# Shaping embodied agent behavior with activity-context priors from egocentric video

**Tushar Nagarajan**
UT Austin and Facebook AI Research
tushar.nagarajan@utexas.edu

**Kristen Grauman**
UT Austin and Facebook AI Research
grauman@fb.com

## Abstract

Complex physical tasks entail a sequence of object interactions, each with its own preconditions—which can be difficult for robotic agents to learn efficiently solely through their own experience. We introduce an approach to discover *activity-context* priors from in-the-wild egocentric video captured with human worn cameras. For a given object, an activity-context prior represents the set of other compatible objects that are required for activities to succeed (e.g., a knife and cutting board brought together with a tomato are conducive to *cutting*). We encode our video-based prior as an auxiliary reward function that encourages an agent to bring compatible objects together before attempting an interaction. In this way, our model translates everyday human experience into embodied agent skills. We demonstrate our idea using egocentric EPIC-Kitchens video of people performing unscripted kitchen activities to benefit virtual household robotic agents performing various complex tasks in AI2-iTHOR, significantly accelerating agent learning. Project page: http://vision.cs.utexas.edu/projects/ego-rewards/

## 1 Introduction

Embodied AI agents that are capable of moving around and interacting with objects in human spaces promise important practical applications for home service robots, ranging from agents that can search for misplaced items, to agents that can cook entire meals. The pursuit of such agents has driven exciting new research in visual semantic planning [68], instruction following [58], and object rearrangement [3, 29], typically supported by advanced simulators [45, 32, 57, 22] where policies may be learned quickly and safely before potentially transferring to real robots. In such tasks, an agent aims to perform a sequence of actions that will transform the visual environment from an initial state to a goal state. This in turn requires jointly learning behaviors for both *navigation*, to move from one place to another, and object *interaction*, to manipulate objects and modify the environment (e.g., pick-up objects, use tools and objects together, turn-on lights).

A key challenge is that changing the state of the environment involves context-sensitive actions that depend on both the agent and environment's current state—what the agent is holding, what other objects are present nearby, and what their properties are. For example, to wash a plate, a plate must be in the sink *before* the agent toggles-on the faucet; to *slice an apple* the agent must first be holding a knife, and the apple must not already be sliced. Understanding these conditions is critical for efficient learning and planning: the agent must first bring the environment into the proper precondition state before attempting to perform a given activity with objects. We refer to these states as *activity-contexts*.

Despite its importance, current approaches do not explicitly model the activity-context. Pure reinforcement learning (RL) approaches directly search for goal states without considering preconditions. This requires a large number of trials for the agent to chance upon suitable configurations, and leads to poor sample efficiency or learning failures [68, 58]. Instead, most methods resort to collecting expert demonstrations to train imitation learning (IL) agents and optionally finetune with

35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.

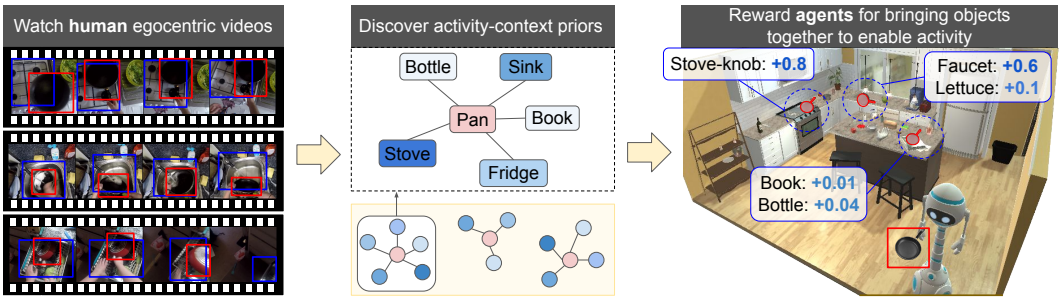


Figure 1: **Main idea. Left and middle panel:** We discover *activity-contexts* for objects directly from egocentric video of human activity. A given object's activity-context goes beyond "what objects are found together" to capture the likelihood that each other object in the environment participates in activities involving it (i.e., "what objects together enable action"). **Right panel:** Our approach guides agents to bring compatible objects—objects with high likelihood—together to enable activities. For example, bringing a pan to the sink increases the value of faucet interactions, but bringing it to the table has little effect on interactions with a book.

RL [68, 58, 30, 15]. While demonstrations may implicitly reveal activity-contexts, they are cumbersome to collect, requiring expert teleoperation, often using specialized hardware (VR, MoCap) in artificial lab settings [67, 48, 27], and need to be collected independently for each new task.

Rather than solicit demonstrations, we propose to learn activity-contexts from real-world, egocentric video of people performing daily life activities. Humans understand activities from years of experience, and can effortlessly bring even a novel environment to new, appropriate configurations for interaction-heavy tasks. Egocentric or "first-person" video recorded with a wearable camera puts actions and objects manipulated by a person at the forefront, offering an immediate window of this expertise in action in its natural habitat.

We present a reinforcement learning approach that infers activity-context conditions directly from people's first-hand experience and transfers them to embodied agents to improve interaction policy learning. Specifically, we 1) train visual models to detect how humans *prepare* their environment for activities from egocentric video, and 2) develop a novel auxiliary reward function that encourages agents to seek out similar activity-context states. For example, by observing that people frequently carry pans to sinks (to clean them) or stoves (to cook their contents), our model rewards agents for prioritizing interactions with faucets or stove-knobs when pots or pans are nearby. As a result, this incentivizes agents to transport relevant objects to compatible locations *before* attempting interactions, which accelerates learning. See Fig. 1.

Importantly, our goal is not direct imitation. Our insight is that while humans and embodied agents have very different action spaces and bodies, they operate in similar environments where the underlying conditions about what state the environment must be in before trying to modify it are strongly aligned. Our goal is thus to guide an agent's exploratory interactions towards these potential progress-enabling states as it learns a new task. Moreover, because our training videos are passively collected by human camera-wearers and capture a wide array of daily actions, they help build a general visual prior for human activity-contexts, while side-stepping the heavy requirements of collecting IL demonstrations for each individual task of interest.

Our experiments demonstrate the value of learning from egocentric videos of *humans* (in EPIC-Kitchens [13]) to train visual semantic planning *agents* (in AI2-iTHOR [32]). Our video model relates objects based on goal-oriented actions (e.g., knives used to cut potatoes) rather than spatial co-occurrences (e.g., knives are found near spoons) [65, 11, 47, 64] or semantic similarity (e.g., potatoes are like tomatoes) [65]. Our agents outperform strong exploration methods and state-of-the-art embodied policies on multi-step interaction tasks (e.g., storing cutlery, cleaning dishes), improving absolute success rates by up to 12% on the most complex tasks. Our approach learns policies faster, generalizes to unseen environments, and greatly improves success rates on difficult instances.

## 2 Related Work

**Interaction in 3D environments** Recent embodied AI work leverages simulated environments [45, 51, 57, 32, 22, 57] to build agents for interaction-heavy tasks like visual semantic planning [68], interactive question answering [25], instruction following [58], and object rearrangement [3, 30].

Prior approaches use imitation learning (IL) based pre-training to improve sample efficiency, but at the expense of collecting expert in-domain demonstrations individually for each task [68, 15, 30, 58]. Instead, we leverage readily available human egocentric video in place of costly IL demonstrations, and we propose an RL approach that benefits from the discovered human activity-priors.

**Exploration for navigation and interaction** Exploration strategies for visual navigation encourage efficient policy learning by rewarding agents for covering area [12, 10, 17], visiting new states [50, 4], expanding the frontier [46], anticipating maps [49], or via intrinsic motivation [41, 6]. For grasping, prior work studies curiosity [26] and disagreement [42] as intrinsic motivation. Beyond navigation and grasping, interaction exploration [39] rewards agents for successfully attempting new object interactions (take potato, open fridge, etc.). In contrast, our agents are rewarded for achieving compatible activity-contexts learned from human egocentric video. Our model incentivizes action sequences that are aligned with activities (e.g., putting a plate in the sink *before* turning-on the faucet), as opposed to arbitrary actions (e.g., turning the faucet on and then immediately off).

**Learning from passive video for embodied agents** Thus far, video plays a limited role in learning for embodied AI, primarily as a source of demonstrations for imitation learning. Prior work learns dynamics models for behavior cloning [56, 66, 43, 53, 52] or crafts reward functions that encourage visiting expert states [1, 16, 54, 35]. Beyond imitation, recent navigation work "re-labels" video frames with pseudo-action labels to learn navigation subroutines [9] or action-conditioned value functions [33]; however, they use videos generated from simulation or from intentionally recorded real-estate tours. In contrast, our work is the first to use free-form human-generated video captured in the real world to learn priors for object interactions. Our priors are not tied to specific goals (as in behavior cloning) and are cast as general purpose auxiliary rewards to encourage efficient RL.

**Learning about human actions from video** Substantial work in computer vision explores models for human action recognition in video [69, 61, 8, 19], including analyzing hand-object interactions [55, 2, 60, 7, 24] and egocentric video understanding [34, 21, 37, 31, 13]. More closely related to our work, visual affordance models derived from video can detect likely places for actions to occur [38, 18, 40], such as where to grasp a frying pan, and object saliency models can identify human-useable objects in egocentric video [14, 5, 20]. These methods learn important concepts from human video, but do not consider their use for embodied agent action, as we propose.

**Semantic priors for embodied agents** Human-centric environments contain useful semantic and structural regularities. Prior work exploits spatial relationships between objects [65, 11, 47] and room layouts [64] to improve navigation (e.g., beds tend to be in bedrooms; kitchens tend to be near dining rooms). Recent work learns visual priors from agent experience (not human video) to understand environment traversability [46], affordances [39], or object properties like mass [36]. These priors encode static properties of objects or geometric relationships between them by learning from co-occurrences in static images. In contrast, our formulation encodes information about objects *in action* from video of humans using objects to answer *what objects should be brought together to enable interactions*, rather than *what objects are typically co-located*. In addition, unlike previous models that require substantial online agent experience to learn priors, we use readily available egocentric video of human activity.

## 3 Approach

Our goal is to train agents to efficiently solve interaction-heavy tasks. Training reinforcement learning (RL) agents is difficult due to sparse task rewards, large search spaces, and context-sensitive actions that depend on what the agent is holding and where other objects are. We propose an auxiliary reward derived from egocentric videos of daily human activity that encourages agents to configure the environment in ways that facilitate successful activities. For example, watching humans wash spoons in the sink suggests that the sink is a good location to bring utensils to before interacting with the nearby faucet or dish soap.

In the following, we begin by defining the visual semantic planning task (Sec. 3.1). Then, we show how to infer activity-context priors from egocentric video (Sec. 3.2) and how to translate those priors to an embodied agent in simulation (Sec. 3.3). We then describe our approach to encode the priors as a dense reward (Sec. 3.4). Finally, we present our policy learning architecture (Sec. 3.5).

### 3.1 Visual semantic planning

We aim to train an agent to bring the environment into a particular goal state specified by the task $\tau$. Agents can perform navigation actions $\mathcal{A}_N$ (e.g., move forward, rotate left/right) and object interactions $\mathcal{A}_I$ (e.g., take/put, open, toggle) involving an object from set $\mathcal{O}$.

Each task is set up as a partially observable Markov decision process. The agent is spawned at an initial state $s_0$. At each time step $t$, the agent in state $s_t$ receives an observation $(x_t, \theta_t, h_t)$ containing the RGB egocentric view, the agent's current pose, and the currently held object (if any). The agent executes an action on an object $o_t$, $(a_t, o_t) \sim \{\mathcal{A}_N \bigcup \mathcal{A}_I\}$, and receives a reward $r_t \sim \mathcal{R}_\tau(s_t, a_t, o_t, s_{t+1})$. For navigation actions, the interaction target $o_t$ is *null*. A recurrent network encodes the agent's observation history over time to arrive at the state representation (detailed below).

A task reward is provided if the agent brings its environment to a goal state $g_\tau$. For example, in the "Clean object" task in our experiments, a *washable* object must be inside the sink, and the faucet must be *toggled-on* for success. A positive reward is given for goal completion, while a small negative reward is given at each time-step to incentivize quick task completion:

$$R_\tau(s_t, a_t, o_t, s_{t+1}) = \begin{cases} 10 & \text{if } g_\tau \text{ satisfied,} \\ -0.01 & \text{otherwise.} \end{cases} \tag{1}$$

The goal is to learn a policy $\pi_\tau$ that maximizes this reward over an episode of length $T$.

### 3.2 Human activity-context from egocentric video

Learning interaction policies requires an understanding of how the environment needs to be configured—where the agent needs to be, and what objects need to be present there—before goals are completed. We infer this directly from a passively collected dataset of human egocentric videos.

The dataset consists of a set of clips $v \in \mathcal{V}$ where each clip captures a person performing an interaction with some object from a video object vocabulary $\mathcal{O}_V$ (e.g., "slice tomato", "pour kettle"). Our model uses the clip's frames only, not the interaction label. Our approach first extracts activity-contexts from each training video frame, and then aggregates their statistics over all training clips to produce an *inter-object activity-context compatibility function*, as follows.

For each frame $f_t$ in an egocentric video clip $v$, we use an off-the-shelf hand-object interaction model [55] that detects active objects—manipulated objects in contact with hands—resulting in a set of class-agnostic bounding boxes. We associate object labels to these boxes using a pre-trained object instance detection model. Specifically, we transfer labels from high-confidence instance detections to active object boxes with large overlap, namely, where intersection over union (IoU) $> 0.5$, resulting in a set of active object boxes and corresponding class labels $\mathcal{D}(f_t) = \{(b_0, o_0)..(b_N, o_N)\}$, where $b_i$ denotes box coordinates and $o_i \in \mathcal{O}_V$. These instances represent objects that are directly involved in the activity, ignoring background objects that are incidentally visible but not interacted with by hands. Detection model details are in Supp.

We infer frame $f_t$'s activity-context $AC(f_t)$ from the Cartesian product of active objects.

$$AC(f_t) = \{(o_i, o_j) \mid o_i, o_j \in \mathcal{D}(f_t) \times \mathcal{D}(f_t)\}, \tag{2}$$

where $o_i \neq o_j$, and each $o_i$ is an object that can be held and moved to different locations, as opposed to a fixed object like a refrigerator or sink. We include a *null* object token to consider cases when the agent visits locations empty-handed. Fig. 2 (left) shows examples.

Each $(o_i, o_j)$ pair represents a particular object $o_i$ and a corresponding activity-context object (ACO) $o_j$—an object that is used with it in an activity. These include movable objects (e.g., tools like knives and cutting-boards in "slice tomato"), receptacles (e.g., kettles and cups in "pour water"), and fixed environment locations (e.g., sinks, faucets in "wash spoon"). This is in contrast to an object's *affordance*, which defines object properties in isolation (e.g. tomatoes are *sliceable* whether a knife is held or not, spoons are *washable* even when they are inside drawers).

Finally, we define the inter-object activity-context compatibility score $\phi(o_i, o_j)$ as follows:

$$\phi(o_i, o_j) = \frac{\sum_{v \in \mathcal{V}} S_v(o_i, o_j)}{\sum_{v \in \mathcal{V}} \sum_{o_k \neq o_i} S_v(o_i, o_k)}, \tag{3}$$

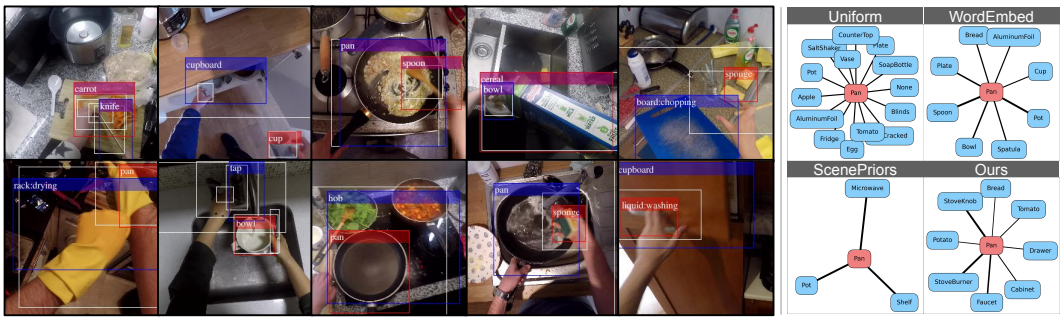


Figure 2: **Left: Detected ACOs from EPIC videos.** White boxes: active object detections. Red/blue boxes: sampled (object, ACO) tuple (Equation 2). Last column shows failure cases—sponge not held (top), liquid:washing false positive (bottom). **Right: ACO compatibility scores in THOR.** Red: active object, blue: top corresponding ACOs. Edge thickness indicates compatibility (Equation 3). Our approach (bottom right) prioritizes object relationships for "Pan" that are relevant for activity (e.g., StoveBurner, StoveKnob) over semantically similar or spatially co-occurring objects (e.g., Pot, Shelf in top-right, bottom-left respectively).

where $S_v(o_i, o_j)$ measures the fraction of frames of video $v$ where $(o_i, o_j) \in AC(f_t)$. By aggregating across clips, we capture dominant object relationships and reduce the effect of object detection errors.

Recall that these relationships are derived from video frames of objects *in action* rather than arbitrary video frames or static images, and are thus strongly tied to human activity, not static scene properties. In other words, $\phi(o_i, o_j)$ measures how likely $o_i$ is *brought together with* $o_j$ during an activity (e.g., knives and cutting boards are used to cut fruits) rather than how likely they generally co-occur in space (e.g., spoons and knives are stored together, but neither enables the action of the other).

### 3.3 Translating to activity-context for embodied agents

Next, we go from compatible object detections from human video, to interactible objects in embodied agents' environments. The compatibility score in Equation 3 is defined over the vocabulary of detectable objects $\mathcal{O}_V$, which may not be the same as the interactible objects $\mathcal{O}$ in the agent's environment. This mismatch is to be expected since we rely on in-the-wild video that is not carefully curated for the particular environments or tasks faced by the agents we train. To account for this, we first map objects in video and those in the agent's domain to a common Glove [44] word embedding space. Then, we re-estimate $\phi(o_m, o_n)$ for each object pair in $\mathcal{O}$ by considering nearest neighbors to $\mathcal{O}_V$ in this space. Specifically, in Equation 3, we set

$$S_v(o_m, o_n) = \sum_{o_i \in \mathcal{N}(o_m)} \sum_{o_j \in \mathcal{N}(o_n)} \sigma_{m,i} \sigma_{n,j} S_v(o_i, o_j), \tag{4}$$

where $\mathcal{N}(o)$ are the nearest neighbors of environment object $o$ among video objects $\mathcal{O}_V$ within a fixed distance in embedding space, and $\sigma_{i,j}$ measures the dot product similarity of the corresponding word embeddings of objects $o_i$ and $o_j$. The resulting compatibility scores represent relationships between objects in the agent's environment, as inferred from similar objects in egocentric video. See Fig. 2 (right) for an illustration of top ACOs compared to other heuristics (like word embedding similarity) and Supp. for further details.

### 3.4 Learning to interact with an activity-context reward

Next, we cast the compatibility scores learned from video into an auxiliary reward for embodied agents. In short, the agent is rewarded for interactions with any objects at any time; however, this reward is enhanced when there are compatible ACOs nearby. For example, it is more valuable to turn-on the stove when there is a pan on it than if there was a knife (or no other objects) nearby. To maximize this reward, the agent must intelligently move objects in the environment to locations with compatible objects before performing interactions.

To achieve this, the agent maintains a persistent memory to keep track of where objects were moved to, and what other objects near it are potential ACOs (and thus, how it affects the value of nearby object interactions). The memory $\mathcal{M}$ starts empty for all objects. At each time-step,

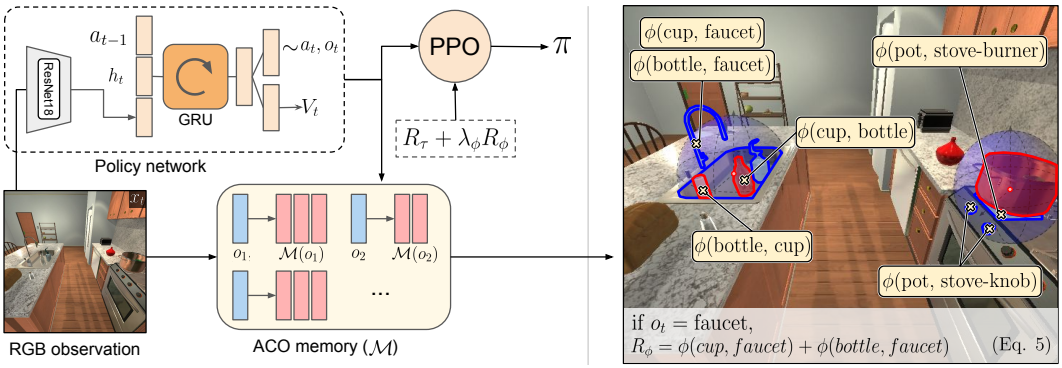


Figure 3: **Policy training framework**. **Left panel:** Our policy network generates an action $(a_t, o_t)$ given the current state observation. If objects are moved by the agent, the ACO memory $\mathcal{M}$ is updated to add objects (red) to their ACO's memory (blue) following Sec. 3.4. **Right panel:** An auxiliary reward $R_\phi$ is provided for object interactions based on nearby ACOs. For example, if the interaction target $o_t$ is the faucet, the agent is rewarded for having the cup and bottle nearby.

if an agent holding an object $o$ *puts-down* the object at location $p$ in 3D space[1], we identify $o$'s neighboring objects. For each neighboring object $o'$, $o$ is a potential ACO and is added to the memory: $\mathcal{M}(o') \bigcup \{(o, p) \mid d(o, o') < \epsilon\}$. We set $\epsilon = 0.5$m. Conversely, if an object $o$ is *picked-up* from a location $p$, it is removed from the list of potential ACOs of all objects it originally neighbored: $\mathcal{M}(o') \setminus \{(o, p)\}\ \forall o' \in \mathcal{M}$, and its own ACO memory is cleared: $\mathcal{M}(o) = \phi$. Fig. 3 (right) shows a snapshot of this memory built until time-step $t-1$. Each object (in red) was placed at its respective locations in previous time-steps, and added to the ACO memories of nearby objects (in blue).

At time-step $t$ the agent is rewarded if it successfully interacts with an object $o_t$ based on this memory. The total activity-context reward $R_\phi$ is the sum of compatibility scores for objects in memory for which $o_t$ is a candidate ACO:

$$R_\phi(s_t, a_t, o_t, \mathcal{M}) = \begin{cases} \sum_{o' \in \mathcal{M}(o_t)} \phi(o', o_t) & \text{if } a_t \in \mathcal{A}_I \wedge c(a_t, o_t) = 0 \\ 0 & \text{otherwise,} \end{cases} \tag{5}$$

where $c(a_t, o_t)$ counts the number of times a particular interaction occurs. Note that the currently held object (or *null*, if nothing is held) is included in $\mathcal{M}$ for all interactions. See Supp. for pseudo-code of the memory update and reward allocation step.

Unlike the sparse task reward that is offered only at the goal state (Sec. 3.1), the activity-context reward is provided at every time-step. These dense rewards are not task-specific, meaning we do not give any privileged information about which video priors are most beneficial to a given task $\tau$. However, they nonetheless encourage the agent to take meaningful actions towards activities, helping to reach states that are strongly aligned with common task subgoals in human videos.

### 3.5 Interaction policy training

Putting it together, our training process is as follows. We adopt an actor-critic policy to train our agent [59]. At each time-step $t$, the current egocentric frame $x_t$ is encoded using a ResNet-18 [28] encoder and then average pooled and fed to an LSTM based recurrent neural network (along with encodings of the held object class and previous action) to aggregate observations over time, and finally to an actor-critic network (MLP) to generate the next action distribution and value. See Fig. 3 (left). In parallel, the agent maintains and updates its activity-context memory and generates an auxiliary reward based on nearby context objects at every time-step following Sec. 3.4.

The final reward is a combination of the task reward (Sec. 3.1) and the activity-context reward:

$$R(s_t, a_t, o_t, s_{t+1}, \mathcal{M}) = R_\tau(s_t, a_t, o_t, s_{t+1}) + \lambda_\phi R_\phi(s_t, a_t, o_t, \mathcal{M}), \tag{6}$$

where $\lambda_\phi$ controls the contribution of the auxiliary reward term. We train our agents using DD-PPO [63] for 5M steps, with rollouts of $T = 256$ time steps. Our model and all baselines use visual

[1]We calculate $p$ from the agent's pose and depth observations using an inverse projection transformation on the interaction target point in the agent's current view.

encoders from agents that are pre-trained for interaction exploration [39] for 5M steps, which we find benefits all approaches. See Fig. 3 and Supp. for architecture, hyperparameter and training details.

# 4 Experiments

We evaluate how well our agents learn complex interaction tasks using our human video based reward.

**Simulator and video datasets.** To train policies, we use the AI2-iTHOR [32] simulator where agents can navigate: $\mathcal{A}_N$ = {move forward, turn left/right 90°, look up/down 30°}, and interact with objects: $\mathcal{A}_I$ = {take, put, open, close, toggle-on, toggle-off, slice}. Our action space of size $|\mathcal{A}|= 110$ is the union of all navigation actions and valid object interactions following [62]. We use all 30 kitchen scenes from AI2-iTHOR, split into training (25) and testing (5) sets. To learn activity-context priors, we use all 55 hours of video from EPIC-Kitchens [13], which contains egocentric videos of daily, unscripted kitchen activities in a variety of homes. It consists of ∼40k video clips annotated for interactions spanning 352 objects ($\mathcal{O}_V$) and 125 actions. Note that we use clip boundaries to segment actions, but we do not use the action labels in our method.

Since kitchen scenes present a diverse set of object interactions in multi-step cooking activities, they are of great interest in this research domain [13, 23, 39]. Further, the alignment of domain with AI2-iTHOR's kitchen environments provides a path to transfer knowledge from videos to agents.

**Visual semantic planning tasks.** We consider seven tasks in our experiments where the agent must **STORE**: put an object that is outside into a drawer, **HEAT**: turn on the stove with a cooking receptacle on it, **COOL**: store a food item or container inside the fridge, **CLEAN**: put an object in the sink and turn on the faucet to wash it, **SLICE**: slice a food item with a knife, **PREP**: place a food item inside a pot/pan, **TRASH**: throw an object into the trash bin,

Each task is associated with a goal state that the environment must be in. For example in the COOL task, an object that was originally outside, must be inside the fridge, and the fridge door must be closed. See Supp. for goal states for all tasks. These tasks represent realistic scenarios in home robot assistant settings, consistent with tasks studied in recent work [58, 39].

We generate 64 episodes per task and per environment with randomized object and agent positions to evaluate our agents. We report task success rates (%) on unseen test environments. This means that our model must both generalize what it observes in the real-world human video to the agent's egocentric views, as well as generalize from training environments to novel test environments.

**Baselines.** We compare several methods:

- **VANILLA** trains a policy using only the task reward (Sec. 3.1). This is the standard approach to train reinforcement learning agents.
- **SCENEPRIORS** [65] also uses only the task reward, but encodes spatial co-occurrences between objects using a graph convolutional network (GCN) model to enhance its state representation. We use the authors' code.
- **NAVEXP** [49] adds an auxiliary navigation exploration reward to the vanilla model. We use object coverage [17, 49], which rewards the agent for visiting (but not interacting with) new objects.
- **INTEXP** [39] is a state-of-the-art exploration method for object interaction that adds a reward for successful object interactions regardless of nearby context objects. We use the authors' code.

SCENEPRIORS represents the conventional approach of introducing visual priors into policy learning—through GCN encoders, not auxiliary rewards—and considers spatial co-occurrence instead of activity based priors. NAVEXP and INTEXP are exploration methods from recent work that incentivize all locations or objects equally. In contrast, our reward is a function of the state of the environment, and how well it aligns with observed states in video of human activity.

## 4.1 Visual semantic planning results

Table 1 shows success rates across all tasks. The numbers in brackets represent the probability that an agent executing random actions will reach the goal state assuming all objects are within reach. *Interaction-heavy* tasks requiring long sequences of steps like STORE (pick up object, open drawer, put object inside, close drawer) are significantly harder than *interaction-light* tasks like PREP (pick up food item, put in pan)—in this particular example, by four orders of magnitude.

Our approach outperforms prior work on all interaction-heavy tasks, and performs competitively on interaction-light tasks. Compared to the VANILLA baseline, SCENEPRIORS's stronger observation

| | COOL(1e-7) | STORE(1e-7) | HEAT(2e-6) | CLEAN(2e-5) | SLICE(1e-3) | PREP(2e-3) | TRASH(2e-3) |
|---|---|---|---|---|---|---|---|
| VANILLA | 0.12±0.07 | 0.00±0.00 | 0.01±0.01 | 0.35±0.12 | 0.30±0.03 | 0.22±0.05 | 0.14±0.05 |
| SCENEPRIORS [65] | 0.14±0.09 | 0.00±0.00 | 0.04±0.01 | 0.35±0.02 | **0.36**±0.05 | 0.26±0.08 | 0.20±0.06 |
| NAVEXP [49] | 0.05±0.04 | 0.01±0.01 | 0.01±0.01 | 0.43±0.02 | 0.29±0.05 | **0.33**±0.02 | **0.25**±0.04 |
| INTEXP [39] | 0.11±0.06 | 0.03±0.03 | 0.06±0.03 | 0.19±0.05 | 0.26±0.07 | 0.19±0.08 | 0.02±0.01 |
| OURS | **0.26**±0.06 | **0.12**±0.07 | **0.13**±0.05 | **0.53**±0.03 | **0.36**±0.06 | 0.26±0.07 | 0.13±0.02 |

Table 1: **Task success rates (%) on test environments.** Numbers in brackets indicate the probability that a random agent performs the correct sequence of interactions to complete a task (e.g., COOL is much harder than PREP). Our video-based activity-context priors result in the best performance across all interaction-heavy tasks. Navigation-based exploration agents perform well for tasks that involve minimal object interactions (e.g. PREP, TRASH). Values are averaged over 3 training runs.

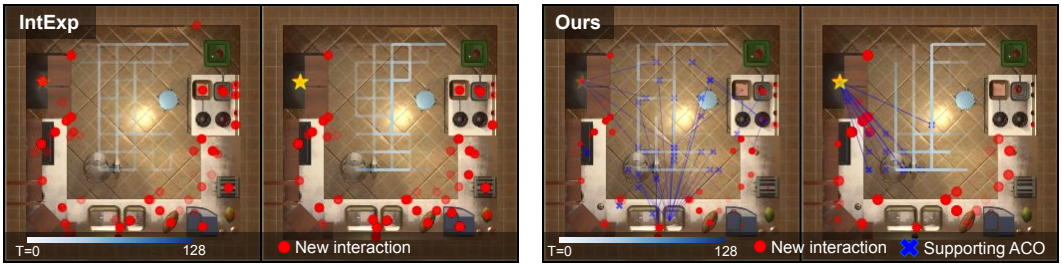


Figure 4: **Policy rollouts for the COOL task during early/late stage training**. Red dots represent auxiliary rewards (size represents value). Our agents learn to move objects to suitable locations (near supporting ACOs, blue crosses) to increase their reward, rather than simply visit *more* objects. See text.

encoding results in better performance with the same task reward; however, while its spatial co-occurrence prior encoding what objects are *found together* (rather than *used together*) is helpful for object search [65], it falls short for object interactions.

INTEXP provides a task-agnostic objective for pre-training interaction agents; it results in strong visual encoder features, but yields poor performance when used as an auxiliary reward. It fails to discriminate between useful interaction sequences that are aligned with task goals (e.g., put pan on stove, turn-on stove) and arbitrary interaction sequences that maximize its reward (e.g., pick up, then put down objects sequentially). NAVEXP performs poorly on interaction-heavy tasks, often achieving close to zero success rates. However, it performs well on tasks like TRASH and PREP where a single object must be moved to a target location for success.

Fig. 4 shows qualitative results of 100 policy rollouts. Each panel shows a side-by-side comparison of agent behavior during early (1M steps) and late (5M steps) training for the same task. Each red dot represents an auxiliary reward given to the agent, sized by reward value; the yellow star denotes where the task reward is issued. INTEXP (left panel) is rewarded uniformly for each interaction (equally sized red dots), and thus has a single strategy to maximize reward — perform *more* interactions. In contrast, our approach (right panel) can maximize reward along two axes: perform more interactions as before, or selectively move objects to favorable positions near ACOs (blue crosses). This guides agent exploration to relevant states for activities, translating to higher task success.

### 4.2 Training speed and task success versus difficulty

Fig. 5 shows consolidated results across all tasks, treating each episode of each task as an individual instance that can be successful or not. These results give a sense of overall performance of each baseline, despite variation across tasks. See Supp. for a task-specific breakdown of results.

Fig. 5 (left) shows convergence speed. Our approach sees an increased success rate at early training epochs as our dense rewards allow agents to make progress towards goals without explicit task rewards. NAVEXP performs poorly in early iterations as it prioritizes navigation actions to visit as many objects as possible. Overall this result shows a key advantage of our idea: learning from human video not only boosts overall policy success rates, but it also accelerates the agent's learning process, giving a "head start" from having observed how people perform general daily interactions. Importantly, our ablations (below) show that access to the video alone is not sufficient; our idea to capture functional activity-contexts is key.

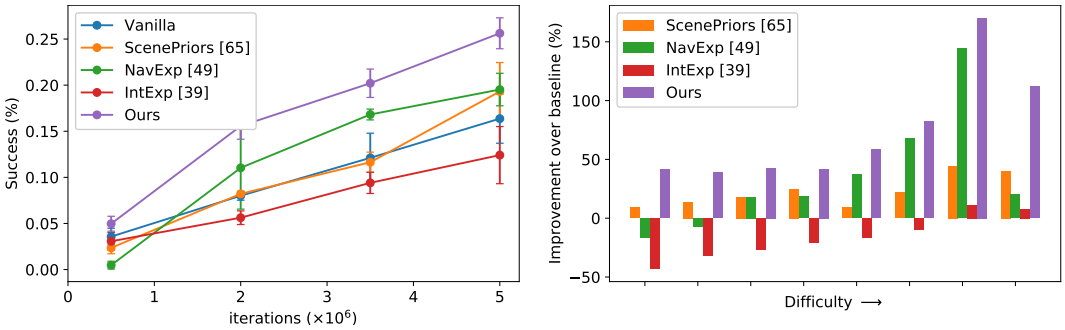


Figure 5: **Consolidated performance across all seven tasks. Left: Success rate vs. training iteration.** Our dense activity-context rewards accelerate performance at early training iterations. **Right: Success rate improvement vs. navigation difficulty.** Episodes are sorted and grouped by ideal navigation distance to task objects (low → high). Our method shows largest improvements over VANILLA on difficult instances.

| | COOL | STORE | HEAT | CLEAN | SLICE | PREP | TRASH |
|---|---|---|---|---|---|---|---|
| UNIFORM | 0.07±0.00 | 0.02±0.01 | 0.12±0.00 | 0.37±0.05 | 0.26±0.03 | 0.22±0.05 | 0.02±0.01 |
| WORDEMBED | 0.19±0.03 | 0.02±0.02 | 0.03±0.00 | 0.40±0.13 | 0.34±0.01 | 0.22±0.01 | 0.04±0.02 |
| SPATIALCOOC | **0.33**±0.05 | 0.02±0.01 | 0.06±0.02 | 0.47±0.09 | 0.33±0.02 | 0.25±0.08 | **0.13**±0.05 |
| OURS (INTSEQ) | **0.33**±0.04 | 0.11±0.03 | 0.07±0.03 | 0.43±0.04 | 0.30±0.05 | 0.20±0.02 | 0.09±0.04 |
| OURS (ACO) | 0.26±0.06 | **0.12**±0.07 | **0.13**±0.05 | **0.53**±0.03 | **0.36**±0.06 | **0.26**±0.07 | **0.13**±0.02 |

Table 2: **Compatibility score ablations.** Simple similarity measures (WORDEMBED) or naively using interaction labels in video (INTSEQ) produces sub-optimal policies. Our activity-context detection based priors perform the best across the majority of tasks. Results are averaged over 3 runs.

Fig. 5 (right) analyzes task success as a function of navigation difficulty. We measure this as the ideal geodesic distance to each of the objects required to reach the goal state. This value is low when objects are close by in small environments, and large when the agent has to move around to find specific objects in large environments. We sort episodes by this criterion into 8 groups of increasing difficulty and compare each method against the VANILLA baseline policy. Our method has the highest success rates across difficulties and offers the largest improvements on more difficult instances.

### 4.3 Compatibility function ablations

Finally we investigate how different sources of priors impact our agents when incorporated into our reward scheme. Specifically, we vary how we compute our compatibility score $\phi(o_i, o_j)$ in Equation 3. **UNIFORM** assigns equal compatibility to all object pairs; **WORDEMBED** uses Glove [44] embedding similarity; **SPATIALCOOC** uses spatial co-occurrence statistics derived from static images[2]; **OURS (INTSEQ)** uses transition probabilities between object interactions from ground truth, labeled sequences of the same egocentric videos.

Table 2 shows the results. Uniform or semantic similarity heuristics are insufficient to capture the role of objects participating in activities. SPATIALCOOC's object co-occurrences are aligned for some tasks (e.g., vegetables are co-located with fridges in static images, and activities involving storing groceries), but also capture unhelpful static relationships (see Fig. 2, right). INTSEQ benefits from egocentric video; however, it uses ground truth video object labels, and falsely assumes that the next object interacted with in time is necessarily relevant to the previous object. Our activity-context score is tightly linked to each object's involvement in interactions, and proves to be the most useful prior.

## 5 Conclusion

We proposed an approach to discover activity-context priors from real-world human egocentric video and use them to accelerate training of interaction agents. Our work provides an avenue for embodied agents to directly benefit from human experience even when the tasks and environments differ. Future work could explore policies that update human-activity prior beliefs during task-specific policy training, and policy architectures that encode such priors jointly in the state representation.

**Acknowledgments**: Thanks to Santhosh Ramakrishnan for helpful feedback and the UT Systems Admin team for their continued support. UT Austin is supported in part by DARPA L2M and the UT Austin NSF AI Institute.

[2]This is different from SCENEPRIORS (Sec. 4.1) which encodes the same priors in the state representation.

## References


[1] Yusuf Aytar, Tobias Pfaff, David Budden, Thomas Paine, Ziyu Wang, and Nando de Freitas. Playing hard exploration games by watching youtube. In *NeurIPS*, 2018. 3

[2] Sven Bambach, Stefan Lee, David J Crandall, and Chen Yu. Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions. In *ICCV*, 2015. 3

[3] Dhruv Batra, Angel X Chang, Sonia Chernova, Andrew J Davison, Jia Deng, Vladlen Koltun, Sergey Levine, Jitendra Malik, Igor Mordatch, Roozbeh Mottaghi, et al. Rearrangement: A challenge for embodied ai. *arXiv preprint arXiv:2011.01975*, 2020. 1, 2

[4] Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. In *NeurIPS*, 2016. 3

[5] Gedas Bertasius, Hyun Soo Park, Stella X. Yu, and Jianbo Shi. First person action-object detection with egonet. In *RSS*, 2017. 3

[6] Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A Efros. Large-scale study of curiosity-driven learning. *ICLR*, 2019. 3

[7] Zhe Cao, Ilija Radosavovic, Angjoo Kanazawa, and Jitendra Malik. Reconstructing hand-object interactions in the wild. *arXiv preprint arXiv:2012.09856*, 2020. 3

[8] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017. 3

[9] Matthew Chang, Arjun Gupta, and Saurabh Gupta. Semantic visual navigation by watching youtube videos. *NeurIPS*, 2020. 3

[10] Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. In *ICLR*, 2020. 3

[11] Devendra Singh Chaplot, Dhiraj Prakashchand Gandhi, Abhinav Gupta, and Russ R Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. *NeurIPS*, 2020. 2, 3

[12] Tao Chen, Saurabh Gupta, and Abhinav Gupta. Learning exploration policies for navigation. In *ICLR*, 2019. 3

[13] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. Scaling egocentric vision: The epic-kitchens dataset. In *ECCV*, 2018. License available at: https://github.com/epic-kitchens/epic-kitchens-55-annotations/blob/master/LICENSE.txt. 2, 3, 7

[14] Dima Damen, Teesid Leelasawassuk, Osian Haines, Andrew Calway, and Walterio W Mayol-Cuevas. You-do, i-learn: Discovering task relevant objects and their modes of interaction from multi-user egocentric video. In *BMVC*, 2014. 3

[15] Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied question answering. In *ICCV-W*, 2018. 2, 3

[16] Debidatta Dwibedi, Jonathan Tompson, Corey Lynch, and Pierre Sermanet. Learning actionable representations from visual observations. In *IROS*, 2018. 3

[17] Kuan Fang, Alexander Toshev, Li Fei-Fei, and Silvio Savarese. Scene memory transformer for embodied agents in long-horizon tasks. In *CVPR*, 2019. 3, 7, 4

[18] Kuan Fang, Te-Lin Wu, Daniel Yang, Silvio Savarese, and Joseph J Lim. Demo2vec: Reasoning object affordances from online videos. In *CVPR*, 2018. 3

[19] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *ICCV*, 2019. 3

[20] A. Furnari, S. Battiato, K. Grauman, and G. Maria Farinella. Next-active-object prediction from egocentric videos. *JVCIR*, 2017. 3

[21] Antonino Furnari and Giovanni Maria Farinella. What would you expect? anticipating egocentric actions with rolling-unrolling lstms and modality attention. In *ICCV*, 2019. 3

[22] Chuang Gan, Jeremy Schwartz, Seth Alter, Martin Schrimpf, James Traer, Julian De Freitas, Jonas Kubilius, Abhishek Bhandwaldar, Nick Haber, Megumi Sano, et al. Threedworld: A platform for interactive multi-modal physical simulation. *arXiv preprint arXiv:2007.04954*, 2020. 1, 2

[23] Z. Gao, R. Gong, T. Shu, X. Xie, S. Wang, and S. C. Zhu. Vrkitchen: an interactive 3d virtual environment for task-oriented learning. *arXiv:1903.05757*, 2019. 7

[24] Guillermo Garcia-Hernando, Shanxin Yuan, Seungryul Baek, and Tae-Kyun Kim. First-person hand action benchmark with rgb-d videos and 3d hand pose annotations. In *CVPR*, 2018. 3

[25] Daniel Gordon, Aniruddha Kembhavi, Mohammad Rastegari, Joseph Redmon, Dieter Fox, and Ali Farhadi. Iqa: Visual question answering in interactive environments. In *CVPR*, 2018. 2

[26] Nick Haber, Damian Mrowca, Stephanie Wang, Li F Fei-Fei, and Daniel L Yamins. Learning to play with intrinsically-motivated, self-aware agents. In *NeurIPS*, 2018. 3

[27] Ankur Handa, Karl Van Wyk, Wei Yang, Jacky Liang, Yu-Wei Chao, Qian Wan, Stan Birchfield, Nathan Ratliff, and Dieter Fox. Dexpilot: Vision based teleoperation of dexterous robotic hand-arm system. *ICRA*, 2020. 2

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6

[29] Unnat Jain, Luca Weihs, Eric Kolve, Ali Farhadi, Svetlana Lazebnik, Aniruddha Kembhavi, and Alexander Schwing. A cordial sync: Going beyond marginal policies for multi-agent embodied tasks. In *ECCV*, 2020. 1

[30] Unnat Jain, Luca Weihs, Eric Kolve, Mohammad Rastegari, Svetlana Lazebnik, Ali Farhadi, Alexander G Schwing, and Aniruddha Kembhavi. Two body problem: Collaborative visual task completion. In *CVPR*, 2019. 2, 3

[31] Hao Jiang and Kristen Grauman. Seeing invisible poses: Estimating 3d body pose from egocentric video. In *CVPR*, 2017. 3

[32] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017. License available at: https://github.com/allenai/ai2thor/blob/main/LICENSE. 1, 2, 7, 4

[33] Ashish Kumar, Saurabh Gupta, and Jitendra Malik. Learning navigation subroutines from egocentric videos. In *CoRL*, 2020. 3

[34] Yin Li, Miao Liu, and James M Rehg. In the eye of beholder: Joint learning of gaze and actions in first person video. In *ECCV*, 2018. 3

[35] YuXuan Liu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Imitation from observation: Learning to imitate behaviors from raw video via context translation. In *ICRA*, 2018. 3

[36] Martin Lohmann, Jordi Salvador, Aniruddha Kembhavi, and Roozbeh Mottaghi. Learning about objects by learning to interact with them. *NeurIPS*, 2020. 3

[37] Zheng Lu and Kristen Grauman. Story-driven summarization for egocentric video. In *CVPR*, 2013. 3

[38] Tushar Nagarajan, Christoph Feichtenhofer, and Kristen Grauman. Grounded human-object interaction hotspots from video. In *ICCV*, 2019. 3

[39] Tushar Nagarajan and Kristen Grauman. Learning affordance landscapes for interaction exploration in 3d environments. *NeurIPS*, 2020. 3, 7, 8, 4

[40] Tushar Nagarajan, Yanghao Li, Christoph Feichtenhofer, and Kristen Grauman. Ego-topo: Environment affordances from egocentric video. In *CVPR*, 2020. 3

[41] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *ICML*, 2017. 3

[42] Deepak Pathak, Dhiraj Gandhi, and Abhinav Gupta. Self-supervised exploration via disagreement. In *ICML*, 2019. 3

[43] Deepak Pathak, Parsa Mahmoudieh, Guanghao Luo, Pulkit Agrawal, Dian Chen, Yide Shentu, Evan Shelhamer, Jitendra Malik, Alexei A Efros, and Trevor Darrell. Zero-shot visual imitation. In *CVPR-W*, 2018. 3

[44] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014. 5, 9, 3

[45] Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. Virtualhome: Simulating household activities via programs. In *CVPR*, 2018. 1, 2

[46] William Qi, Ravi Teja Mullapudi, Saurabh Gupta, and Deva Ramanan. Learning to move with affordance maps. *ICLR*, 2020. 3

[47] Y. Qiu, A. Pal, and H. I. Christensen. Learning hierarchical relationships for object-goal navigation. In *CoRL*, 2020. 2, 3

[48] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *RSS*, 2018. 2

[49] Santhosh K. Ramakrishnan, Dinesh Jayaraman, and Kristen Grauman. An exploration of embodied visual exploration. *IJCV*, 2021. 3, 7, 8, 4, 5

[50] Nikolay Savinov, Anton Raichuk, Raphaël Marinier, Damien Vincent, Marc Pollefeys, Timothy Lillicrap, and Sylvain Gelly. Episodic curiosity through reachability. *ICLR*, 2019. 3

[51] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *ICCV*, 2019. 2, 4

[52] Karl Schmeckpeper, Oleh Rybkin, Kostas Daniilidis, Sergey Levine, and Chelsea Finn. Reinforcement learning with videos: Combining offline observations with interaction. *CoRL*, 2020. 3

[53] Karl Schmeckpeper, Annie Xie, Oleh Rybkin, Stephen Tian, Kostas Daniilidis, Sergey Levine, and Chelsea Finn. Learning predictive models from observation and interaction. *ECCV*, 2020. 3

[54] Pierre Sermanet, Corey Lynch, Jasmine Hsu, and Sergey Levine. Time-contrastive networks: Self-supervised learning from multi-view observation. In *CVPR-W*, 2017. 3

[55] Dandan Shan, Jiaqi Geng, Michelle Shu, and David F Fouhey. Understanding human hands in contact at internet scale. In *CVPR*, 2020. 3, 4

[56] Pratyusha Sharma, Lekha Mohan, Lerrel Pinto, and Abhinav Gupta. Multiple interactions made easy (mime): Large scale demonstrations data for imitation. In *CoRL*, 2018. 3

[57] Bokui Shen, Fei Xia, Chengshu Li, Roberto Martín-Martín, Linxi Fan, Guanzhi Wang, Shyamal Buch, Claudia D'Arpino, Sanjana Srivastava, Lyne P Tchapmi, et al. igibson, a simulation environment for interactive tasks in large realisticscenes. *arXiv preprint arXiv:2012.02924*, 2020. 1, 2

[58] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *CVPR*, 2020. 1, 2, 3, 7

[59] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018. 6

[60] Bugra Tekin, Federica Bogo, and Marc Pollefeys. H+ o: Unified egocentric recognition of 3d hand-object poses and interactions. In *CVPR*, 2019. 3

[61] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *ECCV*, 2016. 3

[62] Luca Weihs, Matt Deitke, Aniruddha Kembhavi, and Roozbeh Mottaghi. Visual room rearrangement. In *CVPR*, 2021. 7

[63] Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, and Dhruv Batra. Dd-ppo: Learning near-perfect pointgoal navigators from 2.5 billion frames. *arXiv preprint arXiv:1911.00357*, 2019. 6

[64] Yi Wu, Yuxin Wu, Aviv Tamar, Stuart Russell, Georgia Gkioxari, and Yuandong Tian. Learning and planning with a semantic model. *arXiv preprint arXiv:1809.10842*, 2018. 2, 3

[65] Wei Yang, Xiaolong Wang, Ali Farhadi, Abhinav Gupta, and Roozbeh Mottaghi. Visual semantic navigation using scene priors. *ICLR*, 2019. 2, 3, 7, 8, 4

[66] Tianhe Yu, Chelsea Finn, Annie Xie, Sudeep Dasari, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot imitation from observing humans via domain-adaptive meta-learning. *RSS*, 2018. 3

[67] Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *ICRA*, 2018. 2

[68] Yuke Zhu, Daniel Gordon, Eric Kolve, Dieter Fox, Li Fei-Fei, Abhinav Gupta, Roozbeh Mottaghi, and Ali Farhadi. Visual semantic planning using deep successor representations. In *ICCV*, 2017. 1, 2, 3

[69] Yi Zhu, Xinyu Li, Chunhui Liu, Mohammadreza Zolfaghari, Yuanjun Xiong, Chongruo Wu, Zhi Zhang, Joseph Tighe, R Manmatha, and Mu Li. A comprehensive study of deep video action recognition. *arXiv preprint arXiv:2012.06567*, 2020. 3

## Supplementary Material

This section contains supplementary material to support the main paper text. The contents include:

- (§S1) A video demonstrating our agents ACO memory creation process and reward calculation described in Sec. 3.4.
- (§S2) A video comparing our agents to baseline policies to supplement Fig. 4.
- (§S3) Algorithm for creating and updating ACO memory described in Sec. 3.4.
- (§S4) Figures illustrating the correspondences between scenes in EPIC Kitchens and observations in THOR using our method (Sec. 3.3)
- (§S5) Additional qualitative figures to supplement Fig. 2 in the main paper.
- (§S6) Detection model architecture details and additional implementation details related to Sec. 3.2
- (§S7) Policy architecture details, training details and hyperparameters for our model in Sec. 3.5
- (§S8) Implementation details for baselines in Sec. 4 (Baselines).
- (§S9) Goal descriptions for each task presented in Sec. 4 (Tasks).
- (§S11) Task-specific breakdown of results presented in Sec. 4.1 and Sec. 4.2.
- (§S10) Additional experiments varying model architecture and combined reward schemes to supplement Sec. 4.

## S1 ACO memory creation and reward calculation demo video

In the video, we show the process of creating and building the activity-context memory described in Sec. 3.4. The video shows the first-person view of the agent as it performs a series of object interactions (top left). We overlay masks for objects that are manipulated (red), nearby objects whose activity-context object (ACO) memory is updated (blue), the distance threshold under which objects are considered *near* to each other (teal). Finally, we show how the presence of these objects affects the reward provided to the agent (top right). The video illustrates that bringing a mug to the sink or a pot to the stove results in high rewards as they are aligned with activities, while bringing a knife to the garbage bin is far less valuable.

## S2 Video demo of our policy compared to baselines

In the video, we show rollouts of various policies during training to supplement Fig. 4 in the main paper. The video compares the baseline approaches to our method. It shows each step in a trajectory and the corresponding reward given to the agent, illustrating the contribution of activity-context objects to the total reward in our method compared to the uniform reward provided in the baselines. As mentioned in Sec. 4.1, our agents receive variable rewards (non-uniformly sized red dots) based on nearby ACOs (blue crosses), which encourages agents to move objects to more meaningful positions aligned with activities.

## S3 ACO memory creation and update algorithm

We present the algorithm for creating and maintaining the ACO memory in Algorithm 1. This corresponds to the steps outlined in Sec. 3.4 of the main paper and the accompanying reward equation Equation 5. Note that in practice, we normalize $\phi(o_t, o)$ in L16 of Algorithm 1 such that the maximum rewarding ACO offers a reward of 1.0, to ensure that agents do not ignore objects with scores distributed across many potential ACOs.

## S4 ACO correspondences between EPIC and THOR scenes

As mentioned in Sec. 3.3 we translate from ACO pairs learned in video to a reward used in simulation. We show qualitative results for which scenes in THOR correspond to activities observed in EPIC in Fig. S1. The first column of each row shows a frame from an EPIC video clip showing a particular human activity. The remaining columns show similar "states" from THOR that our agents deem

**Algorithm 1** Activity-context reward memory.

**Input:** ACO memory $\mathcal{M}$, visitation count $c$, distance metric $d$, distance threshold $\epsilon$
**Input:** State $s_t$, action $(a_t, o_t)$, held object $o$ at time-step $t$

**function** UPDATEMEM($s_t, a_t, o_t, \mathcal{M}$)
  **if** $a_t =$ "put" **then** ▷ Put $o$ at position $p$
    $\mathcal{M}(o') \leftarrow \mathcal{M}(o') \bigcup \{(o, p) \mid d(o, o') < \epsilon\}$
  **else if** $a_t =$ "take" **then** ▷ Take $o_t$ from location $p_t$
    $\mathcal{M}(o_t) \leftarrow \{\}$
    $\mathcal{M}(o') \leftarrow \mathcal{M}(o') \setminus \{(o_t, p_t)\} \mid \forall o' \in \mathcal{M}$
  **end if**
  **return** Updated memory $\mathcal{M}$
**end function**

**function** ACREWARD($s_t, a_t, o_t, \mathcal{M}$)
  **if** $a_t \notin \mathcal{A}_{\mathcal{I}}$ **or** $c(a_t, o_t) > 0$ **then return** 0 ;
  $\mathcal{M} \leftarrow$ UPDATEMEM($s_t, a_t, o_t, \mathcal{M}$,)
  $Z \leftarrow \max_{o \in \mathcal{O}} \phi(o_t, o)$
  $R_{ACO} \leftarrow \sum_{o \in \mathcal{M}(o_t)} \phi(o_t, o)/Z$ ▷ Equation 5
  $c(a_t, o_t) \leftarrow c(a_t, o_t) + 1$
  **return** $R_\phi$
**end function**

desirable to reach based on the distribution of ACOs present. Interactions performed in these states are highly rewarded following our approach.

The figures also illustrate our automatic mapping from the video vocabulary $\mathcal{O}_V$ to the agent environment vocabulary $\mathcal{O}$ for objects using word embedding similarity described in Sec. 3.3. For example, "garlic:paste" in EPIC is mapped to other food-like objects in THOR like "tomatoes" (left, bottom row); "drawers" are mapped to both "cabinets" and "drawers" (right, top row).

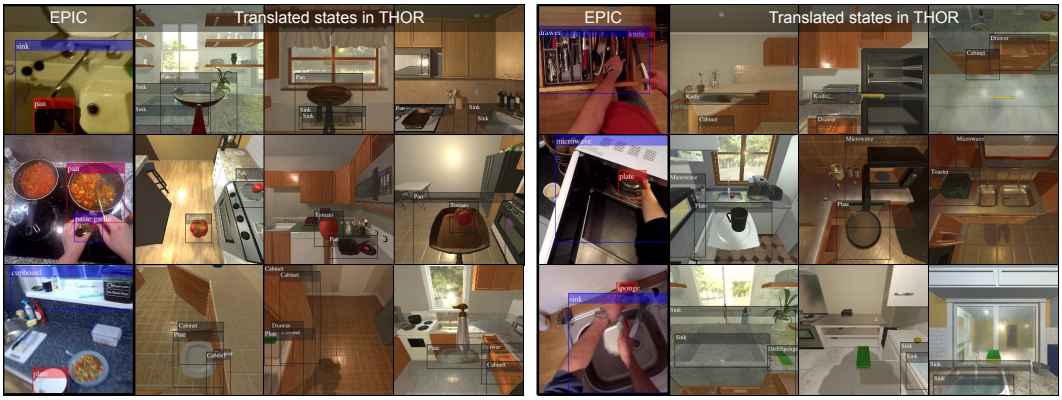


Figure S1: **Discovered EPIC $\longleftrightarrow$ THOR ACO correspondences.** First column shows a human activity from EPIC. Subsequent columns show similar states from THOR which provide high rewards when interactions are performed with objects once the agent is in that state.

## S5 Additional ACO detection results on EPIC-Kitchens

We show additional detection results to supplement Fig. 2 (left). These images show sampled (object, ACO) tuples (Equation 2) in red and blue respectively. The last column shows failure cases due to incorrect active object detections and incorrect object instance detection.

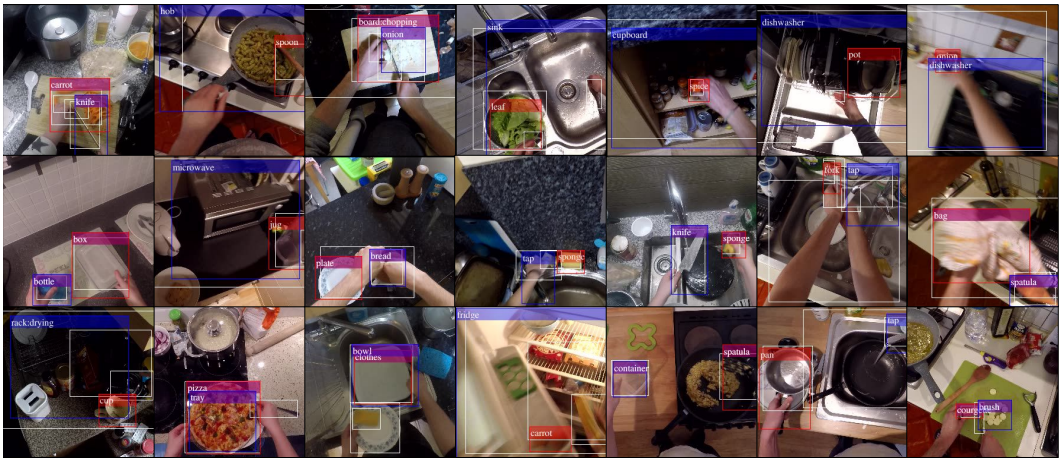

Figure S2: **Additional EPIC detections to supplement Fig. 2.** Last column shows failure cases.

## S6 Pre-trained detection model and ACO scoring details

**Pre-trained detection models** As mentioned in Sec. 3.2, we use two detection models in our approach – (1) An active object detector which generates high-confidence box proposals for objects being interacted with hands (but does not assign object class labels to them) and (2) An object instance detector that produces a set of named objects and boxes for visible object instances. For (1) we use pre-trained models provided by authors of [55][3]. For (2) we use pre-computed detections per frame using a Faster-RCNN model released by the authors of EPIC-Kitchens [13]. We set confidence thresholds of 0.5 for all models.

**Activity context curation details** To infer each frame's activity-context following Equation 2, we use a manually curated list of *moveable* objects from EPIC, though it is possible to automatically infer this list from action labels on video clips (all objects that are *picked* up), or using the aforementioned hand-object detectors. Of the 398 objects in EPIC, 349 are moveable. We list the remaining objects in the table below.

| | | | | | | |
|---|---|---|---|---|---|---|
| tap | top | microwave | machine:washing | toaster | machine:sous:vide | flame |
| cupboard | oven | button | processor:food | knob | window | fire |
| drawer | maker:coffee | juicer | plug | ladder | heater | grill |
| hand | sink | scale | kitchen | wall | door:kitchen | time |
| fridge | heat | rack:drying | floor | tv | table | timer |
| hob | dishwasher | freezer | fan:extractor | shelf | rug | desk |
| bin | blender | light | chair | stand | switch | lamp |

**ACO mapping details** As mentioned in Sec. 3.3, to match object classes between AI2-iTHOR and EPIC Kitchens we map each (object, ACO) tuple in the video object space $\mathcal{O}_V$ to corresponding tuples in environment object space $\mathcal{O}$ following Equation 4. We use a GloVe [44] similarity threshold of 0.6. Lower values lead to undesirable mappings across object classes (e.g., toasters and refrigerators which are both appliances, but participate in distinct activities) .

## S7 Policy architecture and training details

We provide additional architecture and training details to supplement information provided in Sec. 3.5 in the main paper.

**Policy network** As mentioned in Sec. 3.5, we use a ResNet-18 observation encoder pretrained with observations from 5M iterations of training of an interaction exploration agent [39]. We transfer

[3] https://github.com/ddshan/hand_object_detector

the backbone only and freeze it during task training. Each RGB observation is encoded to a 512-D feature. A single linear embedding layer is used to embed the previous action and the currently held object (or *null*) to vectors of dimension 32 each. The total observation feature is the concatenation of these three features. All architecture hyperparams are listed in Table S1 (Policy network).

**Training hyperparameters** We modify the Habitat-Lab codebase [51] to support training agents in the THOR simulator platform [32]. We search over $\lambda_\phi \in \{0.01, 0.1, 1.0, 5.0\}$ for Equation 6 and select $\lambda_\phi = 1.0$ which has the highest consolidated performance on validation episodes for all methods following the procedure in Sec. 4.2. All training hyperparameters are listed in Table S1 (RL training).

| RL training | |
|---|---|
| Optimizer | Adam |
| Learning rate | 2.5e-4 |
| # parallel actors | 64 |
| PPO mini-batches | 2 |
| PPO epochs | 4 |
| PPO clip param | 0.2 |
| Value loss coefficient | 0.5 |
| Entropy coefficient | 0.01 |
| Normalized advantage? | Yes |
| Training episode length | 256 |
| LSTM history length | 256 |
| # training steps ($\times$ 1e6) | 5 |
| **Policy network** | |
| Backbone | resnet18 |
| Input image size | 256×256 |
| LSTM hidden size | 512 |
| # layers | 2 |

Table S1: **RL policy architecture and training hyperparameters.**

## S8 Baseline implementation details

We present implementation details for baselines in Sec. 4 (Baselines). NAVEXP and INTEXP baselines use the same architecture as our model described in Sec. S7, but vary in the reward they receive during training. SCENEPRIORS uses a different backbone architecture that uses a GCN based state encoder as described below.

**NAVEXP** Agents are rewarded for visiting new objects such that the object is visible and within interaction range (less than $1.5m$ away). A constant reward is provided for every new object class visited. This is similar to previous implementations [49, 17, 39]. We use the implementation from [39].

**INTEXP** Following [39], agents are rewarded for new object interactions. The reward provided has the form in Equation 5, but provides a constant reward regardless of ACOs present. We use the author's code.

**SCENEPRIORS** We modify the architecture in [65], which was built for object search. We use the author's code. First, we remove the goal object encoding as the agent is not searching for a single object. Second, we replace the backbone network with our shared ResNet backbone to ensure fair comparison. We use a GCN encoding dimension of 512. The remaining architecture details are consistent with [65].

## S9 Goal condition details for all tasks

We next list out formal goal conditions for all tasks described in Sec. 4 of the main paper (Tasks). Each goal is specified as a conjunction of predicates that need to be satisfied in a candidate goal state. The goal is satisfied if for any object $o$ in the environment, the following conditions are true:

- STORE: inReceptacle(o, Drawer) $\wedge$ isClosed(Drawer) $\wedge$ isStorable(o)
- HEAT: inReceptacle(o, StoveBurner) $\wedge$ isToggledOn(StoveKnob) $\wedge$ isHeatable(o)
- COOL: inReceptacle(o, Fridge) $\wedge$ isClosed(Fridge) $\wedge$ isCoolable(o)
- CLEAN: inReceptacle(o, SinkBasin) $\wedge$ isToggledOn(Faucet) $\wedge$ isCleanable(o)
- SLICE: isHolding(Knife) $\wedge$ isSliceable(o)
- PREP: [inReceptacle(o, Pot) | inReceptacle(o, Pan)] $\wedge$ isCookable(o)
- TRASH: inReceptacle(o, GarbageCan) $\wedge$ isTrashable(o)

where *inReceptacle* checks if an object is inside/on top of a particular object, *is-X-able* filters for objects with specific affordances (e.g., only objects that can be placed on the stove like pots/pans/kettles satisfy *isHeatable*), *isClosed* and *isToggledOn* checks for specific object states, and *isHolding* checks if the agent is holding a specific type of object (e.g., for SLICE this has to be a Knife or a ButterKnife). Further, for each task that involves moving objects to receptacles, the object must originally have been outside the receptacle (e.g., outside the Fridge for COOL; off the stovetop for HEAT).

## S10 Additional ablation experiments

We present a comparison of different backbone architectures (ResNet18 vs. ResNet50) and aggregation modules (LSTM vs. GRU) for both our model and the baselines in Table S3. We evaluate on the unseen test episodes for 4 interaction-heavy tasks. The average results of 2 training runs are in the table below. Using stronger backbones seems to help marginally, but does not offer conclusive results. Using the simpler GRU based aggregation (instead of LSTM) results in large improvements. Overall, the trends remain consistent across all configurations: Vanilla $<$ NavExp $<$ Ours. Architectural changes alone in the baselines (to either the backbone, or the aggregation mechanism) are not enough to compensate for task difficulty — performance on Cool, Store and Heat remain low ($<$10%) for Vanilla and NavExp.

In Table S2 we show the results of our policies combined with rewards from a navigation exploration agent. As mentioned in Sec. 4.1, while our agent excels in interaction-heavy tasks, navigation exploration agents perform well on interaction-light tasks which often require finding a single object and bringing it to the right location. For example in the Trash task, there is a single garbage can that navigation agents quickly find as they cover area, but that our agents struggle to find early on. The two strategies can be combined to address this issue. Table S2 shows average results of 2 runs where we add the two reward functions together with equal weights (=0.5) to achieve the best performance.

| | COOL | STORE | HEAT | CLEAN | SLICE | PREP | TRASH |
|---|---|---|---|---|---|---|---|
| NAVEXP [49] | 0.05 | 0.01 | 0.01 | 0.43 | 0.29 | 0.33 | 0.25 |
| OURS | **0.26** | **0.12** | 0.13 | **0.53** | **0.36** | 0.26 | 0.13 |
| OURS + NAVEXP | 0.25 | 0.05 | **0.19** | 0.50 | 0.34 | **0.41** | **0.26** |

Table S2: **Combined policy experiments.** Navigation exploration offers complementary reward signals that can be combined with our method for stronger performance.

| | ResNet18 + LSTM | | | | ResNet50 + LSTM | | | | ResNet18 + GRU | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | COOL | STORE | HEAT | CLEAN | COOL | STORE | HEAT | CLEAN | COOL | STORE | HEAT | CLEAN |
| VANILLA | 0.07 | 0.00 | 0.02 | 0.29 | 0.04 | 0.00 | 0.01 | 0.26 | 0.31 | 0.03 | 0.03 | 0.38 |
| NAVEXP [49] | 0.02 | 0.02 | 0.01 | 0.44 | 0.02 | 0.00 | 0.00 | **0.42** | 0.00 | 0.00 | 0.03 | 0.35 |
| OURS | **0.30** | **0.16** | **0.11** | **0.55** | **0.15** | **0.11** | **0.12** | 0.36 | **0.52** | **0.31** | **0.17** | **0.73** |

Table S3: **Backbone architecture ablations.** Performance on four interaction-heavy tasks.

## S11 Task-specific results breakdown

We include task level breakdowns of results from Sec. 4.2 of the main paper. These results highlight the strengths and weaknesses of all models on a task-specific level to supplement overall task success results in Table 1 of the main paper. These include Task success vs. training iteration (Fig. S3) and Task success vs. instance difficulty (Fig. S4)

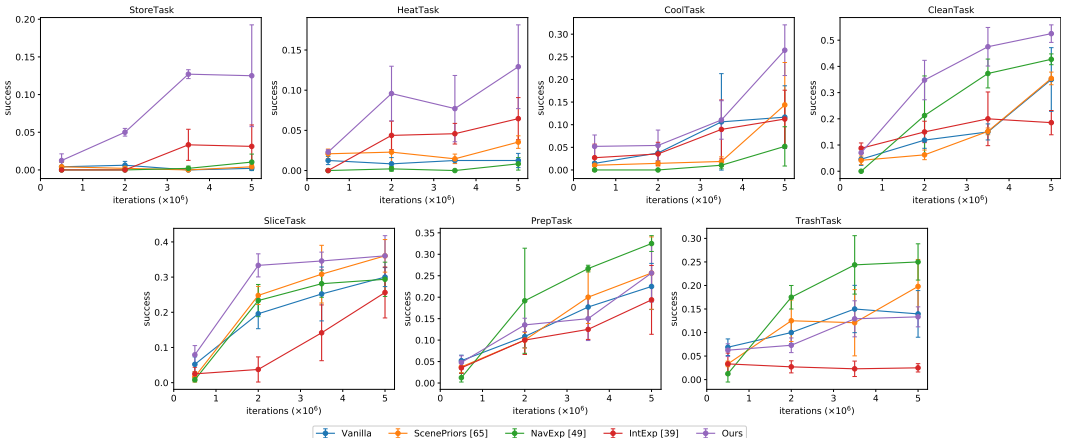


Figure S3: **Task success vs. training iteration.** This is the task-specific version of Fig. 5 (left) that shows convergence rates of all methods.

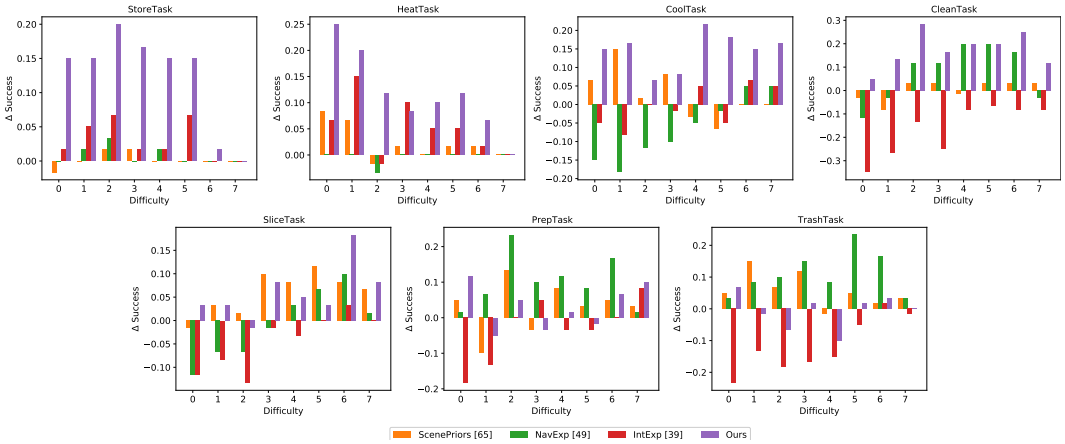


Figure S4: **Task success vs. navigation difficulty.** This is the task-specific version of Fig. 5 (right) that shows improvement in success over the baseline model. Note: we show absolute improvement (instead of relative) as the baseline has zero success for some tasks and some difficulty levels.