Commonsense-Aware Object Value Graph for Object Goal Navigation

Hwiyeon Yoo[1], Yunho Choi[1], Jeongho Park[1] and Songhwai Oh[1]

***Abstract*— Object goal navigation (ObjectNav) is the task of finding a target object in an unseen environment. It is one of the fundamental challenges in visual navigation as it requires both structural and semantic understanding. In this paper, we present OVG-Nav, a novel ObjectNav framework that leverages a topological graph structure called object value graph (OVG), which contains visual observations and commonsense prior knowledge. The high-level planning of OVG-Nav prioritizes subgoal nodes for exploration based on a metric called *object value*, which reflects the closeness to the target object. Here, we propose OVGNet, a model designed to predict the object values of each node of an OVG using observed features along with commonsense knowledge. The structure of high-level planning using OVG and low-level action decisions reduces sensitivity to accumulating sensor noises, leading to robust navigation performance. Experimental results show that OVG-Nav outperforms the baseline in success rate (SR) and success rate weighted by path length (SPL) in the MP3D dataset both in accurate sensing and noisy sensing. In addition, we show that the OVG-Nav can be transferred to the real-world robot successfully.**

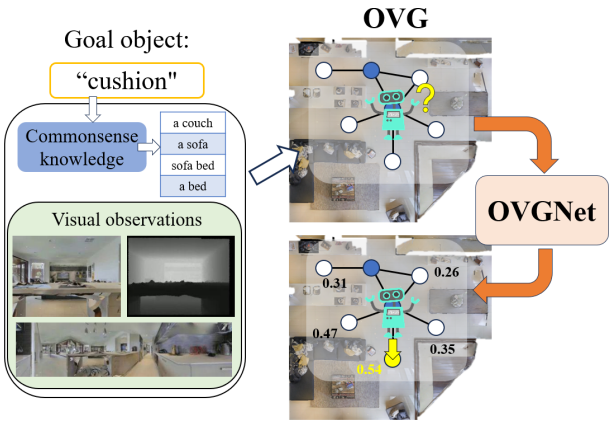


Fig. 1. Overview of OVG-Nav. The agent constructs an object value graph (OVG) with commonsense knowledge, visual observations, and the goal object information. The proposed OVGNet predicts the object value of each candidate node. Then, the agent navigates to the node with the highest object value, predicted as the closest node to the target object. The figure shows an example with a goal object, *cushion*. Here, OVGNet selects the node near a sofa that fits the commonsense knowledge related to the cushion.

I. Introduction

Object goal navigation (ObjectNav) is a fundamental problem in the field of embodied intelligent robotics. The ObjectNav task challenges an agent to navigate through an unknown environment to locate an object instance of a specific category. While navigation in an unknown environment is already a challenging task requiring a spatial understanding of the environment, ObjectNav further complicates this by requiring semantic understanding, such as relations between object categories and between goal objects and scene categories.

Most recent research on ObjectNav has been categorized into two types of strategies. The first popular approach is utilizing explicit metric maps, such as top-down-view occupancy grid maps [1], [2], [3], [4], [5]. Agents using this strategy construct a metric map during navigation and subsequently navigate based on this map. The constructed metric maps are often called semantic maps when combined with semantic information, such as detected objects or encoded visual features [1], [2]. This approach is advantageous as it provides explicit and comprehensive spatial representations, integrating both the environmental structure and pertinent semantic observations.

Another common approach is using end-to-end neural network models to derive action decisions directly from agent observations such as RGB-D observations and agent pose [6], [7], [8], [9], [10]. These methods typically integrate deep visual features from encoders such as ResNet [11] and CLIP visual encoder [12], and other features with encoding networks like RNNs to output the following step action. These end-to-end action policy networks are usually trained using reinforcement learning (RL) [6], [7], [8], [10] or using observation-action paired data [9]. With the power of deep neural networks, these methods can implicitly encode high-level environmental features for navigation, including semantic information.

While both methodologies offer distinct advantages in ObjectNav, they also come with inherent limitations under certain scenarios. For instance, despite its explicit and rich spatial context, the semantic map requires precise agent poses for the metric map construction. Relying on these maps for global navigation and action planning becomes problematic if any pose errors corrupt the map. This becomes especially concerning when agents lack access to precise GPS and compass sensors, such as indoor navigation, leading to noise accumulation through repeated pose estimation. For end-to-end methodologies, implicit feature embedding is disadvantageous in understanding global structural information due

[1]The authors are with the Department of Electrical and Computer Engineering and ASRI, Seoul National University, Seoul, Korea (e-mail: hwiyeon.yoo, yunho.choi, jeongho.park@rllab.snu.ac.kr; songhwai@snu.ac.kr)

This work was 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, 75%) and Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (NRF-2022R1A2C2008239, General-Purpose Deep Reinforcement Learning Using Metaverse for Real World Applications, 25%).

to the integrated embedding features with local observation. This limits their efficiency, especially if the target object is significantly distant from the agent's starting point.

To address these issues, we introduce the object value graph navigation for ObjectNav (OVG-Nav). This framework avoids the limitations of relying heavily on precise metric data or fully implicit feature embedding. Instead, OVG-Nav employs a topological graph structure, the object value graph (OVG), which serves as an episodic memory representation. The OVG captures spatial relationships in the environment, enriched by semantic visual features and commonsense prior knowledge related to the target object, for instance, which kind of space is probable to find the goal object. For leveraging the combined structural and semantic attributes of the OVG, we propose OVGNet that predicts *object values* for each node of an OVG. These object values reflect the closeness to their nearest target object, serving as a metric to estimate which node is physically close to the goal object. Consequently, a node with the highest object value is designated as a subgoal node in high-level navigation, obtaining priority in exploration for efficient object searching. Fig. 1 shows an overview of OVG-Nav.

One notable advantage of OVG-Nav's topological graph-centric approach is its diminished dependence on precise geometrical sensing, making it more robust to pose estimation inaccuracies. The topological representation also emphasizes connections between diverse locations, allowing the agent to make more effective navigation decisions based on enriched spatial-aware semantic contexts. Furthermore, by incorporating commonsense priors from (Comet-) Atomic 2020 [13], OVG-Nav gains insights into locations associated with the target object. For example, according to commonsense knowledge, it is more probable to find a target object bed in a $room$ rather than in a $kitchen$. Thus, an exploration policy that prioritizes observed $room$-like areas over $kitchen$-like areas is likely to find a bed more quickly. By leveraging this type of commonsense prior knowledge, OVG-Nav enhances exploration efficiency for finding a specific target object.

We have performed rigorous experiments on the Matterport3D (MP3D) [14] ObjectNav dataset to evaluate OVG-Nav, using the AI Habitat simulator [15]. Our evaluations spanned scenarios from using precise oracle agent pose to sensory noise on actuation and visual sensing. Impressively, OVG-Nav consistently outperformed the existing methods, particularly in noise-intense settings. Beyond simulation, we have validated the practical applicability of OVG-Nav through real-world robotic experiments.

In summary, the contributions of the paper are as follows:

- We introduce OVG-Nav, a novel ObjectNav framework that merges visual observations and commonsense priors within a topological graph structure, providing a well-balanced and robust solution.
- Our experiments show that OVG-Nav exceeds the performance of the existing methods in both ideal and noisy conditions, confirming its robustness and efficiency.
- We successfully transferred the proposed method to a real-world mobile robot, achieving satisfactory ObjectNav results that demonstrate the real-world applicability of our method.

II. Related Work

A. Object Goal Navigation

Various studies have been conducted to address the ObjectNav task. Methods using semantic maps have been proposed in [16] and [2], where each cell in the map contains the detected object label and occupancy information. These semantic maps serve as input features for a high-level policy network to estimate the next location to explore and are also used for local navigation toward the current goal point. While these methods have demonstrated successful performance in ObjectNav, reliance on semantic maps can be problematic due to potential distortions caused by inaccurate pose estimation and accumulated pose errors.

The other approaches that use end-to-end action policies without building explicit map representations have been explored, often by proposing auxiliary tasks [6] and data augmentations [7]. Although these methods show promising results, their lack of global structural information can lead to inefficient actions. In this paper, we propose OVG-Nav, a topological graph representation method that leverages the spatial structure of the environment while being less affected by positional errors.

Research has also been conducted on utilizing prior knowledge of relationships between the goal object and observed objects or rooms [17], [10], [18], [19]. Studies such as [17] and [18] employ object relation graphs to efficiently locate the goal object based on the currently observed objects. [10] collects prior knowledge about hierarchical relationships between objects and spatial zones and utilizes this information for goal object searching. [19] uses commonsense prior knowledge of the goal object to identify objects with high co-occurrence scores. These methods demonstrate that using prior knowledge can enhance ObjectNav performance; however, they are typically limited to short-range episodes involving one or two rooms where related objects for hints can be easily found, e.g., AI2Thor environments [20]. In contrast, our paper focuses on utilizing prior knowledge for goal object searching in longer-range episodes that span an entire house, such as environments in the MP3D dataset.

B. Topological Representation for Visual Navigation

There have been numerous studies on using topological map representations for visual navigation. Works like [21] and [22] construct topological maps for localization and point goal navigation. Visual graph memory (VGM) [23] and topological semantic graph memory (TSGM) [24] create rich visual feature-based topological graph memories and employ them alongside visual goal features to navigate to the goal point. No RL, no simulation model (NRNS) [25] constructs a spatial-aware topological graph map with an RGB-D camera and navigates toward probable unexplored nodes for image goal navigation. These topological representations use visually observed features as node information, which is suitable for image goal navigation where the rich goal image features

are available. However, in the context of ObjectNav, the goal object category itself often provides less rich information than a goal image in image goal navigation. To address this limitation, we propose a topological representation that incorporates prior knowledge about the object to offer more useful information for goal object searching.

III. Method

We propose commonsense-aware object value graph navigation (OVG-Nav), a novel framework for ObjectNav. This framework is composed of two main parts: a) the construction of the object value graph (OVG), and b) the navigation modules leverage the constructed OVG. OVG is a topological map representing the running environment with both structural and semantic features from visual observations and commonsense prior knowledge. It is used for high-level exploration planning by designating subgoals in OVG-Nav. In this section, we describe the details of each part.

A. Problem Formulation

The goal of the ObjectNav task is to find an object of a certain goal category o_g in an unseen environment. The problem formulation follows the settings from [26]. At each time step t, the agent receives a front-view directional RGB-D image $[I_t^d; D_t^d]$ and an additional 360° RGB image I_t^p which we called a panoramic RGB image. The agent executes discrete actions $a \in A$ which consist of the following actions: $move\_forward$, $turn\_left$, $turn\_right$, and $stop$. An episode terminates when the $stop$ action is executed, and the episode is deemed a success if the agent stops within 1m of o_g. We assume that the agent lacks a pose sensor for obtaining its true pose p_t in the environment; therefore, the agent relies on the estimated pose $\hat{p}_t$ relative to the pose at $t = 0$.

B. Object Value Graph

The object value graph (OVG) is a topological map, $G(N, E)$, used for mapping and planning in OVG-Nav. It represents the environment through the physical connections of the observed navigable points as well as high-level visual and semantic observations. There are two types of nodes in OVG: visited nodes N_v and candidate nodes N_c. The visited nodes $n_v \in N_v$ represent positions that the agent has physically visited during the current episode, while the candidate nodes $n_c \in N_c$ denote positions that are observed to be navigable but have not yet been visited.

OVG is an episodic memory initialized at the beginning of an episode and gradually expanded during the navigation. Graph expansion occurs by adding new candidate nodes that are connected to a visited node where the agent currently resides. To determine whether to add a new candidate node based on the current observation, we build an egocentric local occupancy map using the front-view depth observation D_t^d. Navigable points on the local map that are visible and situated away from the center by an edge length l become the new candidate nodes in OVG. Edges are subsequently added to connect these candidate nodes to the current node. We set l to $1.0m$ and check the navigability of positions at heading angles of $-30°$, $0°$, and $30°$ relative to the egocentric view.

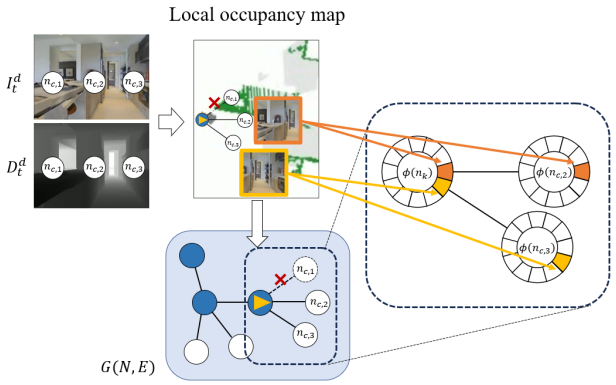


Fig. 2. Graph expansion process of OVG. We first check the navigability of positions at heading angles of $-30°$, $0°$, and $30°$ relative to the egocentric view. The candidate nodes on the navigable locations are added to the OVG based on a local metric map, and the others are discarded, represented as a red 'X' on the edge in the figure. Then the same visual features of the aligned direction are filled into the placeholders of the current node and the corresponding candidate node.

Each node of OVG consists of visual features, commonsense features, and information features. The details of the node features are described in the following.

1) Visual feature: A rich visual feature is an essential component for obtaining a high-level understanding of the environment. To extract visual features that encapsulate information about the semantic relationship with the text-provided knowledge, such as the target goal name, we use CLIP [12] visual encoder ϕ_{vis} as a visual encoder for OVG features. Since visual features are dependent on orientation even when located at the same position, a visual feature of a node, $\phi(n_t)$, uses placeholders for visual features that correspond to discretized orientations, ensuring a full representation of panoramic observations at the node position. We split a panoramic RGB observation I_t^p into $I_{t,1}^p, ..., I_{t,12}^p$, and populate the placeholders for discretized panoramic visual features accordingly as $\phi(n_t) = concat(\phi_{vis}(I_{t,1}^p), ..., \phi_{vis}(I_{t,12}^p))$. Note that we align the relative global orientation of observations with the corresponding placeholders to maintain the sequence of orientations in the visual features.

While the agent can obtain a panoramic observation from a visited node, it lacks access to such an observation from an unvisited candidate node. The only visual observation available for a candidate node originates from the orientation pointing from the agent's current location to that candidate node. We assume that the visual observations from two neighboring nodes are similar when the observation orientations align with the relative direction between the two nodes. Therefore, we use the visual observation of the direction of the candidate node as that node's visual features. Fig. 2 shows the method for updating the visual features of a candidate node. The visual features of a candidate node are stored in the placeholder that corresponds to its global orientation, and placeholders for other orientations are initialized

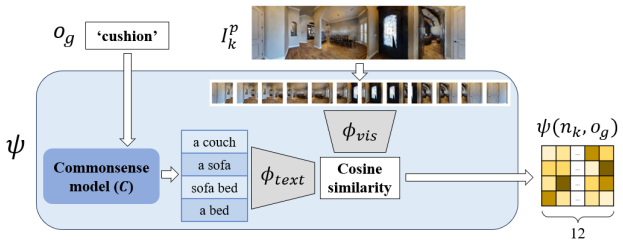


Fig. 3. Commonsense prior model to get commonsense feature. The output of C is a list of text forms of locations where the input o_g is likely to be found based on the commonsense. $\psi(n_k, o_g)$ represents the commonsense scores between the list of locations and each of the divided panoramic visual observations.

to zero. The visual features of a candidate node are updated during navigation when the agent visits connected nodes. Note that these updates consider the observed direction from the agent's position.

2) Commonsense feature: Commonsense knowledge of objects and environments facilitates more efficient exploration in ObjectNav, particularly in unseen environments. For example, if the environment configuration is unknown, it is more reasonable in commonsense to prioritize the exploration of areas resembling $rooms$ over those resembling $kitchens$ for locating a target object such as a bed. To leverage this type of prior knowledge, we use the (Comet-) Atomic 2020 [13], a pretrained commonsense knowledge model C, to identify highly probable locations for the target objects. Among the various commonsense relations provided by [13], we focus on the "A $AtLocation$ B" relation where A is the target goal object o_g. The function $C(o_g)$ yields a list of likely locations, B, for o_g based on commonsense reasoning. We define a commonsense score ψ for a node as the cosine similarity between the CLIP text-encoded feature of these likely locations and the CLIP visual features of the candidate node (Fig. 3).

$$\psi(n_k, o_g) = \{\psi(n_{k,d}, o_g) \mid d = 1, \ldots, 12\}, \tag{1}$$

$$\text{where } \psi(n_{k,d}, o_g) = \frac{\phi_{\text{text}}(C(o_g)) \cdot \phi_{\text{vis}}(I^p_{k,d})}{||\phi_{\text{text}}(C(o_g))||_2 ||\phi_{\text{vis}}(I^p_{k,d})||_2}.$$

Since CLIP embeddings can represent image-text similarity, ψ serves as a score that quantifies the similarity between the visual observation and the locations that are expected to find the target object commonly, as suggested by commonsense prior knowledge. As shown in Equation 1, the commonsense feature of a node n_k, $\psi(n_k, o_g)$, consists of scores for panoramic visual observations at the node, $\psi(n_{k,d}, o_g)$ for $d = 1, ..., 12$.

3) Information feature: The information feature contains static data about the node, unaffected by additional observations. This includes the node's position, type (i.e., whether it has been visited), and the goal object feature. Firstly, each node n_k in OVG stores a relative position p_k with respect to the initial position, which captures spatial relations among nodes. In this work, we consider the scenarios where the agent lacks GPS and compass sensors, meaning the agent relies on pose estimations rather than the oracle pose. We use visual odometry based on the RGB-D image observations to iteratively update the agent's relative pose. We employ SuperGlue [27], an end-to-end keypoint matching algorithm between sequential RGB images for visual odometry. To estimate the relative pose between two frames, we use the perspective-n-point (PnP) RANSAC [28] algorithm. This is achieved by matching keypoints on the target RGB image, obtained through SuperPoint [29] in the process of SuperGlue, with the point cloud projections of the keypoints from the source image based on the depth image. The agent updates its pose by accumulating the relative pose from the previous steps during navigation and uses this information when adding new nodes.

The type of the node is a binary feature which is 1 if the node is visited and 0 otherwise. Besides the position and type, a node also contains information about the goal object of the current episode, represented as $\phi_{text}(o_g)$. The information feature of a node, $\mu(n_k, o_g)$, is a concatenation of these three observation-invariant features.

In summary, the node feature $\theta(n_k, o_g)$ is defined as follow,

$$\theta(n_k, o_g) = concat(\phi(n_k), \psi(n_k, o_g), \mu(n_k, o_g)). \tag{2}$$

C. Navigation module

The navigation policy of OVG-Nav consists of three steps: global navigation, local goal navigation, and last-mile navigation.

1) Global navigation: Global navigation is the step of selecting a subgoal node to explore based on the *object values* in N_c. The object value of a node, $v(n_k)$, is defined as follows:

$$v(n_k) = max(1 - dist(n_k, o_g)/dist_{max}, 0) \tag{3}$$

Here, $dist(n_k, o_g)$ represents the shortest geodesic distance to the goal object from the node position, and $dist_{max}$ is the maximum allowable distance. During ObjectNav, the agent chooses the node with the highest object value as its next subgoal for navigating toward the goal object. However, the true v is inaccessible since the agent cannot determine the oracle location of the nearest goal object in an unseen environment. To address this, we propose OVGNet, shown in Fig. 4, which predicts the object value of each node in input OVG for global navigation. OVGNet is a graph convolutional network (GCN)-based neural network model that predicts object values using both node and edge features from an OVG as inputs. Here, the edge feature is the physical distance between the connected nodes.

$$\hat{v}(n_k) = [O(G(N, E))]_k \tag{4}$$

OVGNet leverages rich information from the OVG, including visual features ($\phi(n)$), spatial relationship between nodes (E), and goal-dependent prior knowledge ($\psi(n)$ and $\mu(n)$) to predict the object values. We omit the argument o_g of ψ and μ for simplification in this section. For visual feature

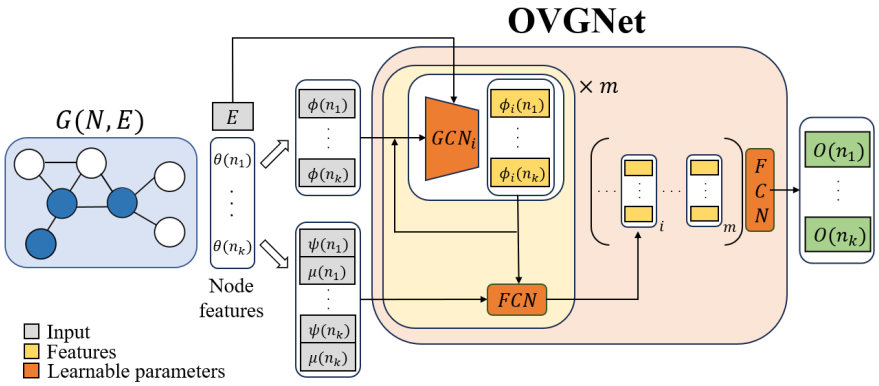


Fig. 4. The overview of OVGNet structure. Visual features from the different sizes of the receptive range are encoded by m GCN layers and the m number of features is concatenated to predict the final object value of each node. Note that only visual features are encoded over m hops of receptive ranges by the GCN layers.

encoding, OVGNet employs m GCN layers to encode visual features of nodes within m hops. Each encoded feature from the i-th GCN layer, $\phi_i(n)$, is concatenated with $\psi(n)$ and $\mu(n)$, and jointly encoded by a fully connected network (FCN) to represent the node features in i-hop observations. Note that $\psi(n)$ and $\mu(n)$ from the input $\theta(n)$ are fixed for encoding each i-hop representation to emphasize the goal-dependent features across different receptive ranges. The total m number of node features from m GCN layers are concatenated and utilized to predict object values $O(n)$ via the output FCN. OVGNet is trained in a supervised manner using a dataset of OVG and oracle v pairs collected in simulation environments.

In global navigation, the node $n_{target} = \arg\max_i \hat{v}(n_{c,i})$ is selected as the next target node to explore. We employ Dijkstra's algorithm to find the shortest path to n_{target} and designate the closest node on this path as the temporal subgoal node.

OVGNet implementation details. We use 10 stacked GCN layers with hidden sizes of 512 each in OVGNet. That is, for the experiments, OVGNet considers the receptive range of 10 hops on the OVG to predict object values.

For training OVGNet, we primarily use the l_2 norm between the oracle and predicted object values as the loss function, $Loss_{l_2}$. We also use an additional loss function, $Loss_{adj}$, defined as follows,

$$Loss_{adj} = \sum_i \sum_k \Big[l_2(v(n_i) - v(\text{Adj}(n_i)_k), \hat{v}(n_i) - \hat{v}(\text{Adj}(n_i)_k)) \Big] \quad (5)$$

where $l_2(,)$ is l_2 distance and $Adj(n_i)_k$ is k-th element of the adjacent nodes of n_i. The objective of $Loss_{adj}$ is to predict the differences in the object values of adjacent nodes, which can help in achieving spatially continuous object value predictions in an OVG. It also helps in correctly ranking the nodes relative to each other, which is important for selecting a target node for the global navigation of OVG-Nav.

The overall training loss for OVGNet is defined as,

$$Loss = \alpha Loss_{l_2} + \beta Loss_{adj}. \quad (6)$$

We use $\alpha = 1.0$ and $\beta = 100.0$ for the training process.

2) Local navigation: Local navigation involves planning actions for navigating between adjacent nodes. According to the OVG construction method, nodes are connected when they are navigable within the edge range distance l. Thus, we construct a compact egocentric local occupancy map similar to that described in III-B, which covers a range up to l instead of building a large-scale occupancy map to cover all observed areas. We employ the fast marching method (FMM) to determine actions for navigation to the relative position of the current subgoal node from the center of the egocentric local map. The local occupancy map is initialized from scratch each time the agent begins local navigation and is updated until the agent reaches the current subgoal.

Navigation in OVG-Nav consists of iterative cycles of global navigation and local navigation: 1) OVGNet predicts the next subgoal when the agent reaches the current one, and 2) actions are determined for navigation to this next subgoal node, which is adjacent to the current node.

3) Last-mile navigation: Last-mile navigation activates when the goal object is observed. We employ a finetuned version of RedNet [6] for detecting goal object masks in the front-view RGB-D observation $[I_t^d; D_t^d]$. Once the target object is detected, a point cloud corresponding to the median depth in the mask is designated as the goal point. Last-mile navigation is similar to local navigation in that the agent constructs an egocentric local occupancy map based on the its current state. Actions are determined using FMM to reach the goal point determined from the detection result. The agent performs a *stop* action at the end of the last-mile navigation.

IV. Experiments

A. Datasets

In the experiments, we use Matterport3D (MP3D) environments [14], which contain photo-realistic indoor scenes and positions of the 21 categories of goal objects in 56 training environments and 11 validation environments. For training OVGNet, we collect a dataset of $343,294$ OVG data including the oracle object values based on the trajectories in

TABLE I

OBJECTNAV RESULTS IN MP3D VAL DATASET

	Noisy Sensing	SR ↑	SPL ↑	DTS ↓
SemExp [1]	X	0.253	0.102	-
Red-Rabbit [6]	X	0.346	0.079	6.04
THDA [7]	X	0.284	0.110	5.60
PONI [2]	X	0.309	0.117	**5.10**
Habitat-Web [9]	X	0.354	0.102	-
OVG-Nav (ours)	X	**0.358**	**0.123**	5.69
SemExp [1]	O	0.112	0.051	6.63
Red-Rabbit [6]	O	0.223	0.058	6.81
PONI [2]	O	0.226	0.050	**6.10**
OVG-Nav (ours)	O	**0.278**	**0.081**	6.40

TABLE II

OVGNET PERFORMANCES RELATED TO THE COMMONSENSE PRIOR

	CS Prior	NRS ↑	ESA ↑	MDT ↓
OVGNet	X	0.909	0.675	**0.573**
OVGNet	O	**0.912**	**0.695**	0.586

TABLE III

ABLATION STUDY OF VALUE ESTIMATION IN MP3D VAL DATASET

	CS Prior	Visual Feature	SR ↑	SPL ↑	DTS ↓
OVG-Nav	X	X	0.197	0.053	6.54
OVG-Nav	O	X	0.259	0.074	6.27
OVG-Nav	X	O	0.345	0.116	**5.67**
OVG-Nav	O	O	**0.358**	**0.123**	5.69

the MP3D ObjectNav dataset for AI Habitat simulator [15]. The OVGs in the dataset are constructed based on the graph expansion method described in III-B, with the agents following the ground truth shortest path in the Habitat ObjectNav dataset. We evaluate OVG-Nav using the validation episodes in the Habitat ObjectNav dataset, employing the navigation policy described in III-C with the trained OVGNet. Note that the environments used for evaluation are not used for training OVGNet.

B. Implementation details

Graph expansion. In addition to the default edge addition in OVG based on D_t^d, we also use panoramic RGB observation I^p for adding edges more densely. We use a pretrained navigability checking network F, a CNN-based binary classification model that predicts if it is navigable up to l range ahead with an RGB input. When the agent arrives at a node, we divide I_t^p into 12 directions, $I_{t,1}^p, ..., I_{t,12}^p$, and predict navigability of each direction, $F(I_{t,1}^p), ..., F(I_{t,12}^p)$. We add a new edge if there exists a candidate node in the direction d, and $F(I_{t,d}^p) = 1$, indicating that the direction d is navigable. The additional edges help in finding a more efficient path on the graph for navigation to the target subgoal node. Note that we only use the local occupancy map, which proves more reliable for checking navigability compared to predictions by F, for the node addition since false positive candidate nodes harm the navigation policy on the graph.

Noisy sensing. We implement noisy sensing of the agent by applying both actuation noises and depth camera noise. We apply significant actuation noise by adopting the ILQR noise of LoCoBot with a noise multiplier of 1.0 as suggested in [15]. Also, we use the RedwoodDepthNoiseModel [30] for realistic noisy depth sensing.

C. Results

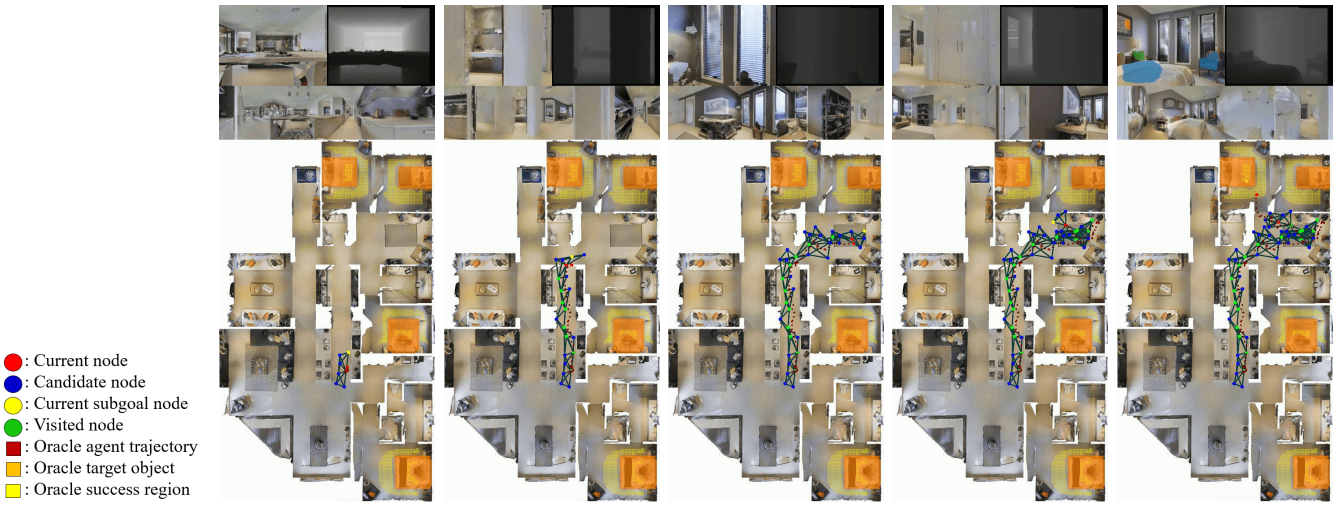
ObjectNav. For evaluating ObjectNav, we use three evaluation metrics that are commonly used in visual navigation tasks: success rate (SR), success rate weighted by path length (SPL), and distance to success (DTS). Table I shows the results of OVG-Nav and baseline object goal navigation methods on the MP3D validation dataset. Under the condition without noisy sensing, which serves as the conventional condition for evaluating ObjectNav, OVG-Nav outperforms the other baselines in SR and SPL.

In DTS, OVG-Nav shows worse performance than baseline methods like PONI [2] and THDA [7]. Since DTS is a mean value of distance to success position in m overall episodes, including failure cases, this result represents that the failure episodes of OVG-Nav are stopped farther from the goal position than these baseline methods on average. That is, OVG-Nav shows superior performance against the baseline method when it is successful but also has a risk of failing big when it fails.

We also compare OVG-Nav's performance under noisy sensing by replacing the ground truth pose sensor with the visual odometry module. OVG-Nav also shows the highest performance and the smallest decline in performance in SR and SPL with noisy sensing, compared to the code-available baseline methods. Similar to the noiseless experiments, OVG-Nav shows slightly worse performance in DTS than PONI, which is a relatively minor difference from the gap OVG-Nav outperformed in SR and SPL. The overall results show the robustness of OVG-Nav under noisy sensing as compared to both metric map-based methods [1], [2] and the end-to-end method [6].

Effectiveness of commonsense prior. To assess the effectiveness of the commonsense prior, we compare two OVGNet models that are trained with and without the commonsense prior feature in the nodes. We use three metrics for evaluating OVGNet: node ranking score (NRS), edge sign accuracy (ESA), and min distance to the oracle top-1 node (MDT). NRS is defined as $1-(rank(n_k)-1)/|N_c|$, where n_k is the selected next target node and $rank(n_k)$ is the oracle object value ranking of n_k. Since n_k is the node with the highest predicted ranking, NRS increases if the oracle ranking of n_k is higher, and if the prediction is not exactly correct, NRS increases when $|N_c|$ is large. ESA is a metric that represents the accuracy of which node has a higher object value between adjacent nodes. Comparisons of object values are important because the global navigation only uses the ranking of the nodes rather than using object values directly. MDT represents the distance to the oracle top-1 node, which has the highest object value, from the selected target node from the OVG outputs. With these metrics, the evaluation results of OVGNet in Table II show that using a commonsense prior (CS prior) helps in the accurate prediction of object values.

We also do ablation studies of OVG-Nav to verify the



(a) step 12 (b) step 53 (c) step 85 (d) step 126 (e) step 149

Fig. 5. OVG-Nav example of the object goal *bed*. The green nodes, blue nodes, red nodes, and yellow nodes represent visited nodes, candidate nodes, the current node, and the next subgoal node, respectively. The estimated goal location in the last-mile navigation is represented as a yellow square. Note that the top-down map is only for visualization and not provided to the navigation agent.

effectiveness of commonsense prior and visual features for ObjectNav (Table III). Compared to the random target node selection model (row 1), using the commonsense prior-only model (row 2) and using the visual feature-only model (row 3) both show better performance in ObjectNav. The proposed OVG-Nav model using both commonsense prior and visual features shows the highest performance among the comparisons.

Fig. 5 shows an example episode of OVG-Nav where the goal object is *bed*. Here, the oracle agent position is represented as red points, and the current subgoal node is represented as yellow. The agent observes its surrounding environment at first (step 12), then the candidate node on an aisle is selected for a subgoal by the OVGNet prediction. At step 53, the agent faces a crossroad and determines the next subgoal node in a location that seems like a living room, which is more reasonable than choosing the direction of another aisle-like location on the other side. Until step 85, the agent explores a living room yet fails to find the goal object, so it turns back to explore other places in step 126. Eventually, the agent finds the goal object *bed*, and executes the last-mile navigation to reach it (step 149). The example in Fig. 5 shows that although the oracle agent trajectory (dark red points) does not precisely follow the graph due to the positional errors, OVG-Nav achieves successful navigation that is less affected by inaccurate poses.

D. Real world experiments

We implement OVG-Nav on a mobile robot, the Clearpath Jackal platform, to deploy the policy in the real world. The robot is equipped with a front-view RGB-D camera and a 360° camera, which matches the simulation setting. We directly apply the OVGNet trained with the MP3D dataset to the real-world environment without any finetuning. We evaluate OVG-Nav in an unseen house environment (House A), with a total of 30 runs, allocated as five for each of the six dominant object goals in the environment, which represent a subset of the 21 goal object categories for the MP3D environments. Table IV shows the categories of the object goals and the success rate of each goal, and Fig. 6 shows an example of the episode with the goal category *bed*. The average distance to the nearest object goal from each starting point is 3.7m and 60% of the episodes start at points where the robot is unable to observe any goal objects at the initial point, necessitating exploration. The results show that OVG-Nav is successfully adaptable to the real-world environment for solving the ObjectNav task, including more complex scenarios than trivial cases.

TABLE IV

SUCCESS RATE OF THE REAL WORLD

Goal	Chair	Table	Sofa	Cushion	Bed	Plant	Total
House A	5/5	4/5	3/5	4/5	4/5	2/5	22/30

Fig. 6. OVG-Nav demonstration in a real-world environment. The green nodes, blue nodes, red nodes, and yellow nodes represent visited nodes, candidate nodes, the current node, and the next subgoal node, respectively. The agent successfully navigates to the goal object, *bed*, which is located in a room.

V. CONCLUSIONS

In this paper, we introduce the OVG-Nav framework, a commonsense-aware object value graph navigation for ObjectNav. OVG-Nav is a navigation framework that uses global navigation on the graph memory OVG and local navigation between nodes using a local occupancy map. For global navigation, we propose OVGNet, which predicts object values of each node in OVG by using commonsense prior and the visual features in the nodes. We show experimental results that OVG-Nav outperforms the baselines in the MP3D ObjectNav dataset and also has relatively robust performance under noisy sensing. In addition, we successfully transferred OVG-Nav to the real world with the Jackal mobile robot. Despite the satisfactory results, there also exist limitations in the proposed method. Since OVG-Nav has robustness in pose errors by avoiding accumulating them in a global metric map, it cannot handle excessive pose errors that occur in one continuous local navigation step. Also, reliance on OVGNet results can be a problem when the object values are estimated incorrectly. Nevertheless, there is still some room to improve the complement of the model in the future, such as by using chained commonsense prior or using a full metric map in parallel.

REFERENCES

[1] D. S. Chaplot, D. Gandhi, A. Gupta, and R. Salakhutdinov, "Object goal navigation using goal-oriented semantic exploration," in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[2] S. K. Ramakrishnan, D. S. Chaplot, Z. Al-Halah, J. Malik, and K. Grauman, "Poni: Potential functions for objectgoal navigation with interaction-free learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[3] H. Luo, A. Yue, Z.-W. Hong, and P. Agrawal, "Stubborn: A strong baseline for indoor object navigation," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[4] T. Campari, L. Lamanna, P. Traverso, L. Serafini, and L. Ballan, "Online learning of reusable abstract models for object goal navigation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[5] M. Zhu, B. Zhao, and T. Kong, "Navigating to objects in unseen environments by distance prediction," in *Procdeedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[6] J. Ye, D. Batra, A. Das, and E. Wijmans, "Auxiliary tasks and exploration enable objectnav," *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2021.

[7] O. Maksymets, V. Cartillier, A. Gokaslan, E. Wijmans, W. Galuba, S. Lee, and D. Batra, "Thda: Treasure hunt data augmentation for semantic navigation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[8] K. Yadav, A. Majumdar, R. Ramrakhya, N. Yokoyama, A. Baevski, Z. Kira, O. Maksymets, and D. Batra, "Ovrl-v2: A simple state-of-art baseline for imagenav and objectnav," *arXiv preprint arXiv:2303.07798*, 2023.

[9] R. Ramrakhya, E. Undersander, D. Batra, and A. Das, "Habitat-web: Learning embodied object-search strategies from human demonstrations at scale," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[10] S. Zhang, X. Song, Y. Bai, W. Li, Y. Chu, and S. Jiang, "Hierarchical object-to-zone graph for object navigation," in *Proceedings of the IEEE/CVF international conference on computer vision (CVPR)*, 2021.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[12] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *Proceedings of the International conference on machine learning (ICML)*, 2021.

[13] J. D. Hwang, C. Bhagavatula, R. Le Bras, J. Da, K. Sakaguchi, A. Bosselut, and Y. Choi, "Comet-atomic 2020: On symbolic and neural commonsense knowledge graphs," in *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI)*, 2021.

[14] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang, "Matterport3d: Learning from rgb-d data in indoor environments," *Proceddings of the International Conference on 3D Vision (3DV)*, 2017.

[15] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, D. Parikh, and D. Batra, "Habitat: A Platform for Embodied AI Research," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[16] D. S. Chaplot, D. Gandhi, S. Gupta, A. Gupta, and R. Salakhutdinov, "Learning to explore using active neural slam," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[17] W. Yang, X. Wang, A. Farhadi, A. Gupta, and R. Mottaghi, "Visual semantic navigation using scene priors," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[18] A. Pal, Y. Qiu, and H. Christensen, "Learning hierarchical relationships for object-goal navigation," in *Proceedings of the Conference on Robot Learning (CoRL)*, 2021.

[19] J. Park, T. Yoon, J. Hong, Y. Yu, M. Pan, and S. Choi, "Zero-shot active visual search (zavis): Intelligent object search for robotic assistants," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[20] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, D. Gordon, Y. Zhu, A. Gupta, and A. Farhadi, "AI2-THOR: An Interactive 3D Environment for Visual AI," *arXiv*, 2017.

[21] N. Savinov, A. Dosovitskiy, and V. Koltun, "Semi-parametric topological memory for navigation," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[22] K. Fang, A. Toshev, L. Fei-Fei, and S. Savarese, "Scene memory transformer for embodied agents in long-horizon tasks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[23] O. Kwon, N. Kim, Y. Choi, H. Yoo, J. Park, and S. Oh, "Visual graph memory with unsupervised representation for visual navigation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[24] N. Kim, O. Kwon, H. Yoo, Y. Choi, J. Park, and S. Oh, "Topological semantic graph memory for image-goal navigation," in *Proceedings of the Conference on Robot Learning (CoRL)*, 2023.

[25] M. Hahn, D. S. Chaplot, S. Tulsiani, M. Mukadam, J. M. Rehg, and A. Gupta, "No rl, no simulation: Learning to navigate without navigating," in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[26] D. Batra, A. Gokaslan, A. Kembhavi, O. Maksymets, R. Mottaghi, M. Savva, A. Toshev, and E. Wijmans, "ObjectNav Revisited: On Evaluation of Embodied Agents Navigating to Objects," in *arXiv:2006.13171*, 2020.

[27] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperGlue: Learning feature matching with graph neural networks," in *Proceedings of the IEEE/CVF international conference on computer vision (CVPR)*, 2020.

[28] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[29] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superpoint: Self-supervised interest point detection and description," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops (CVPR workshops)*, 2018.

[30] S. Choi, Q.-Y. Zhou, and V. Koltun, "Robust reconstruction of indoor scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.