2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

# Commonsense-guided Object Graph Reasoning with Policy Regularization for Object Goal Navigation

Yiyue Meng[1], Aolin Li[1], Jiao Zhan[2], Shenxin Li[3], and Chi Guo[1,*]

***Abstract*— Object goal navigation aims to guide an agent to find a specific target object in an unseen environment using only first-person visual observations. It requires the agent to enhance scene understanding and train a robust navigation policy. To address this, we proposed two complementary techniques, commonsense-guided object graph reasoning (COGR) and policy regularization (PR). Specifically, COGR improves the agent's scene understanding by integrating object relationships, including category proximity and spatial correlation. It extracts co-occurrence embeddings of the target object from a large language model (LLM) as commonsense knowledge to guide object graph reasoning, enabling the agent to reason beyond visual co-occurrence observed in training environments. PR is a knowledge distillation-inspired regularization mechanism, where a commonsense-free model is used to regularize the navigation policy of the commonsense-guided model. We propose PR to mitigate potential performance degradation caused by knowledge bias from the LLM, enabling the training of a more robust navigation policy. Experiments in the AI2-Thor and RoboThor environments demonstrate the effectiveness and efficiency of our proposed method, and real-world deployment further validates its transferability.**

## I. INTRODUCTION

Object goal navigation [1], [2], [3] is a fundamental problem in the field of embodied intelligent robotics. It requires an agent to find a specific target object in an unseen environment using only first-person visual observations. The agent must learn the ability to precisely navigate to the target object. Since the environment is unknown and no complete geometric map is available, the key challenge lies in enabling the agent to enhance scene understanding and train a robust navigation policy.

With first-person observations as input, early works [4], [5], [6], [7] train navigation policies using reinforcement learning (RL) in training environments. However, due to the limited ability of scene understanding, the agent struggles to infer the location of the target object when it is not visible, leading to inefficient exploration and poor generalization. Recent works [8], [9], [10], [11], [12], [13], [14] attempt to extract relational information between objects by constructing object graphs to enhance scene understanding. In these graphs, edges represent relationships between objects, which are typically extracted from external datasets such as Visual Genome [15] or collected in training environments using object detectors such as DETR [16]. However, due to differences in layout and object distribution between training and test environments, these graphs may leave the agent with incomplete or incorrect knowledge about object relationships, as shown in Fig. 1. Thus, the generalization ability of these object graphs remains limited [17].

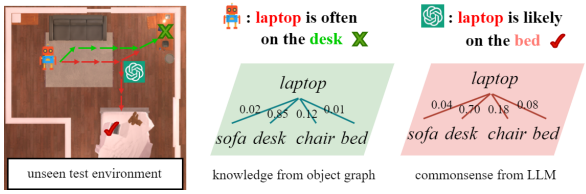


Fig. 1. In the training environment, the agent learns that a laptop is often on the desk, leading to incomplete or incorrect knowledge in the object graph (middle). However, in an unseen test environment (left), the laptop may instead be located on the bed. Commonsense knowledge extracted from the LLM correctly predicts that a laptop can also be on the bed (right), helping the agent to find the laptop.

Recently, the potential of large language models (LLMs) for knowledge extraction and integration in robotics has attracted increasing attention [18]. Pre-trained on vast amounts of textual data, LLMs encode rich commonsense knowledge, enabling the agent to reason about surrounding scenes and plausible actions in a human-like manner. Many studies have explored leveraging LLMs for planning or probabilistic reasoning [19], [20], [21], [22], [23], [24], such as inferring object co-occurrence. Compared to object graphs collected from training environments, LLM-based methods provide more comprehensive commonsense knowledge. However, relying solely on LLMs can lead to limited performance when the extracted commonsense does not align well with the observations [24], [25].

To address these challenges, we propose two complementary techniques, commonsense-guided object graph reasoning (COGR) and policy regularization (PR). Our method not only extracts commonsense knowledge from the LLM for object graph reasoning but also maintains robustness when the extracted knowledge does not fully align with visual observations. Specifically, COGR leverages commonsense knowledge from the LLM to guide object graph reasoning. Instead of relying solely on visual co-occurrence from training environments, COGR extracts co-occurrence embeddings of the target object from the LLM, where candidate objects that are likely to co-occur with the target object serve as intermediate cues, as shown in Fig. 1. This reflects the way humans reason about potential target locations. Inspired by

[1]Yiyue Meng, Aolin Li and Chi Guo are with the School of Robotics, Wuhan University, Wuhan 430072, China (e-mail: {mengyiyue, liaolin, guochi}@whu.edu.cn).[2]Jiao Zhan is with the Hubei Luojia Laboratory, Wuhan 430072, China (e-mail: zhanjiao1994@whu.edu.cn).[3]Shenxin Li is with the School of Geodesy and Geomatics, Wuhan University, Wuhan 430072, China (e-mail: 2024302142027@whu.edu.cn).This work was supported by the Major Science and Technology Project of Hubei Province (No. 2022AAA009), the Key Research and Development Program of Wuhan, and the Device Upgrade Project of the Digital Intelligence Education and Teaching Platform of Wuhan University.*Corresponding author: Chi Guo.

the TransH [26], we incorporate co-occurrence embeddings into object graph reasoning, enabling the inferred object relationships to be guided by commonsense knowledge.

To mitigate potential performance degradation caused by knowledge bias from the LLM, we further propose PR, a knowledge distillation-inspired regularization mechanism in which a commonsense-free model is used to regularize the navigation policy of the commonsense-guided model. PR integrates the standard asynchronous advantage actor-critic (A3C) [27] RL algorithm with soft policy regularization to train a more robust navigation policy. Experiments in the AI2-Thor and RoboThor environments show that the combination of COGR and PR significantly improves the effectiveness and efficiency of navigation in unseen environments. Beyond simulation, we have validated the practical applicability of our method through real-world robotic experiments. In summary, our contributions are listed as follows:

- We propose COGR to enhance scene understanding. COGR extracts co-occurrence embeddings of the target object from the LLM as commonsense knowledge to guide object graph reasoning, enabling the agent to reason beyond visual co-occurrence observed in training environments.
- We propose PR to improve navigation policy. PR integrates A3C with soft policy regularization to mitigate potential performance degradation caused by knowledge bias from the LLM, enabling the agent to train a robust navigation policy.
- Experiments in the AI2-Thor and RoboThor environments demonstrate the effectiveness and efficiency of our proposed method. Furthermore, we validate the practical applicability of our method through successful deployment on a real-world robot.

## II. Related Work

### A. Object Goal Navigation

Object goal navigation is a fundamental task in embodied intelligent robotics. Current methods for this task can be categorized into two types: end-to-end methods and modular methods. Due to the development of deep learning, end-to-end methods directly map visual observations to navigation policies via RL. Early works encode the current and target observation as visual inputs for RL to train a policy network guided by environment rewards [4]. To adaptively adjust policies in unseen environments, [5] introduces model agnostic meta learning (MAML) and combines it with RL, enabling the agent to continually adapt as it interacts with new environments. More recent works enhance semantic understanding through attention mechanisms, such as a spatial attention model [6] or a visual Transformer model [7]. Due to the lack of explicit map construction, end-to-end methods exhibit limited generalization. Consequently, there is growing interest in investigating modular map-based methods.

Modular methods use independent modules for perception, localization, mapping, and planning. They construct an egocentric map to support simultaneous localization and mapping (SLAM). Early works build occupancy grid maps [28] or topological maps [29]. More recent works [30], [31], [32] construct semantic maps, which serve as inputs to the planning module for estimating the next exploration location toward the target. While modular methods perform well in object goal navigation, their reliance on egocentric maps can be problematic due to potential distortions caused by inaccurate pose estimation and accumulated pose errors [33].

### B. Prior Knowledge for Navigation

Recent works [8], [9], [10], [11], [12], [13], [14], [17] introduce prior knowledge by constructing object graphs to enhance scene understanding. By reasoning about object relationships in these graphs, the agent can infer the likely location of the target object. Early works build object graphs from external datasets [8], while more recent works capture visual co-occurrence in training environments using object detectors like Faster R-CNN [9], [10], [12] and DETR [11], [14], [17]. Furthermore, [13] proposes to align object graphs with visual perception using contrastive language-image pretraining (CLIP) [34]. However, due to differences in layout and object distribution between training and test environments, the knowledge encoded in these graphs may be incomplete or incorrect in unseen environments. Our proposed COGR addresses this by extracting co-occurrence embeddings of the target object from the LLM as commonsense knowledge to guide object graph reasoning.

### C. Large Language Model for Navigation

Recent works [19], [20], [21], [22], [23], [24], [25] investigate leveraging LLMs to provide commonsense knowledge for navigation tasks. By utilizing knowledge derived from large-scale textual data, the agent can enhance scene understanding and improve planning in unseen environments. Early works [19], [20], [21], [25] directly use LLM outputs to infer candidate next waypoints or actions. Recent works [22], [23], [24] integrate LLM-based commonsense reasoning with semantic maps or object graphs, enabling more informed navigation decisions. However, the extracted knowledge may not always align with the agent's visual observations, limiting the robustness of navigation. Our proposed PR addresses this issue by integrating A3C with soft policy regularization, complementing COGR to enable more robust navigation policy training.

## III. Proposed Method

Our objective is to enhance scene understanding and train a robust navigation policy for object goal navigation. To achieve this, our navigation framework comprises two major components, as illustrated in Fig. 2: (1) commonsense-guided object graph reasoning (COGR), which extracts co-occurrence embeddings of the target object from the LLM to guide object graph reasoning; (2) Transformer-based visual decoder (TVD), which integrates the object graph with the visual observation to produce state representation for downstream policy learning; (3) policy regularization (PR), which integrates A3C with soft policy regularization to facilitate the training of a more robust navigation policy.

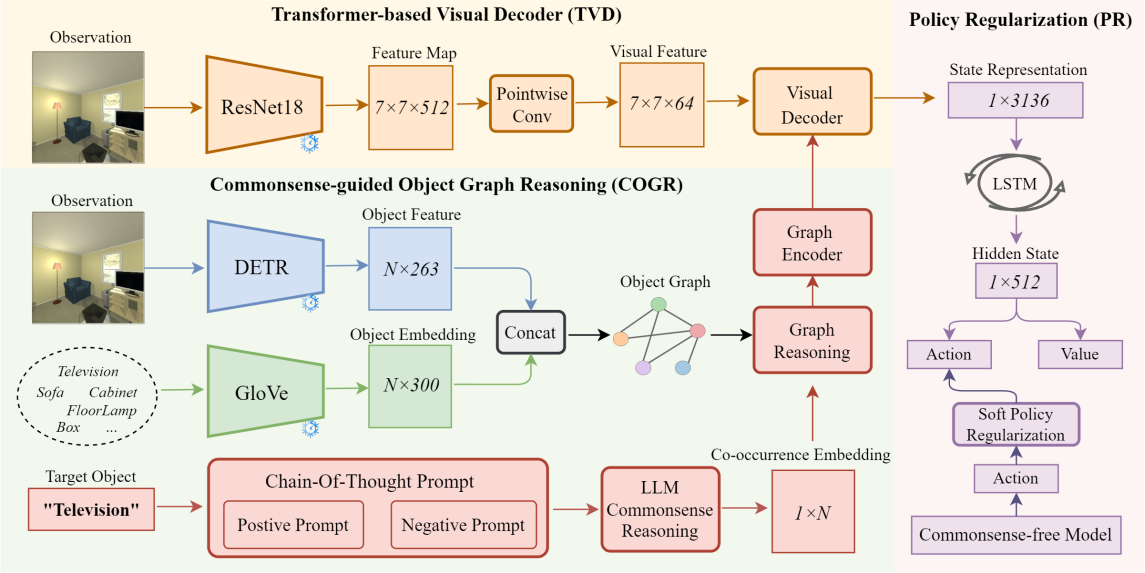


Fig. 2. Overview of our framework. The framework comprises three parts: COGR, TVD, and PR. COGR extracts co-occurrence embeddings of the target object from the LLM to guide object graph reasoning. TVD uses a Transformer-based decoder to integrate the object graph with the visual observation to produce state representation. PR integrates A3C with a soft policy regularization to train a more robust navigation policy.

### A. Task Definition

The goal of object goal navigation is to find a given target object in an unseen environment. At the beginning of each episode, the agent is initially placed at a random location. During navigation, at every time step $t$, the agent perceives only RGB images from an egocentric viewpoint and predicts its next actions. The agent performs one of several discrete actions, including *MoveAhead*, *RotateLeft*, *RotateRight*, *LookUp*, *LookDown*, and *Done*. *MoveAhead* action moves the agent forward by 0.25 m. *RotateLeft* and *RotateRight* rotate the agent by 45°. *LookUp* and *LookDown* tilt the agent's view upward and downward by 30°. An episode is considered successful if the agent executes *Done* within a maximum number of steps at a spot where the target object is within a specified distance (e.g., less than 1.5 m) and is visible in the agent's field of view.

### B. Commonsense-guided Object Graph Reasoning

Since the agent can only perceive an egocentric view of the environment, learning an enhanced scene understanding is crucial. To this end, we construct an object graph to exploit object relationships, including category proximity and spatial correlation. Humans naturally rely on commonsense knowledge to infer the most likely locations of a target object. For example, when finding a remote control, one would typically first examine nearby objects that are likely to co-occur, such as a television or a sofa. Inspired by this intuition, we propose COGR to query the LLM to rank candidate object categories for each target object based on likelihood of co-occurrence, which serves as co-occurrence embeddings to guide object graph reasoning.

We first construct the object graph using the object detector DETR following [7]. Given an RGB image as the visual observation $I_t$, DETR identifies all object instances of interest and transforms $N$ encoded $d$-dimensional features $\mathbb{R}^{N\times d}$ (i.e., $d = 256$) into $N$ detection results, including bounding boxes, confidence scores, and semantic labels. To create node features, we concatenate the $d$-dimensional feature, normalized bounding box $\mathbb{R}^{N\times 4}$, confidence score $\mathbb{R}^{N\times 1}$, top-rated semantic label $\mathbb{R}^{N\times 1}$, and text embedding (i.e., GloVe [35]) $\mathbb{R}^{N\times 300}$ for each detected object. To incorporate information about the target object, a one-hot encoded target vector $\mathbb{R}^{N\times 1}$ is also concatenated. This forms the node features $\mathbb{R}^{N\times 563}$ for constructing a fully connected undirected object relation graph $G(V, E)$.

We then use COGR to extract co-occurrence embeddings from the LLM to guide object graph reasoning. Specifically, we query GPT-4o mini (API version: gpt-4o-mini-2024-07-18) to estimate co-occurrence likelihoods between the target object and all detected objects from DETR. To infer this commonsense knowledge, we design prompts that include both positive prompts (e.g., "*which object is most likely to be near the target object?*") and negative prompts (e.g., "*which object is most unlikely to be near the target object?*"). Relying solely on the estimations of positive prompts may lead to estimation uncertainty, as outlined in [21]. To obtain robust and reliable likelihood estimations, we further use a chain-of-thought (COT) prompting technique [36]. As illustrated in Fig. 3, we combine COT positive and negative prompts to improve the quality and consistency of the likelihood estimations.

In summary, we collect the set of target objects $P$ and the set of detected objects $O$ from the environments. The likelihood scores from COT positive prompts $LLM_{pos}(P, O) \in [0, 1]$ are combined with those from COT negative prompts $LLM_{neg}(P, O) \in [0, 1]$ to construct the co-occurrence matrix $M$ for each target object as follows:

$$LLM_{pos}(P, O) - LLM_{neg}(P, O) = m \in [-1, 1] \quad (1)$$

where each row of $M$ represents the co-occurrence embed-

**Background:** Imagine you are a robot undertaking your first exploration of an indoor environment. Your task is to search for a specific object. You will receive observations that include all object categories within the room. Using common sense derived from Large Language Models (LLMs), your responsibility is to assess the likelihood of locating the target object near each object category. This likelihood is quantified on a scale ranging from 0 to 1, where a higher value indicates a greater probability of finding the target object near that object category.

**Example:** For instance, if you are in a room finding a remote control, you can find it near a television. Therefore, you should assign a higher likelihood estimation to the television.

**Question:** In an indoor environment, you have observed the following object categories: "alarm clock", "book", "bowl", "cell phone", "chair", "coffee machine", "desk lamp", "floor lamp", "fridge", "garbage can", "kettle", "laptop", "light switch", "microwave", "pan", "plate", "pot", "remote control", "sink", "stove burner", "television", "toaster". If you are finding a "alarm clock", could you please provide me with the likelihood of locating the target object near these object categories.

COT positive prompts

**Background:** Imagine you are a robot undertaking your first exploration of an indoor environment. Your task is to search for a specific object. You will receive observations that include all object categories within the room. Using common sense derived from Large Language Models (LLMs), your responsibility is to assess the likelihood of unlikely locating the target object near each object category. This likelihood is quantified on a scale ranging from 0 to 1, where a higher value indicates a lower probability of finding the target object near that object category.

**Example:** For instance, if you are in a room finding a remote control, you should not waste time finding it near a toaster. Therefore, you should assign a higher likelihood estimation to the toaster.

**Question:** In an indoor environment, you have observed the following object categories: "alarm clock", "book", "bowl", "cell phone", "chair", "coffee machine", "desk lamp", "floor lamp", "fridge", "garbage can", "kettle", "laptop", "light switch", "microwave", "pan", "plate", "pot", "remote control", "sink", "stove burner", "television", "toaster". If you are finding a "alarm clock", could you please provide me with the likelihood of unlikely locating the target object near these object categories.

COT negative prompts

Fig. 3. Example of COT prompts. We combine COT positive and negative prompts to obtain robust and reliable co-occurrence likelihood estimations.

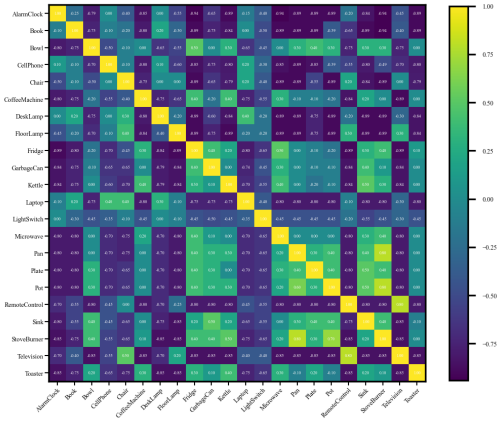


Fig. 4. Example of co-occurrence embeddings. We select 22 objects to represent. Each row is a co-occurrence embedding for a target object.

ding for a target object $p \in P$, denoted as $m_p \in \mathbb{R}^{1\times N}$, as shown in Fig. 4. Inspired by TransH [26], we integrate commonsense knowledge into object graph reasoning by embedding objects into distributed representations guided by this co-occurrence embedding and refine their relationships accordingly. Specifically, we construct an adjacency correlation matrix $A \in \mathbb{R}^{N\times N}$ for the graph $G(V, E)$, where each entry $\alpha(i, j)$ represents the connection weight between object $v_i$ and object $v_j$. Guided by a co-occurrence embedding $m_p$, we project object features into a hyperplane defined by $m_p$:

$$v_{i;m_p} = v_i - w_{m_p}^\top v_i w_{m_p} \tag{2}$$

$$v_{j;m_p} = v_j - w_{m_p}^\top v_j w_{m_p} \tag{3}$$

where $w_{m_p}$ is the normal vector defining the hyperplane of the co-occurrence embedding $m_p$, and $v_{i;m_p}$ and $v_{j;m_p}$ are the object features of $v_i$ and $v_j$ after being projected into the hyperplane $w_{m_p}$. Following this projection, we compute the connection weight $\alpha_{m_p}(i, j)$ using scaled dot-product attention [37] followed by a softmax operation:

$$\alpha_{m_p}(i, j) = \text{softmax}(\frac{v_{i;m_p} \cdot v_{j;m_p}^\top}{\sqrt{d_v}}) \tag{4}$$

where $d_v$ is the dimensionality of the object features. After obtaining $\alpha_{m_p}(i, j)$, we use graph attention layers (GAL) to aggregate node features and add residual connections to stabilize the feature propagation, leading to commonsense-guided graph features:

$$v_i^{(l+1)} = \sigma(\sum_{j\in N(i)} \alpha_{m_p}(i, j)W^{(l)}v_j^{(l)}) \tag{5}$$

where $v_i^{(l)}$ and $v_j^{(l)}$ are the features of nodes $i$ and $j$ at the $l$-th layer, $N(i)$ is the neighbors of node $i$, $\sigma$ is the ReLU activation function, $W^{(l)}$ is the weight matrix at layer $l$, and $v_i^{(l+1)}$ is the updated feature of node $i$ at $l + 1$-th layer.

### C. Transformer-based Visual Decoder

After COGR, the features of the object graph $G(V, E)$ are encoded into $\mathbb{R}^{N\times 64}$. We use a Transformer-based visual decoder (TVD) to integrate these graph features with visual observations, producing state representation for downstream policy learning. The visual observation $I_t$ is first encoded by a pre-trained ResNet-18 and projected into $\mathbb{R}^{7\times 7\times 64}$ through a pointwise convolution. To preserve the spatial position information, a 2D sinusoidal embedding is added before flattening $I_t$ into $\mathbb{R}^{49\times 64}$. TVD is formulated following [37]:

$$\begin{aligned} TVD(I_t, G(V, E)) &= \text{LN}(\text{FC}(c_t) + c_t) \\ \text{where} \quad c_t &= \text{LN}(\text{MHA}(I_t, G(V, E)) + I_t) \end{aligned} \tag{6}$$

where LN is a layer normalization, FC is a fully connected layer, and MHA is a multi-head attention mechanism. This decoding process enables effective fusion of the visual observation and the object graph, yielding state representation for downstream policy learning.

### D. Policy Regularization

To ensure robust navigation in unseen environments, it is crucial not only to leverage commonsense knowledge via COGR but also to mitigate potential biases in this knowledge. While COGR provides commonsense information about object co-occurrence, this knowledge may not always align with the agent's visual observations, potentially misleading policy learning. To address this, we propose PR inspired by knowledge distillation [38], [39] that integrates standard A3C training with soft policy regularization, enabling a more robust navigation policy.

As shown in Fig. 2, the obtained state representation from TVD is fed into a long-short term memory (LSTM) network [40], whose hidden states are fed into A3C to train navigation policy. A3C uses an actor-critic architecture, where the actor outputs the action distribution $\pi(a_t|s_t;\theta)$ and the critic estimates the state value $V(s_t;\theta_v)$. The corresponding loss functions are:

$$\mathcal{L}_{actor} = -\mathbb{E}[\log \pi(a_t|s_t;\theta)(\sum_{t=0}^{T} \gamma^t r_t - V(s_t;\theta_v))] \quad (7)$$

$$\mathcal{L}_{critic} = \mathbb{E}[(\sum_{t=0}^{T} \gamma^t r_t - V(s_t;\theta_v))^2] \quad (8)$$

where the advantage term $\sum_{t=0}^{T} \gamma^t r_t - V(s_t;\theta_v)$ represents the discrepancy between the cumulative discounted reward and the critic's value estimate.

PR introduces a commonsense-free model that infers object relationships without relying on co-occurrence embeddings, which is then used to regularize the commonsense-guided (COGR-enhanced) model. The two models have separate parameters. Training proceeds in two stages: first, the commonsense-free model is trained using the standard actor and critic losses. Then, the commonsense-guided model is trained using A3C combined with policy distillation. The distillation is applied as soft policy regularization on the actor outputs by minimizing the Kullback-Leibler (KL) divergence [41] between the action distributions of the two models:

$$\mathcal{L}_{distillation} = -\sum_{t=0}^{T} P_t \log(\frac{Q_t}{P_t}) \quad (9)$$

where $P_t = \pi(a_t|s_t;\theta)$ is the action probability from the commonsense-guided model and $Q_t$ is from the commonsense-free model. The total loss for training the commonsense-guided model is:

$$\mathcal{L}_{total} = \lambda_a \mathcal{L}_{actor} + \lambda_c \mathcal{L}_{critic} + \lambda_d \mathcal{L}_{distillation}, \quad (10)$$

where $\lambda_a$, $\lambda_c$, and $\lambda_d$ are the weights for actor, critic, and distillation losses, respectively (set as $\lambda_a = 1$, $\lambda_c = 0.5$, $\lambda_d = 1$ in our implementation). This design preserves the unbiased generalization of the commonsense-free model while leveraging the commonsense adaptability of the commonsense-guided model, resulting in a robust navigation policy.

## IV. Experiments

### A. Experimental Setup

*Datasets:* We evaluate the navigation performance of our proposed method using the AI2-Thor [42] and RoboThor [43] simulation environments. The AI2-Thor environment consists of 120 scenes categorized into four types: kitchen, living room, bedroom, and bathroom. We select 20 scenes for training, 5 for validation, and 5 for testing for each scene type. Following the setting in [5], [6], [9], [10], [11], we choose 22 types of objects as targets. The RoboThor environment consists of 89 apartment layouts, with 75 apartments for training and validation. Compared to AI2-Thor, RoboThor has a 2.4 times larger area and a 5.5 times longer trajectory length. We select 60 apartments for training, 5 for validation, and 10 for testing. Following prior works [11], [12], we choose 12 types of objects as targets.

TABLE I

The ablation study of different components

| ID | Component | | | AI2-Thor (ALL/L≥ 5) | | RoboThor (ALL/L≥ 5) | |
|---|---|---|---|---|---|---|---|
| | COGR | TVD | PR | SR↑(%) | SPL↑(%) | SR↑(%) | SPL↑(%) |
| 1 | | | | 71.43/60.72 | 43.47/42.18 | 41.34/31.42 | 24.38/17.45 |
| 2 | ✓ | | | 77.13/69.48 | 44.52/44.18 | 49.81/37.15 | 28.92/22.42 |
| 3 | ✓ | ✓ | | 78.44/70.54 | 45.61/45.50 | 50.64/39.27 | 29.39/23.63 |
| 4 | ✓ | ✓ | ✓ | **80.21/73.52** | **46.01/46.85** | **52.67/41.63** | **30.96/24.94** |

TABLE II

Comparisons of the commonsense-free model, the commonsense-guided model, and our full model

| ID | Model | AI2-Thor (ALL/L≥ 5) | | RoboThor (ALL/L≥ 5) | |
|---|---|---|---|---|---|
| | | SR↑(%) | SPL↑(%) | SR↑(%) | SPL↑(%) |
| 1 | GAL+TVD | 76.98/69.52 | 45.73/45.58 | 49.12/37.83 | 28.47/22.24 |
| 2 | COGR+TVD | 78.44/70.54 | 45.61/45.50 | 50.64/39.27 | 29.39/23.63 |
| 3 | COGR+TVD+PR | **80.21/73.52** | **46.01/46.85** | **52.67/41.63** | **30.96/24.94** |

TABLE III

The ablation study of different LLMs

| ID | LLM | AI2-Thor (ALL/L≥ 5) | | RoboThor (ALL/L≥ 5) | |
|---|---|---|---|---|---|
| | | SR↑(%) | SPL↑(%) | SR↑(%) | SPL↑(%) |
| 1 | GPT-4o mini | **80.21/73.52** | **46.01/46.85** | **52.67/41.63** | **30.96/24.94** |
| 2 | Llama3.1-70B | 79.42/72.35 | 45.06/45.81 | 51.53/40.28 | 29.31/23.82 |
| 3 | Qwen2-72B | 79.32/72.10 | 45.14/46.12 | 50.92/39.85 | 29.58/23.87 |

*Metrics:* Following [5], [6], [9], we evaluate the navigation performance of our proposed method using two key metrics: success rate (SR) and success weighted by path length (SPL). SR measures the agent's success in locating the target object, calculated as $\mathrm{SR} = \frac{1}{N}\sum_{i=1}^{N} S_i$, where $N$ represents the total number of episodes, and $S_i$ is a binary function indicating whether the $i$-th episode is successful. SPL considers both success rate and path length, calculated as $\mathrm{SPL} = \frac{1}{N}\sum_{i=1}^{N} S_i \frac{L_i}{max(L_i,P_i)}$, where $L_i$ represents the optimal trajectory length of the $i$-th episode, and $P_i$ denotes the path length of the $i$-th episode. We evaluate the performance on all trajectories (ALL) and those with an optimal path length of at least 5 (L≥5).

*Implementation details:* Following [5], [9], [10], [11], [12], [13], the RL reward is set as follows: the agent receives a penalty of -0.01 for each step and a reward of 5 if the episode is successful. For optimization, we use the Adam optimizer [44] with a learning rate of $10^{-4}$. We conduct a total of 6 million episodes to train our methods. During the evaluation phase, we execute the same number of episodes for each scene type, resulting in a total of 1000 episodes. The method that achieves the highest SR will be evaluated on the test scenes, and its performance will be reported as the final results. Note that for PR in Sec. III-D, we train the commonsense-free model and select the variant achieving the highest success rate (SR) on the test scenes. For the DETR training, we refer to [7] and fine-tune it using the training data from the environments.

### B. Ablation Experiments

*Ablation studies of different components:* We decompose our proposed method into different components: COGR, TVD, and PR. Table I demonstrates the effectiveness of each component. Specifically, row 1 is the baseline following [11], [17], which constructs the features concatenated from the visual observation and the object graph as the state representation. This representation is then fed into an LSTM to produce hidden states for A3C learning. It is observed that COGR significantly improves the baseline performance (row 2), while TVD brings additional but relatively modest gains (row 3). Additionally, introducing PR further improves navigation performance in SR and SPL. Overall, our method outperforms the baseline with gains of 8.78/12.80 and 2.54/4.67 in SR and SPL (ALL/L≥5, %) in AI2-Thor and 11.33/10.21 and 6.58/7.49 in SR and SPL (ALL/L≥5, %) in RoboThor. The experimental results indicate that our method can effectively and efficiently guide navigation in unseen environments.

*Effectiveness of PR:* To further evaluate the effectiveness of our proposed PR, we provide detailed results in Table II, comparing the commonsense-free model (row 1), the commonsense-guided model (row 2), and our full model (row 3), which uses the former to regularize the policy of the latter through PR. Note that although the commonsense-free model computes object relationships without projecting object features into the hyperplane of co-occurrence embeddings, it still uses GAL to extract graph features. TVD is incorporated in both the commonsense-free and commonsense-guided models. As shown in Table II, our method outperforms both the commonsense-free and commonsense-guided models, achieving the best overall performance.

*Impacts of different LLMs:* The co-occurrence embeddings from the LLM are used to guide object graph reasoning. Since the effectiveness of our proposed method depends on the commonsense reasoning ability of the LLMs, we evaluate navigation performance using different co-occurrence embeddings extracted from several LLMs, including GPT-4o mini, Llama3.1-70B, and Qwen2-72B. As shown in Table III, Llama3.1-70B and Qwen2-72B achieve comparable performance, while GPT-4o mini achieves the best results.

### C. Comparisons With Other Methods

*Quantitative analysis:* Our method is compared with three categories of related works, as shown in Table IV. (I) End-to-end methods. The original implementations of SP [8], SAVN [5], and SpAtt [6] do not use object detectors for enhancing scene understanding. For a fair comparison, we re-implement these methods with object detection features. Compared to the recently proposed AKGVP-CI [13], our method brings the grains of 3.43/8.07 and 6.38/7.84 in SR and SPL (ALL/L≥5, %) in AI2-Thor and 8.14/8.95 and 3.35/4.39 in SR and SPL (ALL/L≥5, %) in RoboThor. (II) Modular methods. These methods rely on explicit semantic maps to guide the agent toward the target object more accurately. Since SSCNav [31] and PONI [32] report results in AI2-Thor, we compare our method with them in this environment. Our method achieves moderate improvements in SR while yielding substantial gains in SPL. (III) LLM-based methods. These methods leverage LLMs for planning or probabilistic reasoning. ESC [19] and LGX [20] report results only in RoboThor, while LOAT [22] reports results only in AI2-Thor. Our method significantly outperforms these methods, benefiting from environment-specific policy learning via RL.

*Qualitative analysis:* As illustrated in Fig. 5, we compare our method with L-sTDE and AKGVP-CI in both AI2-Thor and RoboThor environments. In these scenes, the agent starts from a random location where the target object is not initially visible. L-sTDE and AKGVP-CI often struggle with

TABLE IV

COMPARISONS WITH THE RELATED WORKS IN THE AI2-THOR AND ROBOTHOR ENVIRONMENTS

| ID | Method | AI2-Thor (ALL/L≥ 5) | | RoboThor (ALL/L≥ 5) | |
|---|---|---|---|---|---|
| | | SR↑(%) | SPL↑(%) | SR↑(%) | SPL↑(%) |
| I | Random | 3.56/0.27 | 1.73/0.07 | 0.00/0.00 | 0.00/0.00 |
| | SP [8] | 62.16/50.86 | 37.01/34.17 | 28.04/21.66 | 17.63/15.14 |
| | SAVN [5] | 63.32/52.38 | 37.62/35.31 | 28.42/22.13 | 17.82/15.34 |
| | SpAtt [6] | 65.61/54.11 | 38.93/35.89 | 28.53/22.35 | 18.27/15.45 |
| | ORG [9] | 66.38/55.55 | 38.42/36.26 | 29.61/22.53 | 19.23/15.73 |
| | ORG+TPN [9] | 67.31/57.41 | 39.53/38.27 | 30.01/22.25 | 20.51/16.64 |
| | HOZ [10] | 70.62/62.75 | 40.02/39.24 | 32.27/24.83 | 20.48/16.89 |
| | VTNet [7] | 72.20/63.40 | 44.90/44.00 | 31.62/23.48 | 19.63/16.02 |
| | DOA [11] | 74.32/67.88 | 40.27/40.36 | 36.22/30.16 | 22.12/18.32 |
| | L-sTDE [12] | 74.19/64.01 | 40.30/39.97 | 42.13/32.04 | 24.54/17.44 |
| | AKGVP [13] | 73.63/63.51 | 40.66/39.63 | 39.69/28.55 | 25.84/18.79 |
| | AKGVP-CI [13] | 76.78/65.45 | 39.63/39.01 | 44.53/32.68 | 27.61/20.55 |
| | **Ours** | **80.21/73.52** | **46.01/46.85** | **52.67/41.63** | **30.96/24.94** |
| II | SSCNav [31] | 77.14/71.73 | 35.09/34.33 | -/- | -/- |
| | PONI [32] | 78.58/72.92 | 37.27/36.40 | -/- | -/- |
| III | ESC [19] | -/- | -/- | 38.10/- | 22.20/- |
| | LGX [20] | -/- | -/- | 35.00/- | 21.90/- |
| | LOAT [22] | 73.12/65.26 | 39.56/39.68 | -/- | -/- |

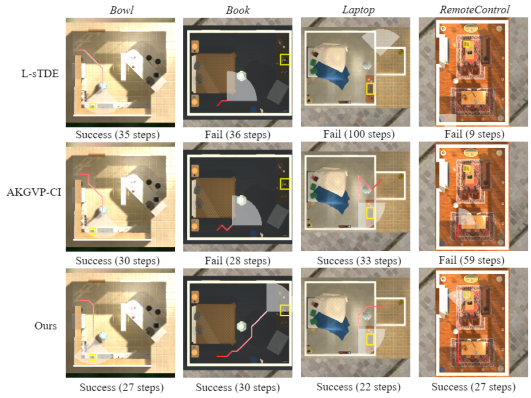


Fig. 5. **Visualization of the agent's trajectories in AI2-Thor .** The red lines depict trajectories, while the yellow boxes indicate the locations of the target objects. Besides, we represent the terminal visibility cone in white.

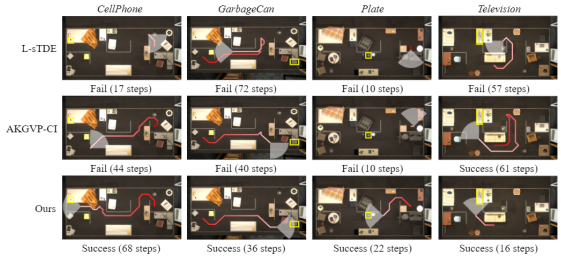


Fig. 6. **Visualization of the agent's trajectories in RoboThor.** The red lines depict trajectories, while the yellow boxes indicate the locations of the target objects. Besides, we represent the terminal visibility cone in white.


Fig. 7. Visualization of the agent's first-person view in AI2-Thor. The target objects found by the agent are marked by yellow boxes.

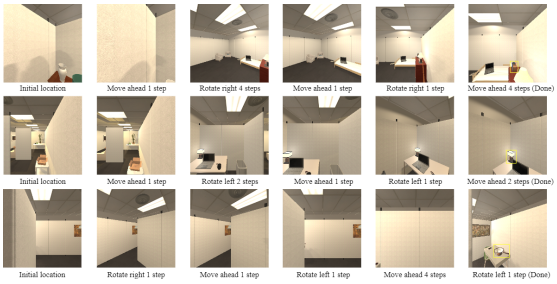


Fig. 8. Visualization of the agent's first-person view in RoboThor. The target objects found by the agent are marked by yellow boxes.

excessive rotations and prematurely issue *Done* action. In contrast, our method enables the agent to better interpret its surroundings and navigate more efficiently, leading to a shorter path to the target object.

For example, as shown in the first column of Fig. 5, when tasked with finding a bowl, both L-sTDE and AKGVP-CI perform redundant rotations, while our method efficiently takes shortcuts. In the fourth column, when finding a remote control, both L-sTDE and AKGVP-CI struggle and get stuck, while our method successfully completes the task with fewer steps. Besides, in the RoboThor environment, as shown in the first and third columns of Fig. 6, when finding a cell phone and a plate in large scenes, both L-sTDE and AKGVP-CI choose incorrect search directions, whereas our method enables the agent to recognize the potential location of the target object and successfully find it.

Fig. 7 and Fig. 8 show extra visualization examples of our proposed method in simulated environments. These figures depict the first-person view of the agent as it navigates unfamiliar scenes to locate various target objects. Fig. 7 showcases examples in AI2-Thor, where the agent is tasked with finding different objects: a garbage can (first row), a kettle (second row), and a light switch (last row). Similarly, Fig. 8 provides examples in the RoboThor environment, where the agent seeks an alarm clock (first row), a desk lamp (second row), and a pot (last row). Our method has good generalization ability in various unfamiliar scenes, even in some big scenes. Besides, for the small objects, such as the light switch, our method still can navigate the agent to the target object in a few steps.

### D. Real-World Deployment

We develop a real-world system to evaluate the transferability of our proposed method in a real-world test environment. As shown in Fig. 9, we deploy our method trained


Fig. 9. Robot and test environment. (a) The Turtlebot4-Lite wheeled robot. (b) The real-world test environment.

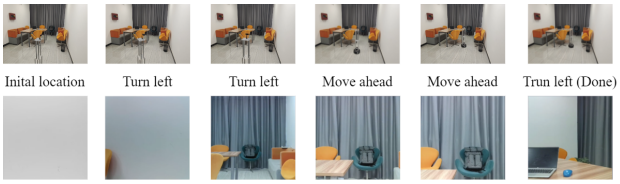


Fig. 10. Visualization of the robot's third-person view and first-person view. The target object is a laptop.

in AI2-Thor on a Turtlebot4-Lite wheeled robot equipped with an OAK-D-Lite camera mounted at a height of 1.0 meters above the ground. The navigation relies solely on RGB camera frames, and DETR [16] pre-trained on the COCO dataset is used for object detection. All algorithms run in real-time on a laptop with an R7-8745H CPU and an NVIDIA GeForce RTX 4050 GPU. A sample navigation trajectory of successfully finding a laptop is visualized in Fig. 10. The robot is initially placed in a location where the laptop is not visible. When the robot turns and observes the desk, it infers that the laptop is likely located there. The deployment demonstrates the effectiveness of our method in transferring navigation capabilities from simulation to real-world environments.

## V. Conclusion

In this paper, we propose a novel object goal navigation framework that leverages co-occurrence embeddings inferred from the LLM to guide object graph reasoning, enabling the agent to reason beyond visual co-occurrence observed in training environments. We also introduce a knowledge distillation mechanism that regularizes the commonsense-guided model with a commonsense-free model, mitigating potential biases from the LLM. Simulated experiments demonstrate that our method significantly improves navigation effectiveness and efficiency. We further deploy our method on a Turtlebot4-Lite wheeled robot to demonstrate its transferability in a real-world environment. Future work may explore the scalability and efficiency of the object graph as the number of objects increases, as well as strategies to narrow the sim-to-real gap across diverse real-world scenarios.

## References

[1] 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," *arXiv preprint arXiv:2006.13171*, 2020.

[2] J. Sun, J. Wu, Z. Ji, and Y.-K. Lai, "A survey of object goal navigation," *IEEE Transactions on Automation Science and Engineering*, vol. 22, pp. 2292–2308, 2024.

[3] T. Zhang, X. Hu, J. Xiao, and G. Zhang, "A survey of visual navigation: From geometry to embodied ai," *Engineering Applications of Artificial Intelligence*, vol. 114, p. 105036, 2022.

[4] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi, "Target-driven visual navigation in indoor scenes using deep reinforcement learning," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 3357–3364.

[5] M. Wortsman, K. Ehsani, M. Rastegari, A. Farhadi, and R. Mottaghi, "Learning to learn how to learn: Self-adaptive visual navigation using meta-learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 6750–6759.

[6] B. Mayo, T. Hazan, and A. Tal, "Visual navigation with spatial attention," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 16 898–16 907.

[7] H. Du, X. Yu, and L. Zheng, "Vtnet: Visual transformer network for object goal navigation," *arXiv preprint arXiv:2105.09447*, 2021.

[8] W. Yang, X. Wang, A. Farhadi, A. Gupta, and R. Mottaghi, "Visual semantic navigation using scene priors," *arXiv preprint arXiv:1810.06543*, 2018.

[9] H. Du, X. Yu, and L. Zheng, "Learning object relation graph and tentative policy for visual navigation," in *European Conference on Computer Vision*. Springer, 2020, pp. 19–34.

[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*, 2021, pp. 15 130–15 140.

[11] R. Dang, Z. Shi, L. Wang, Z. He, C. Liu, and Q. Chen, "Unbiased directed object attention graph for object navigation," in *Proceedings of the 30th ACM International Conference on Multimedia*, 2022, pp. 3617–3627.

[12] S. Zhang, X. Song, W. Li, Y. Bai, X. Yu, and S. Jiang, "Layout-based causal inference for object navigation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 10 792–10 802.

[13] N. Xu, W. Wang, R. Yang, M. Qin, Z. Lin, W. Song, C. Zhang, J. Gu, and C. Li, "Aligning knowledge graph with visual perception for object-goal navigation," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 5214–5220.

[14] L. Chen, Z. He, L. Wang, C. Liu, and Q. Chen, "Temporal scene-object graph learning for object navigation," *IEEE Robotics and Automation Letters*, 2025.

[15] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma *et al.*, "Visual genome: Connecting language and vision using crowdsourced dense image annotations," *International journal of computer vision*, vol. 123, no. 1, pp. 32–73, 2017.

[16] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European conference on computer vision*. Springer, 2020, pp. 213–229.

[17] S. Zhang, X. Song, X. Yu, Y. Bai, X. Guo, W. Li, and S. Jiang, "Hoz++: Versatile hierarchical object-to-zone graph for object navigation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

[18] D. Shah, B. Osiński, S. Levine *et al.*, "Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action," in *Conference on robot learning*. PMLR, 2023, pp. 492–504.

[19] K. Zhou, K. Zheng, C. Pryor, Y. Shen, H. Jin, L. Getoor, and X. E. Wang, "Esc: Exploration with soft commonsense constraints for zero-shot object navigation," in *International Conference on Machine Learning*. PMLR, 2023, pp. 42 829–42 842.

[20] V. S. Dorbala, J. F. Mullen, and D. Manocha, "Can an embodied agent find your "cat-shaped mug"? llm-based zero-shot object navigation," *IEEE Robotics and Automation Letters*, vol. 9, no. 5, pp. 4083–4090, 2023.

[21] D. Shah, M. R. Equi, B. Osiński, F. Xia, B. Ichter, and S. Levine, "Navigation with large language models: Semantic guesswork as a heuristic for planning," in *Conference on Robot Learning*. PMLR, 2023, pp. 2683–2699.

[22] M. Lin, S. Liu, D. Zhang, Y. Chen, Z. Wang, H. Li, and D. Zhao, "Advancing object goal navigation through llm-enhanced object affinities transfer," *arXiv preprint arXiv:2403.09971*, 2024.

[23] J. Loo, Z. Wu, and D. Hsu, "Open scene graphs for open-world object-goal navigation," *arXiv preprint arXiv:2508.04678*, 2025.

[24] L. Sun, A. Kanezaki, G. Caron, and Y. Yoshiyasu, "Enhancing multimodal-input object goal navigation by leveraging large language models for inferring room–object relationship knowledge," *Advanced Engineering Informatics*, vol. 65, p. 103135, 2025.

[25] B. Yu, H. Kasaei, and M. Cao, "L3mvn: Leveraging large language models for visual target navigation," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 3554–3560.

[26] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 28, no. 1, 2014.

[27] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*. PmLR, 2016, pp. 1928–1937.

[28] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 2002.

[29] D. S. Chaplot, R. Salakhutdinov, A. Gupta, and S. Gupta, "Neural topological slam for visual navigation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 12 875–12 884.

[30] D. S. Chaplot, D. Gandhi, S. Gupta, A. Gupta, and R. Salakhutdinov, "Learning to explore using active neural slam," *arXiv preprint arXiv:2004.05155*, 2020.

[31] Y. Liang, B. Chen, and S. Song, "Sscnav: Confidence-aware semantic scene completion for visual semantic navigation," in *2021 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2021, pp. 13 194–13 200.

[32] 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*, 2022, pp. 18 890–18 900.

[33] H. Yoo, Y. Choi, J. Park, and S. Oh, "Commonsense-aware object value graph for object goal navigation," *IEEE Robotics and Automation Letters*, vol. 9, no. 5, pp. 4423–4430, 2024.

[34] 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 *International conference on machine learning*. PmLR, 2021, pp. 8748–8763.

[35] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[36] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in neural information processing systems*, vol. 35, pp. 24 824–24 837, 2022.

[37] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[38] D. Guo, H. Wang, and M. Wang, "Context-aware graph inference with knowledge distillation for visual dialog," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 10, pp. 6056–6073, 2021.

[39] G. Hinton, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[40] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[41] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[42] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, M. Deitke, K. Ehsani, D. Gordon, Y. Zhu *et al.*, "Ai2-thor: An interactive 3d environment for visual ai," *arXiv preprint arXiv:1712.05474*, 2017.

[43] M. Deitke, W. Han, A. Herrasti, A. Kembhavi, E. Kolve, R. Mottaghi, J. Salvador, D. Schwenk, E. VanderBilt, M. Wallingford *et al.*, "Robothor: An open simulation-to-real embodied ai platform," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 3164–3174.

[44] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.