Received 17 May 2024, accepted 9 June 2024, date of publication 13 June 2024, date of current version 20 June 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3413864

RESEARCH ARTICLE

Visual Hindsight Self-Imitation Learning for Interactive Navigation

KIBEOM KIM[1], MOONHOEN LEE[2], MIN WHOO LEE[3], KISUNG SHIN[4], MINSU LEE[5], AND BYOUNG-TAK ZHANG[1,2,3,4,5], (Member, IEEE)

[1]Interdisciplinary Program in Neuroscience, Seoul National University, Seoul 08826, South Korea
[2]Interdisciplinary Program in Cognitive Science, Seoul National University, Seoul 08826, South Korea
[3]Department of Computer Science and Engineering, Seoul National University, Seoul 08826, South Korea
[4]Interdisciplinary Program in Artificial Intelligence, Seoul National University, Seoul 08826, South Korea
[5]AI Institute, Seoul National University (AIIS), Seoul 08826, South Korea

Corresponding authors: Minsu Lee (mslee@bi.snu.ac.kr) and Byoung-Tak Zhang (btzhang@bi.snu.ac.kr)

This work was partly supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) (RS-2021-II212068-AIHub/5%, RS-2021-II211343-GSAI/5%, 2022-0-00951-LBA/20%, 2022-0-00953-PICA/20%), National Research Foundation of Korea (NRF) (RS-2024-00353991-SPARC/20%, RS-2024-00358416/10%, RS-2023-00274280/10%), and Korea Planning & Evaluation Institute of Industrial Technology (KEIT) (RS-2024-00423940/10%) grant funded by the Korean Government.

ABSTRACT Interactive visual navigation tasks, which involve following instructions to reach and interact with specific targets, are challenging not only because successful experiences are very rare but also because complex visual inputs require a substantial number of samples. Previous methods for these tasks often rely on intricately designed dense rewards or the use of expensive expert data for imitation learning. To tackle these challenges, we propose a novel approach, Visual Hindsight Self-Imitation Learning (VHS), which enables re-labeling in vision-based and partially observable environments through Prototypical Goal (PG) embedding. We introduce the PG embeddings, which are derived from experienced goal observations, as opposed to handling instructions as word embeddings. This embedding technique allows the agent to visually reinterpret its unsuccessful attempts, enabling vision-based goal re-labeling and self-imitation from enhanced successful experiences. Experimental results show that VHS outperforms existing techniques in interactive visual navigation tasks, confirming its superior performance, sample efficiency, and generalization.

INDEX TERMS Visual hindsight, interactive navigation, visual reinforcement learning.

I. INTRODUCTION

Embodied AI [1], [2], [3], [4], [5], which focuses on enabling artificial intelligence to sense, understand, and act in a human-like manner, is rapidly evolving and gaining significance. This line of research is increasingly being applied to tasks for service robots such as indoor errands [6], [7], [8], [9], [10]. Many of these tasks can be represented as interactive visual navigation tasks [9], [10], [11], [12], [13], [14], where an agent must be able to perceive and understand its surroundings from first-person perspective and interact with the environment according to given instructions. However, these tasks demand a vast amount of samples for learning [15], [16], [17], considering the high-dimensional visual inputs. In addition, useful feedback is only sparsely provided in such environments, making sophisticated behaviors difficult to attain without prior knowledge.

For these tasks, various solutions have been proposed, such as designing reward functions or imitating the behavior of experts. Among them, studies that utilize dense reward functions [2], [9], [11], [12], [14], [16] require task-specific experts to design such complex rewards. For indoor navigation tasks, these studies [9], [14], [16] consider not only the distance between the goal and the agent but also details like the camera angle and the goal's position within the camera's view. These methods are difficult to generalize, even for similar tasks. An alternative is imitation learning [18], [19], [20], [21], [22], [23], [24], [25] that utilizes expert demonstrations, after which tuning is often applied for learning the environments [26]. This requires building datasets by humans, which is time-consuming and expensive, and each annotator has different criteria, which makes tuning difficult [27].

The associate editor coordinating the review of this manuscript and approving it for publication was Miaohui Wang.

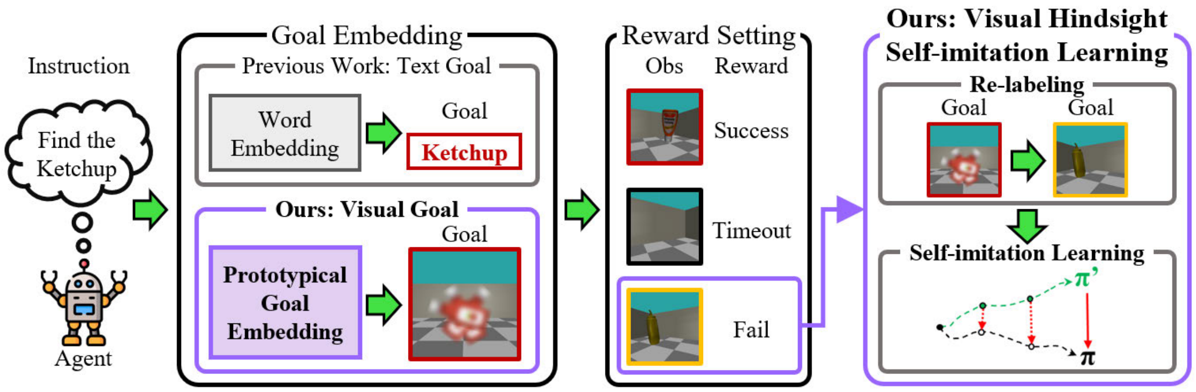


FIGURE 1. Illustration of the learning process for the proposed method. This approach employs prototypical goal embeddings to pursue goals, diverging from traditional word embedding instructions. Our method introduces a strategy for re-labeling goals in failed episodes and facilitates learning through self-imitation. This ultimately benefits the agent in environments with visual inputs and sparse reward settings.

As a way to enhance sample efficiency or exploration without expert data or complex reward shaping, methods based on Hindsight Experience Replay (HER) [28], [29], [30], [31], [32], [33], [34], [35] and learning representations about goals [13], [36] have been proposed. In fully observable environments where numerical coordinates or robot poses are given as goals, re-labeling after failure via HER is simple and feasible. However, in partially observable environments involving first-person view, such information is not directly observed, so it is not straightforward to re-label in a similar manner. Hallucinatory GAN [35] that allows re-labeling visual inputs is limited in that it requires data collection and generative model training in advance, and it does not cope well with dynamically changing visual inputs such as varying backgrounds. Regarding the representation learning methods, GDAN [13] proposes attention networks to learn about goals, while LSA [36] prompts more trials on more difficult-to-reach goals that require further learning. Nonetheless, these works do not adequately utilize experiences, mainly relying on sparse successful experiences.

In this work, we propose the **V**isual **H**indsight **S**elf-Imitation Learning (**VHS**) to address the problems of sample efficiency in instruction-based visual interactive tasks. The pivotal idea behind VHS is 1) to enhance sample efficiency by utilizing failed episodes by re-labeling desired goals or actions to achieved goals or actions and 2) to exploit failed experiences like successful ones via self-imitation, promoting exploration with enriched successful experiences. To enable re-labeling, we devise Prototypical Goal (PG) embedding: instead of using word embedding from instruction [13], [14], [36], we replace the given instruction with prototype features, which is the average of features of the instructed goal as shown in Figure 1. Notably, this embedding is our novel method that allows the usage of visual observations as goals for re-labeling trajectories in hindsight. Our method allows the agent to learn from negative rewards in failed episodes and from self-imitation in re-labeled successful episodes, facilitating efficient learning that does not rely on expert demonstrations. We demonstrate state-of-the-art success rates, sample efficiency, and generalization to complex backgrounds on challenging interactive visual navigation tasks. The visualization of our proposed prototypical goal embedding illustrates its effectiveness in encapsulating the visual characteristics of goals. Additionally, ablation studies reveal that our method outperforms the standard word embedding in interactive visual navigation environments.

Our contribution points in this paper are as follows.

1. We introduce the Prototypical Goal (PG) embedding method for interactive visual navigation tasks. This method embeds goals based on the agent's experience and is more effective than word embedding.

2. We propose novel Visual Hindsight Self-imitation learning (VHS) method which leverages PG embedding to enable goal re-labeling and utilize unsuccessful experience for efficient learning. To the best of our knowledge, our work is the first to make such re-labeling feasible in vision-based, partially observable environments.

3. Experimental results show that our VHS achieves state-of-the-art success rate and sample efficiency in MuJoCo [37] and generalization to complex backgrounds and Maze environment in Miniworld [38]. These results demonstrate that our method is effective in challenging interactive visual navigation tasks.

II. RELATED WORK

A. VISUAL NAVIGATION

Visual navigation tasks, aimed at finding a goal object or location from a first-person perspective, have diversified into specialized categories such as scene-driven navigation, instruction-based visual navigation, and interactive tasks. Among these, scene-driven navigation tasks [17], [23], [39], [40], [41], [42], [43] involve guiding an agent to reach a specific or similar goal in the environment based on a given picture of the goal. Our method is inspired by these studies, particularly the concept of pursuing features from an image or a multi-modal [12] goal, leading us to adopt prototypical goal embedding for the instructions. This enables the extraction and pursuit of desired goals. However, the previous works mentioned above require goal images to be collected beforehand and provided to the agent during execution, which is not feasible in the dynamically changing interactive visual navigation tasks we focus on. In contrast, our approach uniquely leverages the agent's own experiences of goal states during learning, as opposed to using a given target image.

Instruction-based visual navigation is receiving increasing attention [2], [13], [14], [16], [36], [44], [45]. In these studies, the instruction specifies the desired goal among various targets in the environment, with a focus on exploring and recognizing targets [13], [14], [36], [45]. Additionally, interactive tasks [9], [10], [12], [46], [47] require the agent to perform interactions with various goals in interactive indoor environments. However, due to the difficulty of the task, which requires reaching the goals in visual navigation and performing the appropriate interaction in interactive navigation, they rely on pre-trained models for identifying targets and sophisticatedly designed dense rewards for efficient exploration. On the other hand, our approach uses straightforward and intuitively defined rewards of {Success, Timeout, Timestep penalty, and Fail} and can interact with a wide variety of target objects without the need for pre-trained models.

B. SELF-IMITATION LEARNING WITH HER

Self-imitation learning (SIL) [32], [48], [49], [50], [51], [52], [53] is used for exploiting past underestimated experiences to indirectly invoke exploration. SIL involves storing experiences in a replay buffer and focuses on imitating state-action pairs in the replay buffer only when the return in the past trajectory is greater than the agent's estimated value. SIL can be applied to dense reward settings to estimate value and return. SIL offers advantages in enhancing exploration based on previous experiences, eliminating the need for expensive demonstration data. However, SIL might not perform well in environments with sparse rewards. Since it relies on past rewards to guide learning, the lack of frequent rewards can hinder its effectiveness.

In goal-conditioned reinforcement learning, studies have combined SIL and HER to increase the frequency of positive rewards and improve sample efficiency, even in sparse reward settings. ESIL [32] is a self-imitation algorithm that leverages entire re-labeled episodes, whereas original self-imitation learning samples state-action pairs with underestimated value from the experience replay buffer. GRSIL [49] learns self-imitated policies with goal re-labeling and actor policies separately and then combines them to infer the most rewarding actions. These studies are similar to ours in that they use the agent's past experience and re-labeling, but they are limited to environments where the goal coordinates and robot pose are fully observable. Unlike these studies, the method we propose uses features based on the agent's experience for instructions, enabling re-labeling even in vision-based and partially observable environments. As a result, the re-labeled episodes are learned to imitate, yielding sample efficient and effective results.

III. PRELIMINARIES

A. REINFORCEMENT LEARNING

We consider the instruction-based interactive visual navigation tasks, where the agent is given an RGB image from an egocentric camera view and an instruction from the environment. We use the formulation of Partially Observable Markov Decision Process (POMDP) that is augmented with the notion of instruction. In detail, we denote the task as a tuple $(\mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma, \mathcal{I})$, where $\mathcal{S}$ denotes the state space, $\mathcal{O}$ the observation space, $\mathcal{A}$ the action space, $\mathcal{R}$ the reward function, $\mathcal{P}$ the transition probability function, and $\gamma \in [0, 1)$ the discount factor. In the interactive visual navigation tasks that we tackle, the action space is $\mathcal{A} = \mathcal{M} \cup \mathcal{K}$, where $\mathcal{M}$ is the set of actions for agent movements and $\mathcal{K}$ is the set of actions for agent's interaction with goal objects. $\mathcal{I}$ denotes the set of *instructions*, where each instruction $I^{k,x} \in \mathcal{I}$ specifies an interaction $k \in \mathcal{K}$ and a goal $x \in \mathcal{X}$ among possible interactions $\mathcal{K}$ and goal objects $\mathcal{X}$. In each episode, an instruction $I^{k,x}$ specifies with which goal (x) and via which interaction type (k) the agent must interact. As outlined later in Sec. III-C, the reward function is determined based on the given instruction. State-value function in this instruction-based POMDP is $V(s, I^{k,x}) = V(s|k, x) = \mathbb{E}[R_t|S_t = s, X = x, K = k]$, where S_t is state at time t and X, K are goal and interaction type given by the instruction respectively. $R_t = \sum_{t'=t}^{T} \gamma^{(t'-t)} r_{t'}$ denotes the sum of decayed rewards $r_{t'}$ from time step t to terminal step T. The aim of an agent is to find an optimal policy $\pi^* : \mathcal{O} \to \Delta(\mathcal{A})$ that maximizes $\mathbb{E}[R_t]$.

As the base Reinforcement Learning (RL) algorithm, we use Asynchronous Advantage Actor-Critic (A3C) [54]. In this method, the policy gradient for the actor function π_θ and the loss gradient for the critic function V_ϕ are defined as Eq. 1 and 2 respectively:

$$\nabla_\theta \mathcal{L}_{RL} = -\nabla_\theta \log \pi_\theta(a_t|o_t, x, k)(R_t - V_\phi(o_t|x, k)) - \beta \nabla_\theta H(\pi_\theta(\cdot|o_t, x, k)) \tag{1}$$

$$\nabla_\phi \mathcal{L}_{RL} = \nabla_\phi (R_t - V_\phi(o_t|x, k))^2. \tag{2}$$

We note that the critic function $V_\phi : \mathcal{O} \to \mathbb{R}$ serves to estimate the value function V while given only the observation, rather than the actual state.

B. GOAL-AWARE LEARNING

In visual navigation tasks, it is crucial for an agent to recognize goals based on its experiences. We use goal-aware representation learning [13], [36] to train the agent to distinguish between different goals during policy learning. This involves using the SupCon [55] loss, a contrastive learning method that pairs same-labeled data as positives. Additionally, this approach requires datasets with labeled goal observations. When an instruction is successfully executed, the observation at the terminal step is considered a goal observation, and a *(goal observation, desired goal x)* pair is stored in the *Goal Storage* D_g, which serves as the dataset for representation learning. In this setup, the desired goals in the instructions act as labels for the goal observations. Data corresponding to the same goal are treated as positive pairs, while data for different targets are negative pairs. This framework helps train the feature extractor using the Goal-aware SupCon loss function $\mathcal{L}_s$, which is defined as

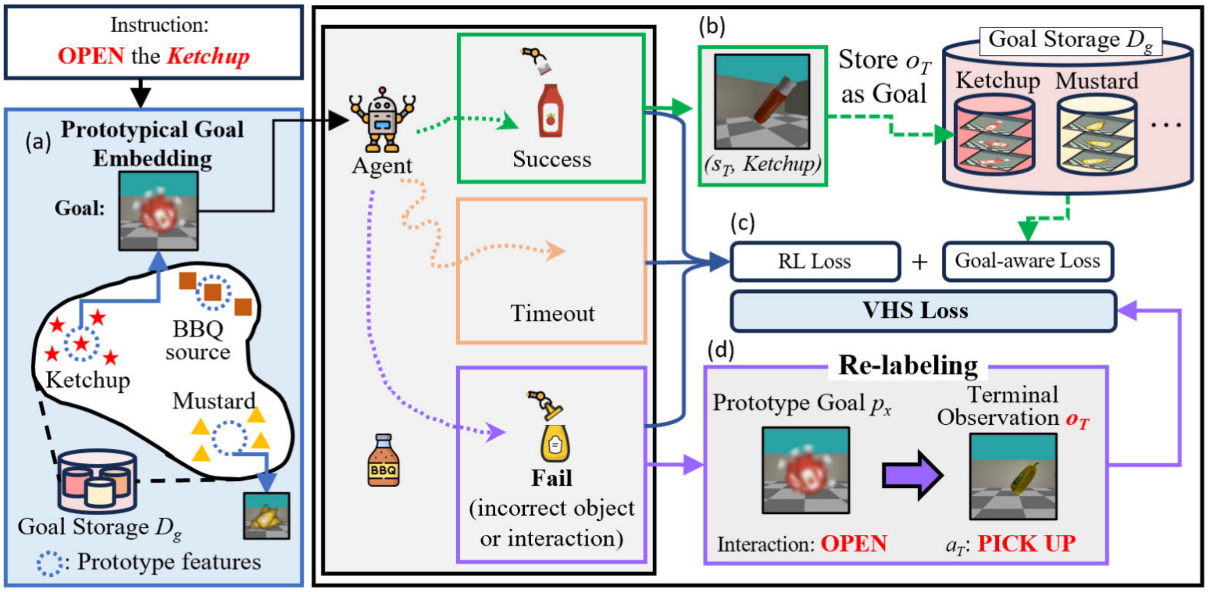


FIGURE 2. The overall architecture. (a) Prototypical Goal (PG) embedding module samples goal observations collected from goal storage and computes prototypical features from the latent representations of those goal observations. (b) If the episode ends successfully, the goal observation is collected by pairing the end-of-episode observation with the desired goal. (c) When the episode ends, the agent updates the RL loss and Goal-aware loss. (d) For each episode that fails, the re-labeling process is run to replace the goal with the last observation of the episode, and then perform the Visual Hindsight Self-Imitation Learning.

follows:

$$\mathcal{L}_s = \sum_{j \in J} \frac{-1}{|P(j)|} \sum_{p \in P(j)} \log \frac{\exp(g_j \cdot g_p/\tau)}{\sum\limits_{h \in J \setminus \{j\}} \exp(g_j \cdot g_h/\tau)} \tag{3}$$

where J is the set of indices of goal states in the batch, and $P(j)$ is the set of all positive pair indices corresponding to the j-th goal state (where j itself is not in $P(j)$). $|P(j)|$ is the cardinality of $P(j)$, $\cdot$ is the dot product operator, g_j is the output of the feature extractor of j-th goal state, and τ is temperature as a hyperparameter.

C. REWARD SETTINGS

In order to perform interactive visual navigation tasks, it is common to design complex rewards based on distance from agent to the goal [2], [9], [11], [14], [16] or observation of agent including goals [12], [14]. To ensure that the agent can perform the task correctly even with a simple and intuitive reward setting, we present the following minimal reward function design.

- **Success**: Given the instruction $I^{k,x}$, if the instructed goal object x is reached correctly and the required interaction k is performed, the agent is judged to be successful and receives a success reward.
- **Timeout**: Timeout penalty is given when the maximum number of steps T is reached.
- **Failure**: If the agent reaches an incorrect goal or executes an incorrect interaction, it is given a failure reward. At this juncture, the agent is not aware of which specific goal was incorrectly reached or which erroneous interaction was performed.
- **Timestep Penalty**: The agent receives a penalty reward for every step to accelerate trial and error.

IV. METHOD

A. INSTRUCTION TO PROTOTYPICAL GOAL EMBEDDING

A crucial aspect of hindsight methods is their ability to redefine the goals of failed episodes. However, in visual navigation tasks [2], [13], [14], [36], [44] with multiple objects or goals, where the instructions or words specify which goal to reach, setting the terminal observation as a new goal poses a practical problem. In general, the tasks specify various goals to be reached as representations within an embedding space, such as word embeddings. Yet, when an agent fails to reach a desired goal, representing the final location of the failed episode in word form is not straightforward, in contrast to tasks where goals are given as coordinates. For instance, if the agent is instructed to navigate and pick up a ketchup but instead picks up a mustard, then re-labeling this terminal observation as "Pick up a mustard" is infeasible without an additional external signal of which object the agent has arrived at.

To address this issue, we introduce the Prototypical Goal (PG) embedding method for instructions, as depicted in Figure 2. This method diverges from the typical approach of word embedding instructions. Instead, it substitutes the instruction input with feature representations of the target x with which the agent is supposed to interact. PG embedding utilizes the goal observations stored during the learning process to replace conventional input instructions with prototype representations. These prototype representations p^x are computed as the average vector of the embedded goal observations associated with each class x as defined by the following equation:

$$p^x = \frac{1}{|D_g^x|} \sum_{o \in D_g^x} f(o). \tag{4}$$

Algorithm 1 VHS With A3C

```
Initialize actor and critic
Initialize representation parameters: $\theta$ and $\phi$
Goal Storage: $\mathcal{D}_g \leftarrow \emptyset$
Global, thread step counter: $T \leftarrow 0, t \leftarrow 0$
repeat
    Environment Reset $t \leftarrow 0$
    Hindsight Episodic Buffer: $\mathcal{D}_f \leftarrow \emptyset$
    Get observation $o_0$, instruction $I^{k,x}$ from environment
    Get Prototypical Goal embedding $p^x$ from Eq. 4
    repeat
        $a_t \sim \pi_\theta(a_t|o_t, p^x, k)$
        Receive reward $r_t$ and next observation $o_{t+1}$
        Store the transition $(o_t, a_t, V_\phi(o_t|p^x, k))$ in $\mathcal{D}_f$
        if Success then
            Collect success states $\mathcal{D}_g \leftarrow \mathcal{D}_g \cup \{(o_t, I^x)\}$
        end if
        $t \leftarrow t + 1$
        $T \leftarrow T + 1$
    until terminal $o_t$ or $t = t_{max}$
    for $i \in \{t-1, \dots, 0\}$ do
        Calculate $\mathcal{L}_{RL}$ with Eq. 1 and 2
    end for
    Calculate Goal-aware loss $\mathcal{L}_s$ from $D_g$ with Eq. 3
    if Fail and Random(0,1) $< \eta$ then
        Re-labeling $p^x$ to $f(o_t)$ and $k$ to $a_t$ in $\mathcal{D}_f$
        Calculate $\mathcal{L}_{VHS}$ from $\mathcal{D}_f$ with Eq. 5
    end if
    Update the parameters $\theta$ and $\phi$ using loss $\mathcal{L}$ in Eq. 6
until $T > T_{max}$
```

In the above equation, $f(\cdot) : \mathcal{O} \rightarrow \mathcal{Z}$ is a convolution neural network that extracts the features of the input observation, $\mathcal{Z}$ is an observation embedding space, and D_g^x is the portion of the goal storage D_g that pertains to goal object x. If there is no goal observation for $I^{k,x}$ in the agent's experience D_g^x, p^x is instead a random vector sampled from $\mathcal{N}(\mathbf{0}, \mathbf{I})$.

This method is similar to prototypical networks for few-shot learning [56], [57], [58] and we use our embedding method to guide the pursuit of a prototype of the desired goal, akin to strategies employed in scene-driven navigation.

B. HINDSIGHT EPISODIC BUFFER

To solve the challenging problem of sparse reward settings, HER learns by replacing agent's unsuccessful experiences with successful ones by re-labeling the goal. Rather than discarding such unsuccessful yet informative feedback, such re-labeling can serve as a textbook for exemplary behavior for the agent, driving its exploration towards potentially successful interactions with various goals. We take this approach and apply it to an interactive visual navigation agent that uses re-labeling to change its experience, as shown in Figure 2. Given an instruction $I^{k,x}$, there are three cases of rewards, outlined in Sec. III-C. Among these, we choose to perform re-labeling when a failure reward is encountered, i.e., when an incorrect goal is reached or an incorrect interaction is performed. In such cases, the observation and action at final time step T can be readily used as the new goal and interaction. Thus, upon receiving failure reward, the goal and interaction are re-labeled from (p^x, k) to $(f(o_T), a_T)$. Note that PG embedding from Sec. IV-A precisely makes such re-labeling of goal possible, by placing both $f(o_T)$ and p^x in the same embedding space $\mathcal{Z}$.

The experience converted by this process, while successful, is a suboptimal path because it is an unintended experience from a suboptimal policy. We describe how to utilize this suboptimal path for learning in Sec. IV-C.

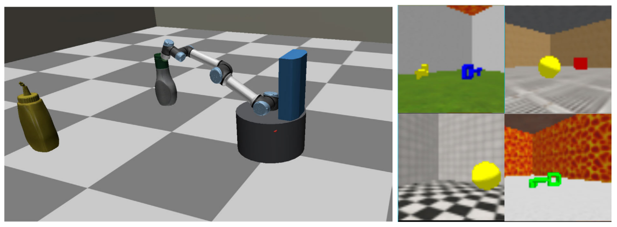


FIGURE 3. Example scenes of our environment, MuJoCo and Miniworld. At each reset, the agent is located at the center of the map, and the objects are placed in a randomized pose. We set up an agent to perform interactive visual navigation by combining a UR5 arm robot and a Fetch mobile robot in MuJoCo. Miniworld experiments with generalization in environments evaluate complex background changes.

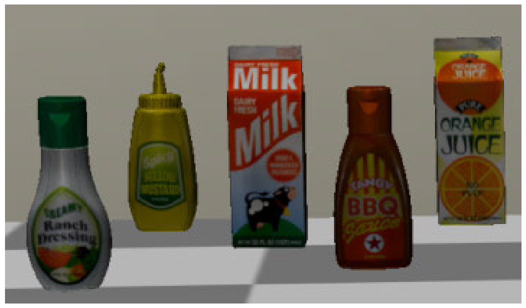


FIGURE 4. Images of the objects used in the MuJoCo experiment. From left to right, the objects are SaladDressing, Mustard, Milk, BBQSauce and OrangeJuice.

C. VISUAL HINDSIGHT SELF-IMITATION LEARNING

We have previously explained an embedding method for re-labeling failed episodes in the vision domain in Sec. IV-A. Using this method, we propose self-imitation for online learning using the newly obtained trajectories. The overall algorithm is outlined in Algorithm 1. First, for each episode, we collect the trajectory $\{(o_t, a_t, V_\phi(o_t|p^x, k))\}_{t=1}^T$ in a buffer $\mathcal{D}_f$. While calculating the RL loss for each episode, if it is a failed episode, we additionally compute the VHS loss according to the reward of the episode as shown below:

$$\mathcal{L}_{VHS} = -\mathbb{E}_{o_i, a_i, v_i \in \mathcal{D}_f} \log \pi_\theta(a_i|o_i, f(o_T), a_T) + \|V_\phi(o_i|f(o_T), a_T) - v_i\|^2. \quad (5)$$

In this way, online learning with hindsight becomes possible. Note that the data $\mathcal{D}_f$ collected by the current agent is not accumulated after the agent learns through behavioral

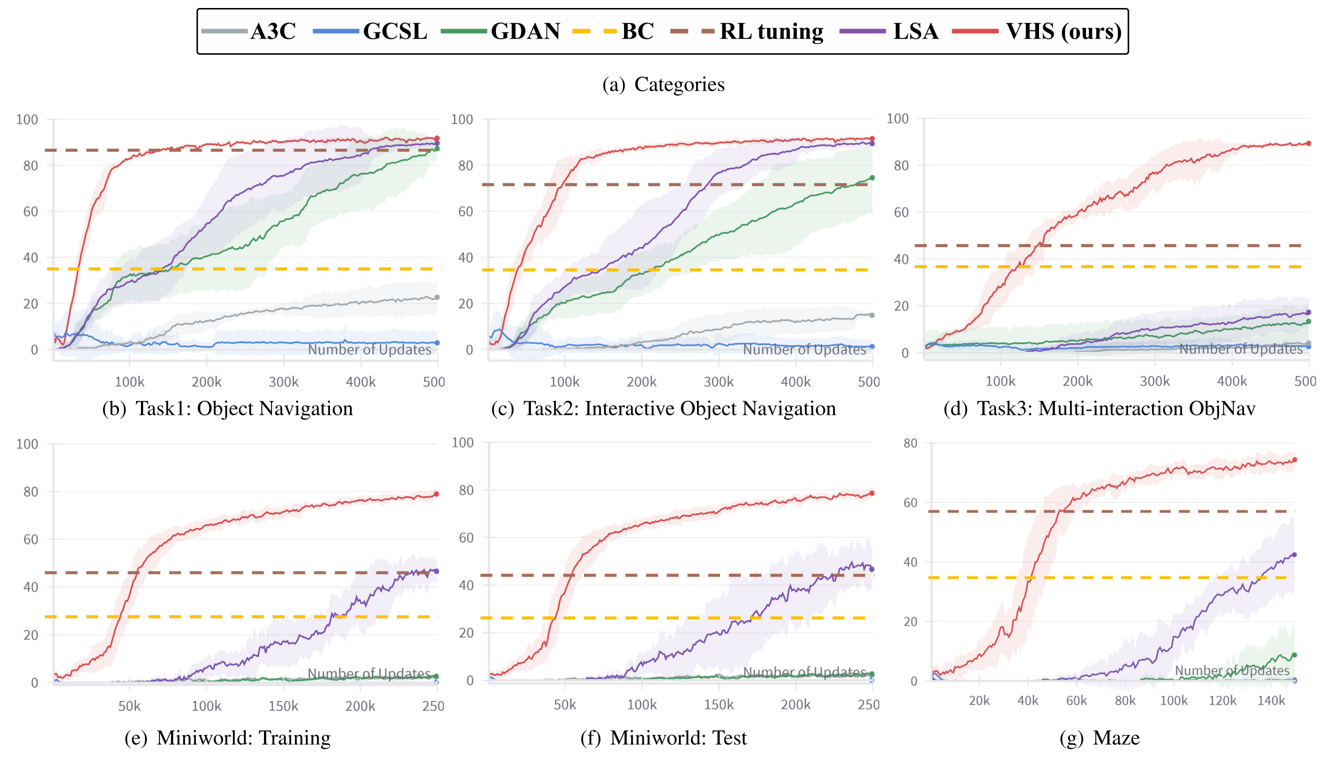


FIGURE 5. **Learning curves for visual navigation tasks in MuJoCo and Miniworld. In all tasks, our method shows rapid improvement and saturation in performance, demonstrating high sample efficiency. Especially in Task3, it shows a significant gap with baselines in task performance. In Miniworld, VHS is superior not only in success rate and sample efficiency but also in generalization. The horizontal axis is the number of updates, and the vertical axis is the success rate. Each curve is produced from 7 trials and indicate bounds as mean ± standard deviation.**

cloning of actions and values, because it is a suboptimal path. We additionally clarify that $\mathcal{D}_f$ is for the purpose of learning the proposed method, VHS loss in Eq. 5, independent of RL loss $\mathcal{L}_{RL}$ that may use its own separate replay buffer.

Instructions of the failed episodes are re-labeled using PG embedding, and observation, action, and value of each time step are used for imitation learning. By generating informative experiences through the hindsight method and self-imitating them, the agent can operate efficiently and effectively with visual inputs, even in hard exploration problems such as interactive navigations with sparse feedback.

Finally, the overall loss function is as follows:

$$\mathcal{L} = \mathcal{L}_{RL} + \alpha\mathcal{L}_s + \beta\mathcal{L}_{VHS}. \tag{6}$$

We also use hyperparameter $\eta \in [0, 1]$, where $\mathcal{L}_{VHS}$ is ignored in Eq. 6 by probability $(1 - \eta)$, in order to mitigate the agent overfitting to suboptimal re-labeled trajectory.

V. EXPERIMENTS

A. EXPERIMENTS SETUP

Through experiments, we aim to evaluate whether our method can efficiently learn interactive visual navigation tasks with sparse reward settings. We set up a fetch mobile robot in the MuJoCo environment [37] where five objects (Mustard, BBQSauce, SaladDressing, OrangeJuice, Milk), which are 3D models from Hope3D dataset [59], are placed at random locations in each episode. To show scalability and generalization, we further experiment with an interactive visual navigation task in Miniworld [38]. This environment comprises seven objects (e.g. RedBox, YellowKey, BlueBox, etc) and is about 2.1 times larger than the MuJoCo environment. In the environment, two tasks have numerous textures applied to the background and one task has walls to form a maze. This allows us to experiment with generalization to unseen textures in a larger, more object-rich, and visually complex environment. At each time step, the agent can choose a movement action from $\mathcal{M}_{mu} = \{MoveForward, TurnLeft, TurnRight\}$ in MuJoCo and from $\mathcal{M}_{mi} = \{MoveForward, MoveBackward, TurnLeft, TurnRight\}$ in Miniworld. An interaction action is from $\mathcal{K}$, where $\mathcal{K}$ depends on the given task as outlined in Sec. V-A1. In case the agent is not at the boundary of any object, an interaction action is neglected, treated as a no-op action. An episode ends upon success/timeout/failure, and agent parameter updates are performed at the end of each episode. The evaluation measures the success rate with 300 episodes. Further details of the experimental details are described in the Appendix A.

1) TASK DETAILS

We set up three tasks to evaluate whether our method learns efficiently with various interaction settings. The agent is located in the center of the room at episode reset. The three tasks are as follows, in order of increasing difficulty.

- **Task1 - Object Navigation**: Success and failure are determined automatically when the agent finds and reaches within a certain boundary of an object. No interaction is required, i.e. $\mathcal{K} = \emptyset$. If the object specified by the instruction is reached, the episode is a success; otherwise, it is a failure.

- **Task2 - Interactive Object Navigation**: Task2 is similar to Task1, except that the agent must perform a single interaction $\mathcal{K} = \{Interaction\}$ with an object to determine success and failure. If an object boundary is reached but no interaction is performed, the episode continues without terminating.
- **Task3 - Multi-interaction ObjNav**: In Task3, the number of interactions increases to three as $\mathcal{K} = \{Interaction1, Interaction2, Interaction3\}$, so the total number of instruction types increases to 15 with five objects. The instruction is given by the environment randomly, and the episode is considered a success if both the object and the interaction are correct, and a failure if either the object or the interaction is incorrect. Compared to Task1 and Task2, Task3 is a challenging task that requires more exploration and is significantly more difficult to success due to the diverse instructions and wider action space.

For experiments in a more diverse environment and setting, we experiment with an interactive visual navigation task in Miniworld.

- **Miniworld - Training**: In training, the task is similar to Task2, with the number of objects increased to 7. This environment is 2.1 times larger in width and height than the MuJoCo environment. It also randomizes the textures on walls, ceiling, and floor among 12 possible textures, providing $12^3 = 1728$ background combinations, to evaluate robustness in a visually complex environment.
- **Miniworld - Test**: We perform experiments to verify the generalization performance with $3^3 = 27$ background combinations using textures that are not learned in training.
- **Maze**: This task is set up with the same size and objects as Miniworld-Training, but instead of changing the textures of the background, it has walls to form a maze. In the maze environment, objects are obscured by walls, so the agent must search efficient paths in order to find the goal and perform interactions.

2) REWARD SETTINGS

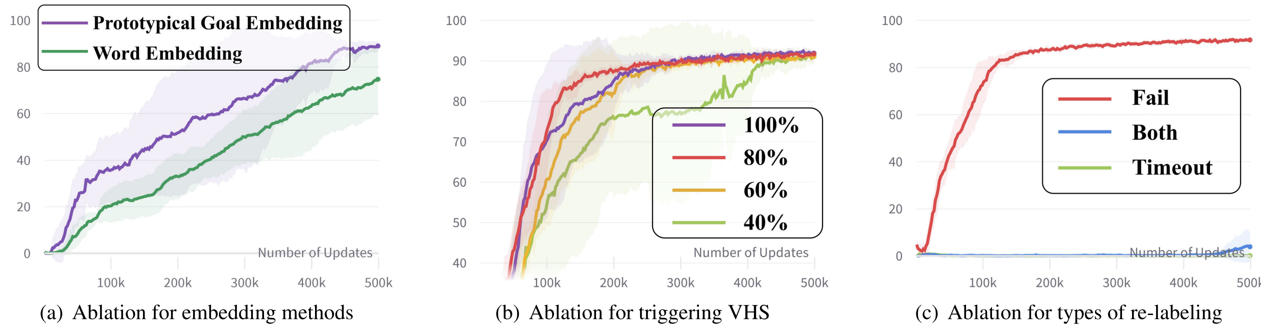
The reward settings for each task are as follows.

- Success: If the goal is reached within a certain range, the episode is considered a success and the agent receives a reward of 10. For tasks that require interaction, if the agent does not perform any interaction $\mathcal{K}$ even after reaching the goal, it only receives a timestep penalty without the episode terminating.
- Failure: If the agent reaches the wrong object or performs the wrong interaction, the episode terminates as a failure. In this case, the agent receives a reward of -1.
- Timeout: The reward for reaching the maximum number of steps T in an episode is -0.1. Timeouts incur a lesser penalty (-0.1) than failure (-1), which initially encourages agents to wait out and avoid haphazardly approaching any goal.
- Timestep penalty: To encourage the agent to explore goals quicker, the environment provides a reward of -0.01 to the agent at every step.

3) BASELINES

We compare our method to the competitive methods below:

- **A3C** (Asynchronous Advantage Actor-Critic) [54]: The most basic reinforcement learning algorithm, it learns asynchronously in multiple environments through multi-processing to accelerate learning speed.
- **BC** (Behavior Cloning) [22]: A method for imitation learning of inputs and outputs, following expert demonstrations. We collected 100K successful trajectories using a trained VHS agent. BC agent is trained on these trajectories.
- **RL tuning** (BC and RL finetuning) [26]: The model pre-trained using BC is further trained by RL finetuning. The method prioritizes tuning the critic for consistent evaluation, and then performs an iterative learning process of finetuning the actor and critic together.
- **GDAN** (Goal-Discriminative Attention Networks) [13]: It learns through cross-entropy loss using goal observations collected for goal-aware learning in visual navigation tasks. It proposes an agent that pursues goal-directed behavior, learns samples efficiently, and further proposes an attention network to maximize the efficiency of the proposed method. The instruction is input by the word embedding method.
- **LSA** (Learning Sampling and Active querying) [36]: This research addresses the phenomenon of experience bias toward easy goals in navigation due to the imbalanced difficulty of reaching each goal object. It proposes sampling for representation learning and an active querying method for collecting experiences, which is state-of-the-art among previous works. Also, this work uses the word embedding method for the instruction input.
- **GCSL** (Goal-Conditioned Self-Supervised Learning) [53]: This method performs efficient learning using only self-imitation learning by re-labeling the next state or the terminal observation as the goal. Because this method does not account for visual navigation tasks, we add our proposed PG embedding to re-label the terminal observation as the goal, and also apply SupCon loss to learn about the goals. The main difference between GCSL and our method is that it performs self-imitation learning without a reward-maximizing RL policy.
- **VHS (Ours)**: In our proposed method, we calculate SupCon loss for learning the goals based on A3C. When learning, we use PG embeddings to pursue the goal, re-label the last observation and interaction in the failed episodes, and learn by self-imitation.



(a) Ablation for embedding methods (b) Ablation for triggering VHS (c) Ablation for types of re-labeling

FIGURE 6. Learning curves of ablation studies. Experiments are performed on Task2 (Interactive Object Navigation). The horizontal axis is the number of updates, and the vertical axis is the success rate.

TABLE 1. Success rate comparison for each algorithm and ablation study for embedding methods on Task1, Task2, Task3, Miniworld and Maze.

Algorithm	**Task1** (%)	**Task2** (%)	**Task3** (%)	**Miniworld-Training** (%)	**Miniworld-Test** (%)	**Maze** (%)
A3C	23.1 ± 7.0	15.5 ± 1.1	4.1 ± 5.7	2.6 ± 1.0	2.8 ± 1.5	2.1 ± 2.2
GCSL	7.6 ± 9.4	8.7 ± 7.4	4.3 ± 0.9	1.2 ± 1.4	1.2 ± 1.3	2.5 ± 1.9
GDAN	87.1 ± 4.6	74.3 ± 15.1	13.3 ± 8.3	2.6 ± 0.6	2.9 ± 0.9	9.2 ± 10.0
BC	36.0 ± 4.9	36.6 ± 4.4	38.0 ± 3.1	28.6 ± 2.0	27.5 ± 1.3	35.5 ± 7.6
RL Tuning	87.0 ± 1.7	71.7 ± 8.9	45.3 ± 1.9	47.5 ± 5.6	43.6 ± 7.2	58.2 ± 4.5
LSA	89.6 ± 1.8	89.9 ± 1.6	16.9 ± 6.5	48.1 ± 5.1	47.1 ± 6.1	42.4 ± 12.2
VHS (Ours)	$\mathbf{92.1 \pm 0.8}$	$\mathbf{92.0 \pm 0.8}$	$\mathbf{89.5 \pm 0.9}$	$\mathbf{78.9 \pm 1.2}$	$\mathbf{76.8 \pm 2.6}$	$\mathbf{75.2 \pm 3.2}$
Word Embedding (GDAN)	87.1 ± 4.6	74.3 ± 15.1	13.3 ± 8.3	2.6 ± 0.6	2.9 ± 0.9	9.2 ± 10.0
PG Embedding (Ours)	90.6 ± 1.7	89.6 ± 2.1	31.8 ± 8.1	8.9 ± 4.6	7.4 ± 4.7	11.7 ± 7.4

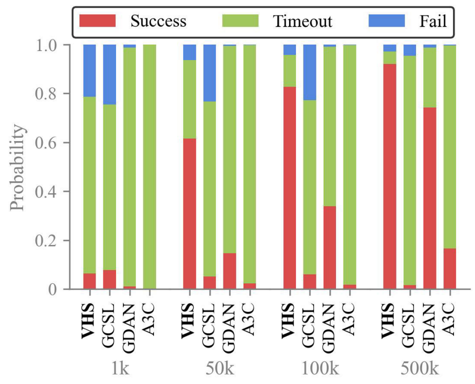


FIGURE 7. Proportion of three reward types. Experiments are performed on Task2. The horizontal axis is the number of updates, and the vertical axis is the probability for each reward type.

We iteratively trained with random seeds at least 7 times, and the details of the hyperparameters used and additional ablations are shown in Sec. VI and Appendix A.

B. EXPERIMENTAL RESULTS

Below, we summarize the results of our experiments with the reward settings and baselines for each environment. The learning curves of the three tasks in the MuJoCo environment and the Training/Test tasks in the Miniworld are shown in Figure 5, and the results are recorded in Table 1.

1) EXPERIMENTAL RESULTS OF MUJOCO

A3C demonstrates marginal progress, with a performance of $23.1 \pm 7.0\%$ in Task1 and negligible improvement in Task3, suggesting that the three tasks are very challenging to learn with a conventional RL algorithm.

GDAN shows learning performance of $87.2 \pm 4.2\%$ and $74.3 \pm 15.1\%$ on the Task1 and Task2 in Figure 5(b) and Figure 5(c), respectively, with slow and steady improvements in learning, indicating that an agent with goal-directed behavior can learn even under sparse reward designs. However, in Figure 5(d) where GDAN achieves $13.3 \pm 8.3\%$ in Task3, the learning curve improves very weakly, suggesting that the goal-directed behavior and attention models alone are not sufficient in this task, which requires a variety of interactions.

In contrast, GCSL underperforms compared to A3C in the first two tasks and fails to exhibit learning across all tasks. This outcome suggests that a self-imitation learning through re-labeling without explicit rewards is inadequate for robust learning.

BC method is less affected by task difficulty because it learns by imitating successful trajectories, and it performs uniformly across all tasks. However, its success rate is somewhat lacking, probably because it only imitates the given expert trajectories. It seems unable to explore and generalize to goal locations that do not occur in the dataset.

RL Tuning finetunes the agent that is pre-trained from BC. This improves upon the BC agent, leading to competitive performance in Task1 and Task2. It also performs better than other baselines in Task3. Nonetheless, we observe a RL Tuning's significant performance gap from our VHS. This may be due to RL Tuning's lack of particular method in encouraging exploration.

LSA is a method that boosts sample efficiency and performance by focusing on one type of instruction at a time. Thus, LSA shows high performance in Task1 and Task2. Nevertheless, it fails to learn well in Task3. This is likely because Task3 involves not only numerous goal objects but also multiple interactions, increasing the number of interaction types from 5 to 15. From the result, we speculate that LSA struggles to scale to a huge variety of instructions.

On the other hand, our proposed method, VHS, shows the highest learning performance as well as the lowest variance in all tasks, converging stably in each iteration. Specifically, VHS exhibits remarkable sample efficiency, achieving state-of-the-art success rates of $\mathbf{92.1 \pm 0.8\%}$, $\mathbf{92.0 \pm 0.8\%}$ and $\mathbf{89.5 \pm 0.9\%}$ on the three tasks, respectively. Notably, in Task3, VHS shows a significant gap compared to the baselines, indicating that our method is robust in sparse reward settings in interactive visual navigation tasks. As depicted in Figure 5(d), where successful trials are exceedingly sparse, our method reaches a saturation point in learning with a high success rate. These results demonstrate the effectiveness of the proposed approach that combines re-labeling with self-imitation learning to facilitate the acquisition of successful experiences via PG embedding.

2) EXPERIMENTAL RESULTS OF MINIWORLD

To evaluate scalability, we use Miniworld environment, which is larger than the MuJoCo environment and has a greater diversity of objects. In addition, the Training and Test environments serve to evaluate robustness to visually complex scenes, as well as generalization to unseen background textures. Maze task poses a more complex environment, contain inner walls that obstruct the agent's view and movement. This serves to evaluate the agent's ability to scale to structurally complex task.

In this task, Figure 5(e) and 5(f) show that most baselines, including A3C and GCSL, as well as GDAN, do not improve their learning curves. On the other hand, BC, which learns from successful experiences, and RL tuning based on BC, have a relatively high success rate and show good generalization even with complex backgrounds. However, the improvement in success rate is limited, which we speculate is due to the limitations of BC and RL tuning, which lack representation learning of the goal and feedback on unsuccessful experiences. In addition, LSA, which increases the experience of goals that need to be trained, also shows a delayed improvement in success rate. This is likely due to the large number of goals and complexity of their backgrounds, which results in intensive training on individual goals but not enough feedback on failed experiences. This proves that sufficient experience with success and feedback on failures are essential to solving the task.

Figure 5(g) depicts the learning curves for the Maze task. A3C and GCSL fail to learn, and GDAN starts to improve late. LSA and RL Tuning, which lack re-labeling, achieve higher performances than the baselines but are still insufficient.

For these complex and difficult tasks, the proposed method has much steeper learning curves with higher success rates, showing rapid saturation. These results support that our method can not only scale well to larger environments with greater diversity of objects and to a more structurally complex maze, but also generalize well to unseen, visually noisy backgrounds. In other words, for difficult tasks with frequent failures, our proposed VHS, which re-labels the experience and learns by self-imitation, is more effective and sample efficient.

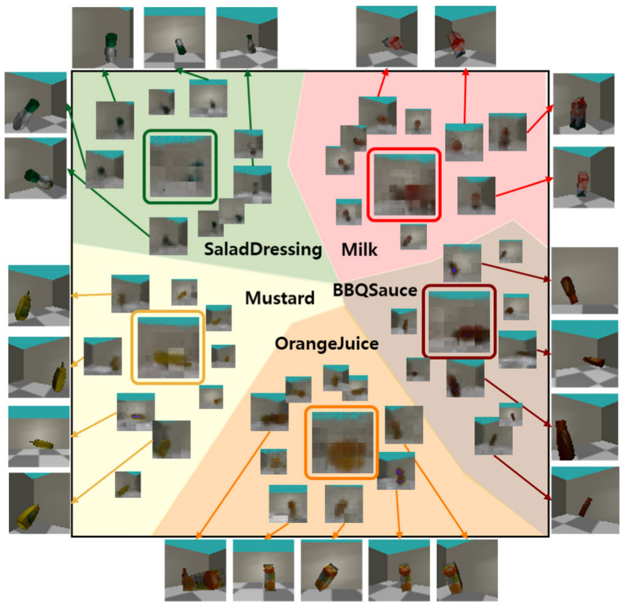


FIGURE 8. Visualization of prototypical goals and embeddings of goal observations. The center of each region shows the prototypical goal embedding of the corresponding object, while the neighboring images visualize data from goal storage with close feature distances and point outward to the source images.

VI. ANALYSES

A. ABLATION STUDIES

1) EMBEDDING METHODS

To evaluate the efficacy of PG embedding, we contrast it with the conventional word embedding technique. The learning curves are shown in Figure 6(a) for Task2 and the success rates are shown in Table 1 for all tasks. The algorithm we used for training is based on SupCon loss $\mathcal{L}_s$ for goal-aware learning with A3C as $\mathcal{L}_{RL}$, applying each embedding method. The learning curves reveal consistent progress for both the word embedding and our PG embedding methods. Regarding the performance improvement of PG embedding compared to word embedding, Table 1 shows that the more difficult the task, the greater the improvement. In Task3, the success rate of word embedding is only 13%, while PG embedding increases it to 32%. Overall, the agent better performs with PG embedding compared to word embeddings.

2) PROBABILITY OF TRIGGERING VHS LOSS

We calculate and update the VHS loss by re-labeling when the agent has failed episodes. As mentioned in Sec. IV-C, we apply the VHS loss $\mathcal{L}_{VHS}$ with probability η to lessen potential overfitting to suboptimal re-labeled trajectories. We perform an ablation study on this probability η by conducting experiments on Task2 with $\eta \in \{40\%, 60\%, 80\%, 100\%\}$. The learning curve for each probability of VHS loss is plotted in Figure 6(b). Overall, the learning curve improves and saturates more rapidly with higher η, indicating that our proposed

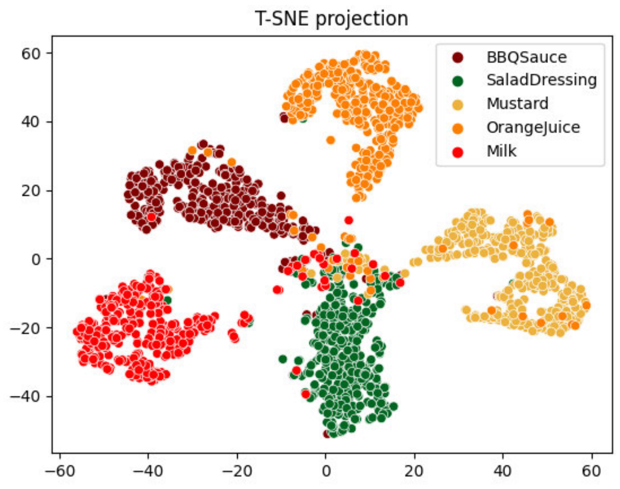


FIGURE 9. Visualization of goal observations using t-SNE. Note the distinct regions for different objects, indicating that the features are well-segmented.

method is effective. However, there is a slight gap between 80% and 100%, with 80% saturating more rapidly and reliably. This suggests that the trajectories used for self-imitation learning are indeed suboptimal paths and that some of the experiences cause noise or overfitting.

3) PROPORTION OF REWARD TYPES

Figure 7 shows the reward distribution across different methods during the learning process. Timeouts are notably more frequent than successes and failures, a trend attributed to the agent's rarity in reaching any goal through random behavior.

A3C consistently records the highest rate of timeouts and the lowest rate of failures, suggesting a tendency towards non-productive trial and error instead of successful task completion. GCSL, occasionally reaching incorrect goals due to PG embedding and SupCon loss, shows prolonged periods of high failure rates. This method, which relies on imitation learning without RL incentives, suggests that its failure to reach a correct goal does not significantly impact its learning outcomes. GDAN predominantly encounters timeouts early in learning but exhibits gradual improvements in goal differentiation over time, leading to reduced timeouts. However, since GDAN does not account for interactions, its learning remains inefficient. In contrast, VHS starts with a higher failure rate that diminishes over time. It enhances learning through self-imitation and penalization of unsuccessful attempts. Consequently, VHS achieves the highest success rates and sample efficiency.

4) ABLATION ON RE-LABELING OF GOALS

For VHS, we propose Prototypical Goal embedding method to re-label the terminal observation as goal, only in the case of failed episodes. For an ablation study on re-labeling methods, we perform additional experiments by re-labeling when episode ends in a failure, when the episode timeout occurs, and when either a failure or a timeout occurs (referred

TABLE 2. Sample efficiency measures in Task2. SRR (lower the better) and SEI (higher the better) are measured with GDAN performance as a reference. "Number of Updates" indicates the number of updates required to reach the reference performance.

Algorithm	Number of Updates	SRR (%) $\downarrow$	SEI (%) $\uparrow$
GDAN (Word embedding)	500,000	100	0
PG embedding (Ours)	345,116	69.0	44.9
LSA	290,972	58.2	71.8
VHS (Ours)	103,002	$\mathbf{20.6}$	$\mathbf{385.4}$

to as *Both*). These experiments are conducted in Task2. For timeout, we re-label the terminal observation as goal when the maximum number of steps $T = 20$ is reached within an episode.

Figure 6(c) shows the learning curves, and we can see that unlike the curve for re-labeling on failure, there is no improvement in performance when learning by re-labeling on timeout, and a slight improvement at the end for re-labeling on "Both." When the agent ends the episode with a timeout, the agent is still approaching the wall without sufficiently having learned to reach the goals, or it is performing unnecessary actions without reaching the goal that requires interaction. For this reason, re-labeling on timeout is not considered appropriate. Additionally, we can see a small improvement for re-labeling on "Both," from which we speculate that re-labeling on timeout is mostly interfering with learning. Thus, our results support that only the proposed re-labeling of failed episodes leads to efficient learning in an interactive visual navigation task.

5) SAMPLE EFFICIENCY MEASURE

To quantitatively measure the sample efficiency, we use the Sample Requirement Ratio (SRR) and Sample Efficiency Improvement (SEI) metrics as proposed in [13]. With algorithm A as the reference, these measures are calculated based on the success rate $X\%$ of this reference algorithm A and the number of updates n_A required for A to reach the corresponding success rate. Then we find the number of updates n_B required to reach the success rate $X\%$ for the algorithm B we want to measure, and calculate $SRR = n_B/n_A$. The SEI is calculated as $(n_A - n_B)/n_B$. Lower SRR and higher SEI indicate better sample efficiency relative to the baseline.

We show the measurements of the sample efficiency in Table 2. We choose GDAN as the reference algorithm and take 74.3% after 500k updates as the criterion in Task2 to measure the efficiency of PG embedding and VHS. The results show that VHS is significantly more sample efficient than GDAN and LSA, with **SRR** of **20.6**% and **SEI** of **385.4**%. In addition, PG embedding has SRR of 69% and SEI of 44.9%, indicating that it also improves sample efficiency compared to word embedding.

B. VISUALIZATION OF PROTOTYPICAL GOALS

We visualize the extracted features and the prototypical goal embeddings for the goals pursued by the agent. To perform the visualization, we add a decoder training by Variational

AutoEncoder (VAE) [60], [61] with the proposed method, where we train the decoder with samples from the goal storage. Then, the prototypical goal embeddings obtained with Eq. 4 are visualized using the decoder.

The visualization of the five objects used in our experiments is shown in Figure 8, where we visualize the PG embeddings in the center of each region. The surrounding images are sampled from the goal storage with close feature distances and visualized in the same way, with arrows pointing to the original images. Since each object is placed in different orientations and positions in the environment, we can see that the visualization of the embedding reflects the shape and color of objects. The surrounding images also reflect the shape and color well. For the goal pursued by the agent, it is shown that these visual aspects of the desired goal can be well-extracted and pursued through PG embedding. Thus, when re-labeling is performed in the failure episode, PG embedding-like features can be set as the goal even for the incorrect goal similar to PG embedding.

C. VISUALIZATION USING T-SNE

About 1,000 goal observations are collected for each class by an agent with uniform random policy, and the features are extracted by the learned VHS and visualized using t-SNE [62] in Figure 9. This figure shows that there are distinct regions for different classes, and the features for each goal are distinguishable once the representations are sufficiently trained. This result supports Figure 8 where the regions are well-segmented by features.

VII. CONCLUSION

In this paper, we present the VHS approach with prototypical goal embedding, designed to enhance the task performance and sample efficiency in interactive visual navigation tasks with sparse rewards. Our comprehensive experimental results reveal that VHS significantly improves success rates and sample efficiency by hindsight visual goal re-labeling of unsuccessful episodes and by boosting deep exploration through leveraging self-imitation with enriched successful episodes. Notably, our PG embedding, crucial for enabling goal re-labeling from visual observations, operates effectively within this framework without reliance on any pre-trained models. This work is the first one to demonstrate the feasibility of using the hindsight experience replay technique in real-time vision-based tasks, eliminating the need for prior data collection. It paves the way for the integration of this approach into a range of related fields.

Moreover, we anticipate that subsequent research will validate the utility of VHS in incremental learning, especially for effectively interacting with new, unseen objects. Our proposed method can be combined with LLMs (Large Language Models) [27], [63], [64], [65] to generalize to more advanced navigation tasks, such as navigation by reasoning the locations of arbitrary goal objects using commonsense knowledge. Our method can also be extended to more general, high-level interactive tasks with VLMs (Vision Language Models) [66], [67], [68], such as having an arm robot prepare a meal through multiple steps of interacting with ingredients. Such extension may lead the future research closer to a more human-like embodied AI.

A current limitation, however, is that the approach has not been evaluated in settings that require a continuous action space. Additionally, failure rewards and timeout rewards are assumed to be distinguishable for the re-labeling method.

APPENDIX A
EXPERIMENTS DETAILS

A. EXPERIMENTS DETAILS

We share the details of the implementation, such as neural network architecture and hyperparameters used in the experiments.

1) IMPLEMENTATION DETAILS

The agent receives a 2-frame-stack of 64×64 RGB images as an input. The feature extractor processes this input and outputs 256 hidden features. The feature extractor is a 4-layered Convolutional Neural Network, and batch normalization is applied to each layer. All convolutional layers have kernel size 3, stride 2 and padding 1, and the output dimension is 256. Afterward, the input image and the PG embedding are processed via gated-attention [69], and the resulting feature vector is fed into a Long Short-Term Memory (LSTM) module, where the output hidden vector and context vector have dimensions of 256. In Task3, which involves various interactions, the interactions provided by the instruction are embedded and concatenated into the features before and after the LSTM. Finally, the action and value are output by feeding the LSTM's output hidden vector into the policy and value function, each composed of a 2-layered Multi-Layer Perceptron (MLP).

2) HYPERPARAMETERS

The hyperparameters used in the experiment are recorded in Table 3. For VHS trigger η, the higher the value, the more the loss calculation is reflected, While we used the η that we found during hyperparameter tuning, we believe that it can be set higher depending on the experimental environment. The sampling size for PG embedding was set according to the hardware used for training, and other values were set similarly or identically to [36].

B. ENVIRONMENT DETAILS

1) MUJOCO ENVIRONMENT

The first-person perspective navigation in the MuJoCo environment we used for our experiments is shown in Figure 3(a). The map we used for our experiments is $3m \times 3m$ in size, and the agent has a stride length of 0.5 m. The maximum number of steps in an episode T is 15 in all MuJoCo tasks. At each episode reset, the agent is located in the center of the map, and the objects are located in random positions within the map boundaries and in random postures. Our agent is

TABLE 3. Hyperparameters used in our experiments.

Parameter Name	Value
Probability of VHS Trigger η	0.8
Samples for PG Embedding	64
Temperature of SupCon τ	0.07
Warmup	150
Batch Size for SupCon	64
SupCon Loss Coefficient η	0.5
Discount γ	0.99
Optimizer	Adam
AMSgrad	True
Learning Rate	1e-4
Clip Gradient Norm	10.0
Entropy Coefficient	0.01
Number of Training Processes	5
Number of Test Processes	1
Backpropagation Through Time	End of Episode
Non-Linearity	ReLU

a combination of a fetch mobile robot and a UR5 arm and is set up to perform the task of approaching and interacting with objects. Our environment is characterized by the need to perform the task in a sparse reward setting, without a pre-trained model such as object detection or recognition. The five objects that need to be reached are illustrated in Figure 4, all of which are placed in random postures, including orientation. All of the objects are easily encountered in real-world environments.

2) MINIWORLD ENVIRONMENT

The Miniworld environment, which implements an interactive visual navigation task to experiment with generalization and scalability, is shown in Figure 3(b). The environment we used is 9$m \times 9$ m in size, and the agent has a stride length of 0.7 m. The maximum number of steps in an episode T is 50. The number of steps required from the left end of the environment to the right end is 13 steps, compared to 6 steps in the MuJoCo environment, so this environment is 2.1 times larger when map size and stride width are considered together. In each episode, like the MuJoCo environment, the agent is located in the center of the environment, and objects are placed in random locations. The objects and textures used from those provided by the environment are shown below:

- Objects: RedBox, GreenBox, YellowKey, GreenKey, BlueKey, PurpleBall, YellowBall.
- Seen textures: brick_wall, airduct_grate, asphalt, cardboard, ceiling_tile_noborder, ceiling_tiles, cinder_blocks, concrete, concrete_tiles, floor_tiles_bw, grass, lava.
- Unseen textures: marble, metal_grill, picket_fence.

C. EXPERIMENTAL COMPLEXITY

We perform the experiments in the machine with the following specifications.

- CPU: AMD Threadripper 3970X (32 cores)
- RAM: 256 GB
- GPU: RTX 3090 × 2

The training time and RAM spent on the Miniworld-Training experiment using this machine is shown below. (Note that we do not aggregate RAM usage for the learning environment).

- A3C: 7.5 hours, 13GB
- GCSL: 18.5 hours, 13.7GB
- GDAN: 8.2 hours, 13GB
- BC: 36 hours, 23.4GB
- RL Tuning: BC + 4.5 hours, 21.7GB
- LSA: 9.2 hours, 13GB
- VHS (Ours): 9.8 hours, 13.7GB

Both the memory usage and the training time of VHS is increased marginally compared to the other baselines. We equalized the number of updates for comparison with all baselines, but we expect our method to take less than half the time in practice compared to baselines due to early convergence with high sample efficiency.

Furthermore, our research enables online learning in vision-based tasks using re-labeling. This work can be extended by collecting, sharing, and updating experiences with large-scale multi-agent systems [70], [71], recognizing and processing a wider variety of objects. Combined with anomaly detection methods [72] for event-triggering, the agents may proactively respond to unseen or rewarding states. Such extension may allow our method to be applied in the real world.

REFERENCES

[1] J. Duan, S. Yu, H. L. Tan, H. Zhu, and C. Tan, "A survey of embodied AI: From simulators to research tasks," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 6, no. 2, pp. 230–244, Apr. 2022.

[2] 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 *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9338–9346.

[3] S. Franklin, "Autonomous agents as embodied AI," *Cybern. Syst.*, vol. 28, no. 6, pp. 499–520, Sep. 1997.

[4] F. Xia, A. R. Zamir, Z. He, A. Sax, J. Malik, and S. Savarese, "Gibson Env: Real-world perception for embodied agents," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 9068–9079.

[5] A. Das, S. Datta, G. Gkioxari, S. Lee, D. Parikh, and D. Batra, "Embodied question answering," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 2135–213509.

[6] J.-B. Yi and S.-J. Yi, "Mobile manipulation for the HSR intelligent home service robot," in *Proc. 16th Int. Conf. Ubiquitous Robots (UR)*, Jun. 2019, pp. 169–173.

[7] B. Ramalingam, J. Yin, M. Rajesh Elara, Y. K. Tamilselvam, M. Mohan Rayguru, M. A. V. J. Muthugala, and B. Félix Gómez, "A human support robot for the cleaning and maintenance of door handles using a deep-learning framework," *Sensors*, vol. 20, no. 12, p. 3543, Jun. 2020.

[8] C.-Y. Lee, H. Lee, I. Hwang, and B.-T. Zhang, "Visual perception framework for an intelligent mobile robot," in *Proc. 17th Int. Conf. Ubiquitous Robots (UR)*, Jun. 2020, pp. 612–616.

[9] K.-H. Zeng, L. Weihs, A. Farhadi, and R. Mottaghi, "Pushing it out of the way: Interactive visual navigation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 9863–9872.

[10] L. Weihs, M. Deitke, A. Kembhavi, and R. Mottaghi, "Visual room rearrangement," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 5918–5927.

[11] 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 *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 6743–6752.

[12] Z. Al-Halah, S. K. Ramakrishnan, and K. Grauman, "Zero experience required: Plug & play modular transfer learning for semantic visual navigation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 17010–17020.

[13] K. Kim, M. W. Lee, Y. Kim, J. Ryu, M. Lee, and B.-T. Zhang, "Goal-aware cross-entropy for multi-target reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 2783–2795.

[14] Y. Wu, Y. Wu, G. Gkioxari, and Y. Tian, "Building generalizable agents with a realistic and rich 3D environment," 2018, *arXiv:1801.02209*.

[15] E. Wijmans, A. Kadian, A. Morcos, S. Lee, I. Essa, D. Parikh, M. Savva, and D. Batra, "DD-PPO: Learning near-perfect pointgoal navigators from 2.5 billion frames," in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–12.

[16] O. Maksymets, V. Cartillier, A. Gokaslan, E. Wijmans, W. Galuba, S. Lee, and D. Batra, "THDA: Treasure hunt data augmentation for semantic navigation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 15354–15363.

[17] L. Mezghan, S. Sukhbaatar, T. Lavril, O. Maksymets, D. Batra, P. Bojanowski, and K. Alahari, "Memory-augmented reinforcement learning for image-goal navigation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2022, pp. 3316–3323.

[18] W. Sun, J. Andrew Bagnell, and B. Boots, "Truncated horizon policy search: Combining reinforcement learning imitation learning," 2018, *arXiv:1805.11240*.

[19] Z. Cheng, L. Shen, and D. Tao, "Off-policy imitation learning from visual inputs," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2023, pp. 12402–12413.

[20] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Comput. Surveys*, vol. 50, no. 2, pp. 1–35, Mar. 2018.

[21] J. Ho and S. Ermon, "Generative adversarial imitation learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016, pp. 1–6.

[22] S. Schaal, "Is imitation learning the route to humanoid robots?" *Trends Cognit. Sci.*, vol. 3, no. 6, pp. 233–242, Jun. 1999.

[23] Q. Fang, X. Xu, X. Wang, and Y. Zeng, "Target-driven visual navigation in indoor scenes using reinforcement learning and imitation learning," *CAAI Trans. Intell. Technol.*, vol. 7, no. 2, pp. 167–176, Jun. 2022.

[24] R. K. Thakur, M.-N. S. Sunbeam, V. G. Goecks, E. Novoseller, R. Bera, V. J. Lawhern, G. M. Gremillion, J. Valasek, and N. R. Waytowich, "Imitation learning with human eye gaze via multi-objective prediction," 2023, *arXiv:2102.13008*.

[25] H. Karnan, G. Warnell, X. Xiao, and P. Stone, "VOILA: Visual-observation-only imitation learning for autonomous navigation," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2022, pp. 2497–2503.

[26] R. Ramrakhya, D. Batra, E. Wijmans, and A. Das, "PIRLNav: Pretraining with imitation and RL finetuning for ObjectNav," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 1–13.

[27] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. Christiano, J. Leike, and R. Lowe, "Training language models to follow instructions with human feedback," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2022, pp. 27730–27744.

[28] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. Pieter Abbeel, and W. Zaremba, "Hindsight experience replay," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–15.

[29] T. Dai, H. Liu, K. Arulkumaran, G. Ren, and A. A. Bharath, "Diversity-based trajectory and goal selection with hindsight experience replay," in *Proc. 18th Pacific Rim Int. Conf. Artif. Intell.*, 2021, pp. 32–45.

[30] M. Fang, T. Zhou, Y. Du, L. Han, and Z. Zhang, "Curriculum-guided hindsight experience replay," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–20.

[31] B. Manela and A. Biess, "Bias-reduced hindsight experience replay with virtual goal prioritization," *Neurocomputing*, vol. 451, pp. 305–315, Sep. 2021.

[32] T. Dai, H. Liu, and A. Anthony Bharath, "Episodic self-imitation learning with hindsight," *Electronics*, vol. 9, no. 10, p. 1742, Oct. 2020.

[33] M. Fang, C. Zhou, B. Shi, B. Gong, J. Xu, and T. Zhang, "Dher: Hindsight experience replay for dynamic goals," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–19.

[34] Y. Tang and A. Kucukelbir, "Hindsight expectation maximization for goal-conditioned reinforcement learning," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2021, pp. 2863–2871.

[35] H. Sahni, T. Buckley, P. Abbeel, and I. Kuzovkin, "Addressing sample complexity in visual tasks using her and hallucinatory gans," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–16.

[36] K. Kim, H. Lee, M. Whoo Lee, M. Lee, M. Lee, and B.-T. Zhang, "L-SA: Learning under-explored targets in multi-target reinforcement learning," 2023, *arXiv:2305.13741*.

[37] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2012, pp. 5026–5033.

[38] M. Chevalier-Boisvert, B. Dai, M. Towers, R. de Lazcano, L. Willems, S. Lahlou, S. Pal, P. Samuel Castro, and J. Terry, "Minigrid miniworld: Modular customizable reinforcement learning environments for goal-oriented tasks," 2023, *arXiv:2306.13831*.

[39] 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 *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 3357–3364.

[40] Q. Wu, X. Gong, K. Xu, D. Manocha, J. Dong, and J. Wang, "Towards target-driven visual navigation in indoor scenes via generative imitation learning," *IEEE Robot. Autom. Lett.*, vol. 6, no. 1, pp. 175–182, Jan. 2021.

[41] Y. Lyu, Y. Shi, and X. Zhang, "Improving target-driven visual navigation with attention on 3D spatial relationships," *Neural Process. Lett.*, vol. 54, no. 5, pp. 3979–3998, Oct. 2022.

[42] A. Devo, G. Mezzetti, G. Costante, M. L. Fravolini, and P. Valigi, "Towards generalization in target-driven visual navigation by using deep reinforcement learning," *IEEE Trans. Robot.*, vol. 36, no. 5, pp. 1546–1561, Oct. 2020.

[43] X. Ruan, P. Li, X. Zhu, and P. Liu, "A target-driven visual navigation method based on intrinsic motivation exploration and space topological cognition," *Sci. Rep.*, vol. 12, no. 1, p. 3462, Mar. 2022.

[44] D. S. Chaplot, D. Gandhi, A. Gupta, and R. R. Salakhutdinov, "Object goal navigation using goal-oriented semantic exploration," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2020, pp. 4247–4258.

[45] H. Du, L. Li, Z. Huang, and X. Yu, "Object-goal visual navigation via effective exploration of relations among historical navigation states," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 2563–2573.

[46] F. Xiang, Y. Qin, K. Mo, Y. Xia, H. Zhu, F. Liu, M. Liu, H. Jiang, Y. Yuan, H. Wang, L. Yi, A. X. Chang, L. J. Guibas, and H. Su, "SAPIEN: A simulated part-based interactive environment," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11094–11104.

[47] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, M. Deitke, K. Ehsani, D. Gordon, Y. Zhu, A. Kembhavi, A. Gupta, and A. Farhadi, "AI2-THOR: An interactive 3D environment for visual AI," 2017, *arXiv:1712.05474*.

[48] J. Oh, Y. Guo, S. Singh, and H. Lee, "Self-imitation learning," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 3878–3887.

[49] Y. Li, Y. Wang, and X. Tan, *Self-Imitation Guided High-Efficient Goal-Conditioned Reinforcement Learning*, document SSRN 4419852, 2023.

[50] S. Luo, H. Kasaei, and L. Schomaker, "Self-imitation learning by planning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 4823–4829.

[51] Y. Tang, "Self-imitation learning via generalized lower bound q-learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 13964–13975.

[52] G. Pshikhachev, D. Ivanov, V. Egorov, and A. Shpilman, "Self-imitation learning from demonstrations," 2022, *arXiv:2203.10905*.

[53] D. Ghosh, A. Gupta, A. Reddy, J. Fu, C. M. Devin, B. Eysenbach, and S. Levine, "Learning to reach goals via iterated supervised learning," in *Proc. Int. Conf. Learn. Represent.*, 2020, pp. 1–25.

[54] 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 *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.

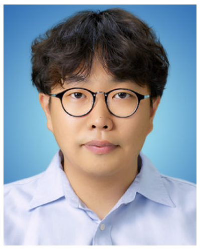
[55] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, "Supervised contrastive learning," in *Proc. NIPS*, 2020, pp. 18661–18673.

[56] J. Wang and Y. Zhai, "Prototypical Siamese networks for few-shot learning," in *Proc. IEEE 10th Int. Conf. Electron. Inf. Emergency Commun. (ICEIEC)*, Jul. 2020, pp. 178–181.

[57] Z. Ji, X. Chai, Y. Yu, Y. Pang, and Z. Zhang, "Improved prototypical networks for few-shot learning," *Pattern Recognit. Lett.*, vol. 140, pp. 81–87, Dec. 2020.

[58] F. Pahde, M. Puscas, T. Klein, and M. Nabi, "Multimodal prototypical networks for few-shot learning," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2021, pp. 2643–2652.
[59] S. Tyree, J. Tremblay, T. To, J. Cheng, T. Mosier, J. Smith, and S. Birchfield, "6-DoF pose estimation of household objects for robotic manipulation: An accessible dataset and benchmark," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2022, pp. 13081–13088.
[60] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," 2013, *arXiv:1312.6114*.
[61] D. P. Kingma and M. Welling, "An introduction to variational autoencoders," *Found. Trends Mach. Learn.*, vol. 12, no. 4, pp. 307–392, 2019.
[62] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 11, pp. 1–26, 2008.
[63] H. Touvron et al., "Llama 2: Open foundation and fine-tuned chat models," 2023, *arXiv:2307.09288*.
[64] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.
[65] W. Xin Zhao et al., "A survey of large language models," 2023, *arXiv:2303.18223*.
[66] T. Brooks, B. Peebles, C. Holmes, W. DePue, Y. Guo, L. Jing, D. Schnurr, J. Taylor, T. Luhman, E. Luhman, C. Ng, R. Wang, and A. Ramesh. (2024). *Video Generation Models as World Simulators*. [Online]. Available: https://openai.com/research/video-generation-models-as-world-simulators
[67] G. Team et al., "Gemini: A family of highly capable multimodal models," 2023, *arXiv:2312.11805*.
[68] H. Liu, C. Li, Q. Wu, and Y. J. Lee, "Visual instruction tuning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 36, 2024, pp. 1–5.
[69] D. S. Chaplot, K. M. Sathyendra, R. K. Pasumarthi, D. Rajagopal, and R. Salakhutdinov, "Gated-attention architectures for task-oriented language grounding," in *Proc. AAAI Conf. Artif. Intell.*, 2018, vol. 32, no. 1, pp. 1–15.
[70] C. Yu, X. Yang, J. Gao, H. Yang, Y. Wang, and Y. Wu, "Learning efficient multi-agent cooperative visual exploration," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 497–515.
[71] H. Wang, W. Wang, X. Zhu, J. Dai, and L. Wang, "Collaborative visual navigation," 2021, *arXiv:2107.01151*.
[72] S. Aberkane and M. Elarbi-Boudihir, "Deep reinforcement learning-based anomaly detection for video surveillance," *Informatica*, vol. 46, no. 2, pp. 1–34, Jun. 2022.

KIBEOM KIM received the B.E. degree in computer engineering from Sejong University, Seoul, South Korea, in 2017. He is currently pursuing the Ph.D. degree with the Interdisciplinary Program in Neuroscience, Seoul National University, Seoul. His research interests include reinforcement learning, embodied AI, and robot learning.



MOONHOEN LEE received the B.S. degree in computer science and the M.S. degree in engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2005 and 2007, respectively. He is currently pursuing the Ph.D. degree with the Interdisciplinary Program in Cognitive Science, Seoul National University, Seoul, South Korea. His research interest includes reinforcement learning.

MIN WHOO LEE received the B.Eng. degree in computer science and engineering from Seoul National University, South Korea, in 2021, where he is currently pursuing the Ph.D. degree in computer science and engineering. His research interests include out-of-distribution generalization and multi-agent reinforcement learning.

KISUNG SHIN received the B.E. degree in computer engineering from Kyung Hee University, South Korea, in 2020, and the M.S. degree in brain and cognitive engineering from Korea University, South Korea, in 2023. He is currently pursuing the Ph.D. degree with the Interdisciplinary Program in Artificial Intelligence, Seoul National University, South Korea. His research interests include reinforcement learning, embodied artificial intelligence, and brain-inspired artificial intelligence.

MINSU LEE received the B.S. degree in mathematics and the M.S. and Ph.D. degrees in computer science and engineering from Ewha Womans University, South Korea. Currently, she is a Research Professor with the AI Institute, Seoul National University. Prior to this, she was a Research Professor with Ewha Womans University and a Visiting Scholar with Indiana University, USA. Her research interests include active learning and machine learning techniques for video understanding and real-world reinforcement learning applications.

BYOUNG-TAK ZHANG (Member, IEEE) received the B.S. and M.S. degrees in computer science and engineering from Seoul National University, South Korea, in 1986 and 1988, respectively, and the Ph.D. degree in computer science from the University of Bonn, Germany, in 1992. He is currently a POSCO Chair Professor of computer science and engineering with Seoul National University (SNU) and the Director of the SNU Artificial Intelligence Institute. He was the President of Korean Society for Artificial Intelligence, from 2010 to 2013, and Korean Society for Cognitive Science, from 2016 to 2017.