's next sub-goal is $zone_1$ (green). In the same way, the agent keeps updating sub-goal until it arrives at the target. Note that each color implies specific location and direction where agent can observe similar views.

With the visual input of egocentric observation, previous works [29, 27, 28] learn a deep reinforcement learning policy by maximizing the reward. The key challenge in those works is the generalization to unseen environments [38], especially when the target is not in the sight. Therefore, more recent works [40, 9] attempt to embed prior knowledge, such as object graph and relation graph, to improve the navigation model's generalization ability. In particular, Yang *et al.* [40] construct an object-to-object graph, which provides correlated objects as auxiliary information to locate the target object. Their object graph is too general to fit into specific environments. Additionally, Du *et al.* [9] propose to learn object relation graph, which fits the testing environments better than the general object graph. The above approaches focus on constructing object-oriented graph to provide some clues to the navigation when the target is not in the view. However, since object relations and spatial layout are usually inconsistent in different environments, the generalization ability of the above methods are still limited.

Motivated by enhancing the generalization ability of the navigation model, we carry out this study from two aspects: 1) learning an adaptive spatial knowledge representation that is applicable to various environments; 2) adapting the learned knowledge to guide navigation in the unseen environments. Besides, regions in larger area are considered in our knowledge, which are denoted as zones. Compared with objects, larger zones are more likely to be observed by agent. Thus our core idea for navigation guidance is zone.

In this paper we propose the hierarchical object-to-zone (HOZ) graph to capture the prior knowledge of scene layout for object navigation (see Figure 1). During training, we construct a general HOZ graph from all scenes, as rooms in the same scene category have same spatial structures. Each scene node corresponds to a scene-wise HOZ graph, whose zone nodes are obtained by matching and merging the room-wise HOZ graphs. For each room-wise HOZ graph, each zone node represents a group of relevant objects and each zone edge models the adjacent probability of two zones. Then we train a zone-to-action LSTM policy via deep reinforcement learning in the photo-realistic simulator AI2-Thor [19]. For each episode, the pre-learned HOZ graph helps to plan an optimal path from current zone to target zone, thus deducing the next potential zone on the path as a sub-goal. The sub-goal is embedded with graph convolutional network (GCN) to predict actions. Considering that different environments have diverse zone layouts, we also propose an online-learning mechanism to update the general learned HOZ graph according to current unseen environment. In this way, the initial HOZ graph will evolve towards current environment's specific layout and help agent to navigate successfully. Note that the update only holds for an episode and each episode starts from the initial HOZ graph. In addition to widely used evaluation metrics Success Rate (SR) and Success weighted by Path Length (SPL), we also propose a new evaluation metric of Success weighted by Action Efficiency (SAE) that considers the efficiency of the navigation action into SR. Our experiments show that the HOZ graph outperforms the baseline by a large margin. In summary, our contributions are as follows:

- We propose to learn the hierarchical object-to-zone (HOZ) graph that captures prior knowledge to guide object navigation agent with easier sub-goals.
- We propose a new evaluation metric named Success weighted by Action Efficiency (SAE).
- By integrating HOZ graph into a zone-to-action policy, the navigation performance can be significantly improved in SR, SPL and SAE metrics.

2. Related Work

Geometry-based navigation: Conventional navigation methods typically use a map as reference, whether it is constructed in advance or built simultaneously during visual navigation. [16, 3] utilize the metric-based map to perceive the environment and [10] keeps updating a probabilistic chessboard representation during agent's locomotion. Comparatively, [34, 5, 4] adopt coarse-grained topological map, with nodes showing semantic features and edges reasoning spatial relationships. [35, 36] both integrate metric-based map and topological map to improve mobile robot navigation. [23] constructs an experience graph to deal with long-term appearance changes. In addition, [12] adopts a belief map as spatial memory. Rather than relying on a specific map, our HOZ graph acts as prior knowledge to aid navigation in unseen environments.

Learning-based navigation: Deep learning has gained popularity in end-to-end localization, exploration and so on [12, 34]. As an early try, [25] takes neural networks to build a hallway follower model in indoor navigation. Nowadays, many researches turn to reinforcement learning (RL) to help agents make action decisions [33, 3, 15]. To improve generalization, [41, 40, 39] all employ Actor-Critic model [28]. Moreover, [6] learns exploration policies using an intrinsic coverage reward in imitation learning. [22] trains a task generator and a meta-learner to learn transferable meta-skills. [7] uses a generative model with probabilistic framework to benefit the similarity calculation of two observations. [34, 2] propose a waypoint navigation to find simpler sub-goals. [30] utilizes semantic information to boost deeper understanding. Meanwhile, [11] puts forward a memory-based policy. They embed each observation into a memory and perform this spatial-temporal memory on three visual navigation tasks. [26] proposes a reachability estimator that provides the navigator a sequence of target observations to follow. This line of works mostly treat the policy network as a black box and train it via RL, whereas our HOZ graph includes coarse-to-fine inputs of object, region, and scene, which allows for interpretable navigation.

Goal-driven navigation: This kind of navigation is carried out for subjective purposes, mainly conducted by natural language instructions or target images. It can be distinguished into PointGoal navigation [12, 3] and ObjectGoal navigation [27, 11, 38, 30, 40, 39]. In particular, sometimes the target may be presented as an image [4, 41]. Our work focuses on object navigation in unseen indoor environments. [38] proposes a self-adaptive visual navigation method to help agent learn to learn in an unseen environment via meta-reinforcement learning. [9] proposes an object representation graph to learn the spatial correlations among different object categories, and uses imitation learning to train the agent. A memory-augmented tentative policy network is used to detect deadlock condi-

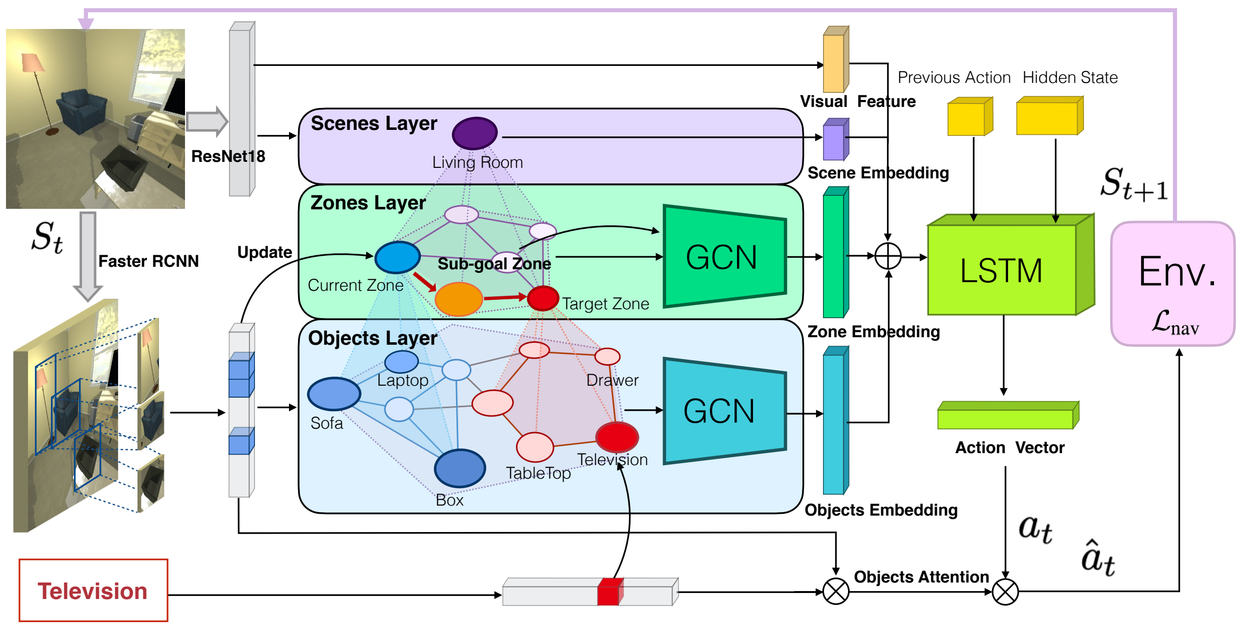

Figure 2. **Model Overview.** Our model is composed of the hierarchical object-to-zone (HOZ) graph and the zone-to-action LSTM. Given the target object and current observations, the agent first recognizes the scene category, locates the current zone, and deduces the next sub-goal zone according to the HOZ graph. The HOZ graph is updated at each timestamp based on the observations of the unseen environment. The zone-to-action LSTM learns to predict efficient actions based on the concatenated information provided by the HOZ graph.

tions and provides additional action guidance during testing. Recent works have applied knowledge graphs to image classification [24], segmentation [42], zero-shot recognition [37] and navigation [40, 39]. [39] proposes Bayesian Relational Memory that captures the room-to-room prior layout of environments during training to produce sub-goals for semantic-goal visual navigation. [40] establishes an object-to-object graph by extracting the relationships among object categories in Visual Genome [20]. While in our work, we conduct the online-learning hierarchical object-to-zone (HOZ) graph to serve as prior knowledge for object navigation, which provides more general regional information.

3. Preliminary Notation

Considering a set of environments Q and objects P, in each navigation episode, agent is initialized to a random location $l = \{x, z, \theta_{yaw}, \theta_{pitch}\}$ in an environment $q \in Q$. x, z represent the plane coordinate and $\theta_{yaw}, \theta_{pitch}$ represent the yaw and pitch angle (of the agent). At each timestamp t, agent learns a policy function $\pi\left(a_t|o_t, p\right)$, which predicts an action $a_t \in \mathcal{A}$ based on first-person view o_t and the target object $p \in P$. The discrete action space $\mathcal{A} = \{MoveAhead, RotateLeft, RotateRight, LookDown, LookUp, Done\}$. Note that the action *Done* is judged by the agent itself rather than informed by the environment. The success of object navigation task requires agent finally capturing and getting close to the target object (less than a threshold).

4. Hierarchical Object-to-Zone (HOZ) Graph

Our goal is to navigate agent to the given target without a precise map in the unseen environment. Thus, a great challenge in such task is to locate objects. Previous works [9, 38, 40] directly take the target object embedding as the goal to guide action prediction. However, it's typically difficult to plan an efficient path without prior knowledge about the unknown environment. The agent in those works might not find the path at the beginning, leading to some meaningless actions, such as frequently spinning around and backing. In order to provide stronger guidance, our navigation model considers a wider range region where the target object may be located, which is denoted as zone.

Each zone usually consists of a group of relevant objects. For instance, microwave, cooker and sink usually appear in the same zone. Thus, navigating to microwave may first require locating such zone. Since precise map information is not available in the unseen environment, how to collect suitable zones information and construct a hierarchical object-to-zone (HOZ) graph remains challenging. Therefore, we start from seen scenes to construct HOZ graph (Section 4.1) and later adaptively update it when navigating in the unseen scenes (Section 4.2).

We consider the zones from the following hierarchical structure. Our environments consist of several scenes, such as bedroom, living room, and kitchen, etc, and each scene contains several rooms. In each room $i \in \{1, 2, \ldots, n\}$, we get room-wise HOZ graph $\Omega_i\left(V_i, E_i\right)$, whose zone nodes

Algorithm 1 Scene-wise HOZ graph construction

Input: K: zone number
Input: $(Room_1, \ldots, Room_n)$ of same scene category
Create room-wise HOZ graphs set Ω
for $i \leftarrow 1$ to n **do**
  Get features and locations $[(f_1, l_1), \cdots, (f_d, l_d)]$ in $Room_i$ by agent with random exploring
  Create a graph $G_r(V_r, E_r)$
  $(C_1, \cdots, C_K) \leftarrow$ K-Means$(f_1, \cdots, f_d, K)$
  $V_r \leftarrow$ cluster centers $(C_1, \cdots, C_K)$
  $E_r \leftarrow$ calculate edges with Equation 1
  Add room-wise HOZ graph to $\Omega_i \leftarrow G_r(V_r, E_r)$
end for
Create scene-wise HOZ graph $G_s(V_s, E_s)$
Initialize $G_s(V_s, E_s) \leftarrow \Omega_1$
for $i \leftarrow 2$ to n **do**
  Create weighted bipartite graph $G^b(V^b, E^b)$
  $V^b \leftarrow V_s$ (all nodes of G_s), V_i (all nodes of Ω_i)
  $\omega(E^b) \leftarrow$ calculate similarity by Equation 2
  Perfect matching $\Psi^* \leftarrow$ Kuhn-Munkres($\omega(E^b)$)
  Update $G_s \leftarrow Avg(G_s, \Omega_i, \Psi^*)$ refer to Figure 3
end for
Output: scene-wise HOZ graph $G_s(V_s, E_s)$

are obtained by clustering the egocentric observation features and edges are defined as the adjacent probability of two zones (traced back to co-occurrence probability of each contained objects). Then we fuse these room-wise HOZ graphs grouped by scene to obtain scene-wise HOZ graphs $G_s(V_s, E_s)$. All scene-wise HOZ graphs have the same structure and constitute our final HOZ graph (Section 4.1).

4.1. HOZ Graph Construction

4.1.1 Room-wise HOZ graph

Similar scenes (e.g. "living room") may consist of common objects and object layouts [14, 43]. For instance, when referring to the living room, an area composed of sofa, pillow and table, or an area composed of TV set and TV cabinet may appear in our mind. When searching for an object, humans tend to first locate the typical area where the object most likely to appear. In our work, we denote such areas as zones and embed zones to guide agent. In order to obtain those representative zones, we sample visual features around the room and make a clustering on them.

In a specific room i, we first let the agent explore the room to collect a set of visual tuple features (f, l), where $f \in \mathbb{R}^{N\times 1}$ is a bag-of-objects vector obtained by Faster-RCNN [32], representing the objects that appear in the current view. It should be noticed that we use the bag-of-objects vector composed of 0 and 1 to represent the object category. If the current view contains many objects belonging to the same category, we only record them once. N denotes the number of object categories, and $l = \{x, z, \theta_{yaw}, \theta_{pitch}\}$ denotes the observation location defined in Section 3. Then we make K-Means clustering on features f to get K zones, forming the zone nodes in room-wise HOZ graph $\Omega_i(V_i, E_i)$. We use v_k and $\delta(v_k)$ to represent the k-th zone node and its embedded feature. The embedded feature represents the cluster center, which is calculated by $\delta(v_k) = \frac{1}{|zone_k|}\sum_{(f_\gamma, l_\gamma)\in zone_k} f_\gamma$, where $zone_k$ is a group of clustered visual tuple features (f, l) after K-Means, and $|zone_k|$ is the element number. Each dimension's value of $\delta(v_k)$ shows the connection relationship between the zones layer and objects layer (Figure 2), representing the co-occurrence frequency of objects belonging to the $zone_k$.

The edge $e(v_k, v_j)$ in the zones layer, represents the probability that two zones are adjacent to each other, which can be calculated as follows:

$$
\begin{aligned}
e(v_k, v_j) &= \frac{\sum_{(f_\gamma, l_\gamma)\in zone_k}\sum_{(f_\zeta, l_\zeta)\in zone_j}\eta(l_\gamma, l_\zeta)}{|zone_k|\times|zone_j|} \\
\eta(l_\gamma, l_\zeta) &= \begin{cases} 1 & |x_\gamma - x_\zeta| + |y_\gamma - y_\zeta| \le \varepsilon \\ 0 & otherwise \end{cases}
\end{aligned}
\tag{1}
$$

where ε is a hyper-parameter threshold. Then we use all node features to recognize the scene category. For each room i, we construct a room-wise HOZ graph $\Omega_i(V_i, E_i)$.

4.1.2 Scene-wise HOZ graph

To obtain scene-wise HOZ graph, we group all room-wise HOZ graphs by scene category. Take one scene as an example, we can obtain the room-wise set $\Omega = \{\Omega_1(V_1, E_1), \ldots, \Omega_n(V_n, E_n)\}$. Since the zones number K is fixed, each room-wise HOZ graph has the same structure for later matching and merging. Considering that directly computing the maximum matching of all room-wise HOZ nodes is expensive, we propose pairwise perfect matching and merging on two graphs each time until all graphs merge into the final one. The matching between $\Omega_i(V_i, E_i)$ and $\Omega_{i+1}(V_{i+1}, E_{i+1})$ graphs can be regarded as the weighted bipartite graph matching. We construct a bipartite graph $G^b = (V_i \cup V_{i+1}, E^b)$, where V_i is the nodes set in Ω_i, $|V_i| = |V_{i+1}|$, and E^b represents all fully connected edges. A perfect matching is to find a subset $\Psi \subseteq E^b$, where each node has exactly one edge incident on it. The maximum perfect matching satisfies $\Psi^* = \underset{\Psi}{argmax}\sum_{e^b\in\Psi}\omega(e^b)$, where $e^b \equiv e^b(v_k, v_j)$ represents the edge matching nodes v_k and v_j, $v_k \in V_i$, $v_j \in V_{i+1}$. The weight function $\omega(e^b)$ calculates the similarity of two nodes as

$$
\omega(e^b(v_k, v_j)) = 1/d(\delta_k, \delta_j) \tag{2}
$$

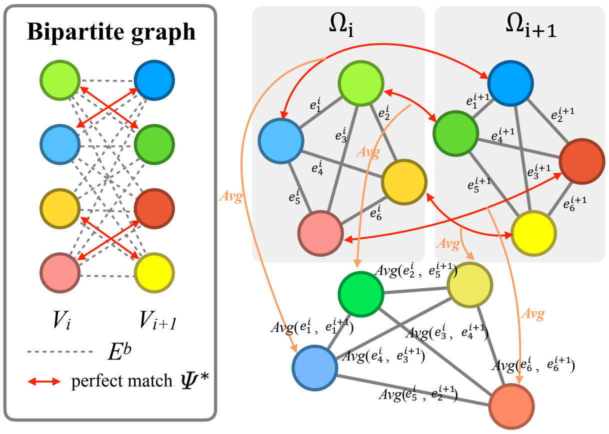

Figure 3. **Matching and Merging.** The left part shows the perfect maximum matching on weighted bipartite graph with Kuhn–Munkres algorithm. The right part shows the average calculation of merging two matched room-wise HOZ graphs into a new graph. For instance, two nodes (in red) are matched, and merged with average pooling (written as Avg). Correspondingly, edges between these nodes are merged with average pooling.

and $d(\delta_k, \delta_j)$ is defined as

$$d(\delta_k, \delta_j) = \sqrt{(\delta_k - \delta_j)^T (\delta_k - \delta_j)} + \frac{1}{\delta_k^T \delta_j + \alpha} \tag{3}$$

where $\delta_k \equiv \delta(v_k)$, $\delta_j \equiv \delta(v_j)$. α is a parameter to balance the two distances. We utilize the Kuhn–Munkres algorithm [21, 31] to solve this perfect maximum matching problem. Once getting the perfect matching, we averagely merge the matched nodes and edges as shown in Figure 3. The newly generated edge is the average of original edges between nodes involved by the new nodes. In this way, we can fuse room-wise HOZ graphs two-by-two each time and finally get the compositive graph, which is defined as scene-wise HOZ graph $G_s(V_s, E_s)$. Algorithm 1 summarizes the construction of scene-wise HOZ graph. All scene-wise HOZ graphs constitute our final HOZ graph.

4.2. HOZ Graph Updating and Embedding

4.2.1 Zone Updating and Embedding

With all training data, we can obtain a general HOZ graph $G(V, E)$ for the seen environments. Since different environments have various layouts, especially in the new unseen environment, it is difficult to construct a precise graph from scratch. Therefore, we first learn a general HOZ graph, and then propose an online-learning method to update current zone node according to agent's real-time view. In this way, the initial HOZ graph will evolve towards current environment. Note that the zone update only holds for an episode and each episode starts from the initial HOZ graph.

Through object detection, the agent obtains a bag-of-objects feature $f_t \in \mathbb{R}^{N\times 1}$ for object categories appearing in the egocentric view o_t at timestamp t. According to the visual feature f_t, target object $p \in P$ and HOZ graph $G(V, E)$, the agent calculates the current zone Z_c, target zone Z_t and sub-goal zone Z_{sub}, which will be detailed in Section 5.1. These zone indicator vectors $Z_c, Z_t, Z_{sub} \in \mathbb{R}^{K\times 1}$ are one-hot vectors that only activate representative zones. The proposed HOZ graph $G(V, E)$ is embedded with GCN. At time $t = 0$, the input matrix $\delta(V^0) \in \mathbb{R}^{K\times N}$ represents embedded features for all zone nodes V. Then $\delta(V^t)$ will be updated based on f_t, which can be formulated as

$$\delta(V^t) = \lambda Z_c f_t^T + (I - \lambda Z_c Z_c^T)\,\delta(V^{t-1}) \tag{4}$$

where λ is a learnable parameter that determines the current observation's impact on the general HOZ graph. Following [18], we perform normalization on edges E and obtain $\hat{E}$. With updated zone nodes $\delta(V^t)$, adjacent relationship $\hat{E}$, our GCN outputs a node-level representation $H_z \in \mathbb{R}^{K\times N}$ as the zones embedding

$$H_z = \sigma\left(\hat{E}\delta(V^t) W_z\right) \tag{5}$$

where $\sigma(\cdot)$ denotes the ReLU activation function, and $W_z \in \mathbb{R}^{N\times N}$ is the parameter of GCN layers. Then we take the encoded vector $H_z^T Z_{sub}$ as the output of zones layer, which informs agent about the next sub-goal zone and its relative position to other zones.

4.2.2 Object Embedding

Following [9], we set up objects layer with objects as nodes and relations between objects as edges, and encode them with GCN. For current egocentric view, we can get the detection feature $F_t = \{f_t^b, f_t^s, f_t^v\}$, where $f_t^b \in \mathbb{R}^{\mathrm{N}\times 4}$ is bounding box position, $f_t^s \in \mathbb{R}^{N\times 1}$ is confidence score and $f_t^v \in \mathbb{R}^{\mathrm{N}\times 512}$ is the visual feature of objects. If multiple instances belonging to the same category appear simultaneously, the one with the highest confidence score provided by the detector will be selected. Define $X_o = [f_t^b, f_t^s, p] \in \mathbb{R}^{\mathrm{N}\times 6}$ as the input of GCN, where $p \in \mathbb{R}^{N\times 1}$ is a one-hot vector representing the target object. The GCN outputs

$$H_o = \sigma(AX_oW_o) \tag{6}$$

Both the adjacency matrix A and the GCN network parameter $W_o \in \mathbb{R}^{6\times N}$ need to be learned. Then we integrate $H_o f_t^v$ as the objects embedding, which provides object-level information.

5. Navigation Policy

5.1. Zone Localization and Navigation Planning

Current zone We compare current view bag-of-objects vector f_t with the nodes in the pre-learned HOZ graph

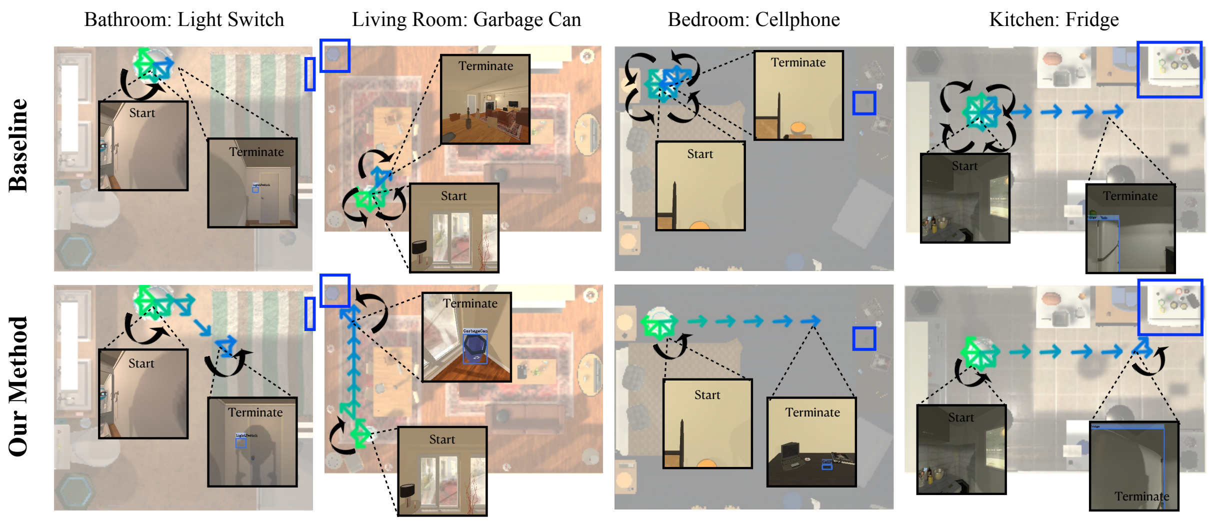

Figure 4. **Visualization in testing environments.** Black arrows represent rotations. The trajectory of the agent is illustrated with green and blue arrows, where green is the beginning and blue is the end.

$G(V, E)$, and take the most similar node as the current zone, which can be formulated as

$$Z_c = \chi^K\left(\underset{k}{argmin}\left(d\left(f_t, \delta\left(v_k\right)\right)\right)\right), v_k \in V \qquad (7)$$

where $\chi^K(\cdot)$ is an indicator that produces a one-hot vector $\chi^K(i) = [x_1, \ldots x_K]^T$, where $x_i = 1, x_{j\neq i} = 0$. $d(\cdot)$ is defined in Equation 3. Then the HOZ graph is updated by the current zone Z_c and the real-time feature f_t (Equation 4).

Target zone We take the node with the highest occurrence probability of the target object as the target zone.

$$Z_t = \chi^K\left(\underset{k}{argmax}\left(\delta\left(v_k\right)^T p\right)\right), v_k \in V \qquad (8)$$

Sub-goal zone To navigate agent from current zone to target zone, we search for a path with the maximum connection probability. If an edge has a higher value, the two related zones are more likely to be adjacent so that agent can easily arrive. Besides, when the target zone is far away from the current zone or is not visible in the current view, the agent may not be well guided. Therefore, we take the second child zone starting from the current zone on this path as the sub-goal zone, which provides information about where to go next. Our goal is to find an optimal maximum connectivity path $\Gamma = \{v_{\tau_0}, v_{\tau_1}, \ldots, v_{\tau_T}\}$, where $\tau_i \in \{1, \ldots, K\}$ denotes the node index and v_{τ_0} represents the current zone and v_{τ_T} represents the target zone, so that the connection probability along the path is maximized as:

$$\Gamma^* = \underset{\Gamma}{argmax}\, \Pi_{i=1}^{T} e\left(v_{\tau_{i-1}}, v_{\tau_i}\right) \qquad (9)$$

After obtaining Γ^*, we can get the sub-goal zone $Z_{sub} = \chi^K(\tau_1^*)$. Whenever the current zone changes, the network will adaptively replan an optimal path and a sub-goal zone.

5.2. Policy Learning

Action policy The conventional works [38, 9, 40, 41] learn a policy $\pi(a_t|o_t, p)$ based on current observation. While in our work, we learn a zone-to-action LSTM action policy $\pi_z(a_t|S_t, p)$, where S_t is the joint representation of current observation o_t, the sub-goal zone embedding $H_z^T Z_{sub}$ and object embedding $H_o f_t^v$. Following [41, 27] formulating this task as a reinforcement learning problem, we optimize the LSTM via the Asynchronous Advantage Actor-Critic (A3C) algorithm [28] that learns policy function and value function by minimizing navigation loss $\mathcal{L}_{nav}$ to maximize the reward. The policy function outputs a_t, representing actions probability at each time, and the value function is used to train the policy network.

Done reminder To remind agent to stop in time when it encounters the target object, we propose the done reminder. Combining objects detection confidence f_t^s and the target object p, we weight a_t with $\beta p^T f_t^s$ to represent the effect of *done* action (β is a learnable parameter). In this way, we can get the final action output $\hat{a}_t$.

6. Experiments

6.1. Experiment Setup

We evaluate our methods on AI2-Thor simulator [19], which provides near photo-realistic observation in 3D indoor scenes. AI2-Thor contains a total of 120 scenes in 4 types: living room, kitchen, bedroom, and bathroom, where spatial layout, object types and appearance are all different.

Table 1. **Comparisons with sub-goal zone and target zone (%).** The input of zone-to-action LSTM during training and testing is set to the sub-goal zone (S) or target zone (T) respectively.

Method	Training		Testing		ALL			$L \geq 5$		
	S	T	S	T	SR	SPL	SAE	SR	SPL	SAE
Baseline					$57.35_{\pm 1.92}$	$33.78_{\pm 1.33}$	$19.02_{\pm 1.36}$	$45.77_{\pm 2.17}$	$30.65_{\pm 1.01}$	$20.04_{\pm 1.87}$
HOZ	√		√		**70.57**$_{\pm 1.11}$	**40.84**$_{\pm 1.12}$	**27.19**$_{\pm 1.96}$	**61.52**$_{\pm 1.47}$	**40.46**$_{\pm 0.63}$	**29.61**$_{\pm 1.08}$
HOZ	√			√	$69.04_{\pm 1.07}$	$40.07_{\pm 1.04}$	$26.19_{\pm 0.95}$	$59.27_{\pm 1.33}$	$39.12_{\pm 0.83}$	$28.34_{\pm 1.24}$
HOZ		√		√	$69.16_{\pm 1.15}$	$39.05_{\pm 0.88}$	$26.04_{\pm 0.91}$	$60.28_{\pm 1.42}$	$38.61_{\pm 0.86}$	$29.08_{\pm 0.98}$

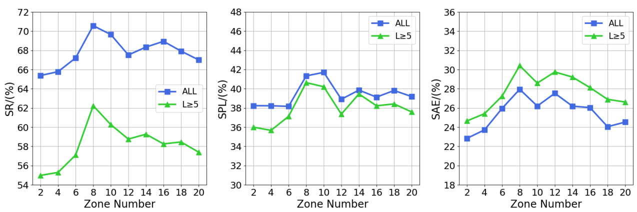

Figure 5. **Ablation results of zone number.** We evaluate the impact of the zone number (cluster number) on navigation metrics such as SR, SPL, and SAE.

Following the setting in [38], a subset of 22 types of objects is considered, ensuring that each scene contains at least four objects. For each scene type, we choose 20 rooms for training, 5 for validation, and 5 for test.

6.2. Implementation Details

The baseline is the A3C [28] navigation policy with a simple visual embedding layer to encode inputs. We train our models with 12 asynchronous workers, in a total of 6M navigation episodes. In policy learning, the agent receives a -0.01 penalty for each step and a reward of 5 if the episode is successful. We use Adam optimizer [17] to update our network parameters with a learning rate of 10^{-4}. ResNet18 [13] pretrained on ImageNet [8] is used as our backbone to extract the features of each egocentric view. In the HOZ graph construction, we finetune Faster-RCNN [32] architecture on 50% training data of AI2-Thor. The hyper-parameters in our model are initialized to $\varepsilon = 0.25$, $\alpha = 0.1$ and $\beta = 0.6$.

For evaluation, we randomly select agent's initial starting position and the target object, and repeatedly run 5 trials. We report results (with average and variance) for all targets (All) and a subset of targets ($L \geq 5$) whose optimal trajectory length is longer than 5.

6.3. Evaluation Metrics

We use Success Rate (SR), Success Weighted by Path Length (SPL) [1], and Success Weighted by Action efficiency (SAE) metrics to evaluate our model. SR refers to the success rate of agent in finding the target object, which is formulated as $SR = \frac{1}{N}\sum_{n=1}^{N} Suc_n$, where N is the total number of episodes and Suc_n is an indicator function to indicate whether the n-th episode succeeds. SPL considers both the success rate and the path length. It is defined as $SPL = \frac{1}{N}\sum_{i=1}^{N} Suc_i \frac{L_i^*}{max(L_i, L_i^*)}$, where L_i is the actual path length and L_i^* represents the shortest path provided by the simulator. Although SPL calculates the proximity between the path and the optimal path, it ignores the efficiency of action sequence. For instance, unnecessary rotations take time and reduce efficiency, which are not considered in SPL. Therefore, we propose the SAE metric to measure the efficiency of all actions. It is formulated as $SAE = \frac{1}{N}\sum_{i=1}^{N} Suc_i \frac{\sum_{t=0}^{T} \mathbb{I}(a_t^i \in \mathcal{A}_{change})}{\sum_{t=0}^{T} \mathbb{I}(a_t^i \in \mathcal{A}_{all})}$, where $\mathbb{I}(\cdot)$ is the indicator function, a_t^i is agent's action at time t in episode i, $\mathcal{A}_{all}$ is the set of all action categories and $\mathcal{A}_{change}$ refers to those actions that can change agent's location. In our settings $\mathcal{A}_{change} = \{MoveAhead\}$.

6.4. Ablation Study

Effectiveness of sub-goal zones As discussed in Section 5.1, besides the target zone, we also consider the sub-goal zone. The ablation study respectively trains the policy network with the sub-goal zone and the target zone as illustrated in Table 1 *line2* and *line4*. Compared to the target zone, sub-goal zone can better guide agent efficiently. Training with the embedding of sub-goal zone outperforms target zone by 1.41/1.24, 1.79/1.85 and 1.15/0.53 in SR, SPL and SAE (ALL/$L \geq 5$, %) respectively.

Impacts of the number of zones The cluster number is a hyper-parameter that specifies the zone number in a scene. Figure 5 indicates that performance is reduced when the number of zones is either too large or too small. Besides, a large zone number requires significant computing resources when planning the path. The results suggest that the optimal number of zones is 8. Therefore, the number of zones is set to 8 in the remaining evaluations.

Other ablation studies We dissect the proposed HOZ graph into different components. The ablation study in Table 2 demonstrates the efficacy of each component of our method. Specifically, it is observed that the object layer significantly improves the baseline performance. Additionally, scene and zone layers can considerably increase the performance on SPL and SAE metrics. Although the done

Table 2. **The ablation study of different components (%).** We evaluate the effect of various modules. These modules include the scene layer (Scene), the zone layer (Zone), the object layer (Object) in Section 4.2 and the done reminder (Reminder) in Section 5.2.

Baseline	Scene	Zone	Object	Reminder	All			$L \geq 5$		
					SR	SPL	SAE	SR	SPL	SAE
√					$57.35_{\pm 1.92}$	$33.78_{\pm 1.33}$	$19.02_{\pm 1.36}$	$45.77_{\pm 2.17}$	$30.65_{\pm 1.01}$	$20.04_{\pm 1.87}$
√			√		$65.12_{\pm 1.03}$	$37.86_{\pm 0.93}$	$24.36_{\pm 0.91}$	$53.42_{\pm 1.43}$	$35.37_{\pm 0.71}$	$25.32_{\pm 1.04}$
√	√		√		$65.81_{\pm 1.11}$	$38.83_{\pm 0.59}$	$22.45_{\pm 0.99}$	$57.23_{\pm 0.93}$	$36.25_{\pm 0.65}$	$25.53_{\pm 0.87}$
√			√	√	$66.73_{\pm 1.01}$	$37.82_{\pm 0.83}$	$24.81_{\pm 0.84}$	$57.55_{\pm 1.19}$	$36.48_{\pm 0.52}$	$27.79_{\pm 1.07}$
√	√	√	√		$70.57_{\pm 1.11}$	$\mathbf{40.84}_{\pm 1.12}$	$27.19_{\pm 1.96}$	$61.52_{\pm 1.47}$	$\mathbf{40.46}_{\pm 0.63}$	$29.61_{\pm 1.08}$
√	√	√	√	√	$\mathbf{70.62}_{\pm 1.70}$	$40.02_{\pm 1.25}$	$\mathbf{27.97}_{\pm 2.01}$	$\mathbf{62.75}_{\pm 1.73}$	$39.24_{\pm 0.56}$	$\mathbf{30.14}_{\pm 1.34}$

Table 3. **Comparisons with the related works (%).** Constrained by space, variance is detailed in supplementary materials.

Method	All			$L \geq 5$		
	SR	SPL	SAE	SR	SPL	SAE
Non-adaptive method						
Random	3.56	1.73	0.41	0.27	0.07	0.06
A3C (baseline)	57.35	33.78	19.02	45.77	30.65	20.04
SP [40]	62.16	37.01	23.39	50.86	34.17	24.35
ORG [9]	66.38	38.42	25.36	55.55	36.26	27.53
Ours (HOZ)	**70.62**	**40.02**	**27.97**	**62.75**	**39.24**	**30.14**
Self-supervised method						
SAVN [38]	63.32	37.62	21.97	52.38	35.31	24.64
ORG-TPN [9]	67.31	**39.53**	23.07	57.41	38.27	26.37
Ours (HOZ-TPN)	**73.15**	39.22	**29.49**	**64.58**	**39.80**	**30.92**

reminder decreases the SPL metric, it increases the SR and SAE metrics, indicating that adding the done reminder lengthens the episodes. Overall, our method outperforms the baseline model with the gains of 13.27/16.98, 6.24/8.59 and 8.95/10.10 in SR, SPL and SAE (ALL/$L \geq 5$, %). The experimental results indicate that our method is capable of effectively guiding navigation in the unseen environments.

In addition, considering that the construction of the scene-wise HOZ graph may be inconsistent due to different merging order of room-wise HOZ graph. We test 20 different merging orders to get the variance of 0.83/0.81, 0.78/0.81, 0.81/0.82 in SR, SPL and SAE (ALL/$L \geq 5$, %). These results indicate that the merging-related potential inconsistency has little effect on the navigation performance.

6.5. Comparisons to the State-of-the-art

Related works can be categorized into non-adaptive models [9, 40] and self-supervised models [38, 9]. Compared with the non-adaptive methods in Table 3, our method outperforms the state-of-the-art by a large margin in SR, SPL and SAE metrics. Particularly, we obtain the gains of 4.24/7.20, 1.60/2.98, 2.61/2.61 in SR, SPL and SAE (ALL/$L \geq 5$, %) over the state-of-the-art model [9].

Compared to the non-adaptive models, the self-supervised models are updated with self-supervision in test. This self-supervision can somehow improve performance, but also consume additional computing resources. We also implement our methods with self-supervision (denoted as HOZ-TPN). In comparison to HOZ, HOZ-TPN improves SR but achieves equivalent results in SPL and SAE, which are more indicative of navigation efficiency. The comparison between HOZ and HOZ-TPN (as well as ORG and ORG-TPN) demonstrates that while self-supervision may aid in successfully navigating to target objects, it also introduces additional actions. More experimental results are detailed in supplementary materials.

Case study Figure 4 qualitatively compares our HOZ with the baseline model. In these scenarios, the agent is placed at an initial position where the target object cannot be seen. The baseline model often falls into rotations when the target object is not in the view. However, our HOZ method helps the agent locate the current zone and offers guidance from the current zone to the target zone, thus the agent has better performance. Notably, with the guidance of sub-goal zone , the agent equipped with our HOZ graph can choose a better rotation direction than the baseline method.

7. Conclusions

We propose the hierarchical object-to-zone (HOZ) graph that captures the prior knowledge of objects in typical zones. The agent equipped with HOZ is capable of updating prior knowledge, locating the target zone and planning the zone-to-zone path. We also propose a new evaluation metric named Success weighted by Action Efficiency (SAE) that measures the efficiency of actions. Experimental results show that our approach outperforms baseline by a large margin in SR, SPL and SAE metrics.

Acknowledgements: This work was supported by National Key Research and Development Project of New Generation Artificial Intelligence of China, under Grant 2018AAA0102500, in part by the National Natural Science Foundation of China under Grant 62032022, 61902378 and U1936203, in part by Beijing Natural Science Foundation under Grant L182054 and Z190020, in part by the Lenovo Outstanding Young Scientists Program, in part by the National Postdoctoral Program for Innovative Talents under Grant BX201700255.

References

[1] 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. 6.3

[2] Somil Bansal, Varun Tolani, Saurabh Gupta, Jitendra Malik, and Claire Tomlin. Combining optimal control and learning for visual navigation in novel environments. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura, editors, *3rd Annual Conference on Robot Learning, CoRL 2019, Osaka, Japan, October 30 - November 1, 2019, Proceedings*, volume 100 of *Proceedings of Machine Learning Research*, pages 420–429. PMLR, 2019. 2

[3] Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural SLAM. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. 2

[4] Devendra Singh Chaplot, Ruslan Salakhutdinov, Abhinav Gupta, and Saurabh Gupta. Neural topological SLAM for visual navigation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 12872–12881. IEEE, 2020. 2

[5] Kevin Chen, Juan Pablo de Vicente, Gabriel Sepulveda, Fei Xia, Alvaro Soto, Marynel Vázquez, and Silvio Savarese. A behavioral approach to visual navigation with graph localization networks. In Antonio Bicchi, Hadas Kress-Gazit, and Seth Hutchinson, editors, *Robotics: Science and Systems XV, University of Freiburg, Freiburg im Breisgau, Germany, June 22-26, 2019*, 2019. 2

[6] Tao Chen, Saurabh Gupta, and Abhinav Gupta. Learning exploration policies for navigation. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. 2

[7] Mark Cummins and Paul M. Newman. Probabilistic appearance based navigation and loop closing. In *2007 IEEE International Conference on Robotics and Automation, ICRA 2007, 10-14 April 2007, Roma, Italy*, pages 2042–2048. IEEE, 2007. 2

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 248–255, 2009. 6.2

[9] Heming Du, Xin Yu, and Liang Zheng. Learning object relation graph and tentative policy for visual navigation. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part VII*, pages 19–34, 2020. 1, 2, 4, 4.2.2, 5.2, 3, 6.5

[10] Alberto Elfes. Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57, 1989. 2

[11] Kuan Fang, Alexander Toshev, Fei-Fei Li, and Silvio Savarese. Scene memory transformer for embodied agents in long-horizon tasks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 538–547. Computer Vision Foundation / IEEE, 2019. 2

[12] Saurabh Gupta, James Davidson, Sergey Levine, Rahul Sukthankar, and Jitendra Malik. Cognitive mapping and planning for visual navigation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 7272–7281. IEEE Computer Society, 2017. 2

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778, 2016. 6.2

[14] Hamid Izadinia, Fereshteh Sadeghi, and Ali Farhadi. Incorporating scene context and object layout into appearance modeling. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pages 232–239, 2014. 4.1.1

[15] Gregory Kahn, Adam Villaflor, Bosen Ding, Pieter Abbeel, and Sergey Levine. Self-supervised deep reinforcement learning with generalized computation graphs for robot navigation. In *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*, pages 1–8. IEEE, 2018. 2

[16] Kiyosumi Kidono, Jun Miura, and Yoshiaki Shirai. Autonomous visual navigation of a mobile robot using a human-guided experience. *Robotics Auton. Syst.*, 40(2-3):121–130, 2002. 2

[17] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 6.2

[18] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017. 4.2.1

[19] Eric Kolve, Roozbeh Mottaghi, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. AI2-THOR: an interactive 3d environment for visual AI. *CoRR*, abs/1712.05474, 2017. 1, 6.1

[20] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *Int. J. Comput. Vis.*, 123(1):32–73, 2017. 2

[21] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955. 4.1.2

[22] Juncheng Li, Xin Wang, Siliang Tang, Haizhou Shi, Fei Wu, Yueting Zhuang, and William Yang Wang. Unsupervised reinforcement learning of transferable meta-skills for embodied navigation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 12120–12129. IEEE, 2020. 2

[23] Chris Linegar, Winston Churchill, and Paul Newman. Work smart, not hard: Recalling relevant experiences for vast-scale but time-constrained localisation. In *IEEE International Conference on Robotics and Automation, ICRA 2015, Seattle, WA, USA, 26-30 May, 2015*, pages 90–97, 2015. 2

[24] Kenneth Marino, Ruslan Salakhutdinov, and Abhinav Gupta. The more you know: Using knowledge graphs for image classification. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 20–28, 2017. 2

[25] Min Meng and Avinash C Kak. Mobile robot navigation using neural networks and nonmetrical environmental models. *IEEE Control Systems Magazine*, 13(5):30–39, 1993. 2

[26] Xiangyun Meng, Nathan D. Ratliff, Yu Xiang, and Dieter Fox. Scaling local control to large-scale topological navigation. In *2020 IEEE International Conference on Robotics and Automation, ICRA 2020, Paris, France, May 31 - August 31, 2020*, pages 672–678, 2020. 2

[27] Piotr Mirowski, Razvan Pascanu, Fabio Viola, Hubert Soyer, Andy Ballard, Andrea Banino, Misha Denil, Ross Goroshin, Laurent Sifre, Koray Kavukcuoglu, Dharshan Kumaran, and Raia Hadsell. Learning to navigate in complex environments. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017. 1, 2, 5.2

[28] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, pages 1928–1937, 2016. 1, 2, 5.2, 6.2

[29] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015. 1

[30] Arsalan Mousavian, Alexander Toshev, Marek Fiser, Jana Kosecká, Ayzaan Wahid, and James Davidson. Visual representations for semantic target driven navigation. In *International Conference on Robotics and Automation, ICRA 2019, Montreal, QC, Canada, May 20-24, 2019*, pages 8846–8852. IEEE, 2019. 2

[31] James Munkres. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38, 1957. 4.1.2

[32] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(6):1137–1149, 2017. 4.1.1, 6.2

[33] Fereshteh Sadeghi and Sergey Levine. CAD2RL: real single-image flight without a single real image. In Nancy M. Amato, Siddhartha S. Srinivasa, Nora Ayanian, and Scott Kuindersma, editors, *Robotics: Science and Systems XIII, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, July 12-16, 2017*, 2017. 2

[34] Nikolay Savinov, Alexey Dosovitskiy, and Vladlen Koltun. Semi-parametric topological memory for navigation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. 2

[35] Sebastian Thrun. Learning metric-topological maps for indoor mobile robot navigation. *Artif. Intell.*, 99(1):21–71, 1998. 2

[36] Sebastian Thrun, Jens-Steffen Gutmann, Dieter Fox, Wolfram Burgard, and Benjamin Kuipers. Integrating topological and metric maps for mobile robot navigation: A statistical approach. In Jack Mostow and Chuck Rich, editors, *Proceedings of the Fifteenth National Conference on Artificial Intelligence and Tenth Innovative Applications of Artificial Intelligence Conference, AAAI 98, IAAI 98, July 26-30, 1998, Madison, Wisconsin, USA*, pages 989–995. AAAI Press / The MIT Press, 1998. 2

[37] Xiaolong Wang, Yufei Ye, and Abhinav Gupta. Zero-shot recognition via semantic embeddings and knowledge graphs. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 6857–6866, 2018. 2

[38] Mitchell Wortsman, Kiana Ehsani, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. Learning to learn how to learn: Self-adaptive visual navigation using meta-learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 6750–6759, 2019. 1, 2, 4, 5.2, 6.1, 3, 6.5

[39] Yi Wu, Yuxin Wu, Aviv Tamar, Stuart J. Russell, Georgia Gkioxari, and Yuandong Tian. Bayesian relational memory for semantic visual navigation. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 2769–2779, 2019. 2

[40] Wei Yang, Xiaolong Wang, Ali Farhadi, Abhinav Gupta, and Roozbeh Mottaghi. Visual semantic navigation using scene priors. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019. 1, 2, 4, 5.2, 3, 6.5

[41] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J. Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE International Conference on Robotics and Automation, ICRA 2017, Singapore, Singapore, May 29 - June 3, 2017*, pages 3357–3364, 2017. 2, 5.2

[42] Yukun Zhu, Raquel Urtasun, Ruslan Salakhutdinov, and Sanja Fidler. segdeepm: Exploiting segmentation and context in deep neural networks for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 4703–4711, 2015. 2

[43] Zhen Zuo, Bing Shuai, Gang Wang, Xiao Liu, Xingxing Wang, Bing Wang, and Yushi Chen. Learning contextual dependence with convolutional hierarchical recurrent neural

networks. *IEEE Trans. Image Process.*, 25(7):2983–2996, 2016. 4.1.1