Distilling Internet-Scale Vision-Language Models into Embodied Agents

Theodore Sumers [1] [*] Kenneth Marino [2] Arun Ahuja [2] Rob Fergus [2] Ishita Dasgupta [2]

Abstract

Instruction-following agents must ground language into their observation and action spaces. Yet learning to ground language is challenging, typically requiring domain-specific engineering or large quantities of human interaction data. To address this challenge, we propose using pretrained vision-language models (VLMs) to supervise embodied agents. We combine ideas from model distillation and hindsight experience replay (HER), using a VLM to retroactively generate language describing the agent's behavior. Simple prompting allows us to control the supervision signal, teaching an agent to interact with novel objects based on their names (e.g., planes) or their features (e.g., colors) in a 3D rendered environment. Fewshot prompting lets us teach abstract category membership, including pre-existing categories (food vs toys) and ad-hoc ones (arbitrary preferences over objects). Our work outlines a new and effective way to use internet-scale VLMs, repurposing the generic language grounding acquired by such models to teach task-relevant groundings to embodied agents.

1. Introduction

Embodied agents capable of understanding and fulfilling natural language instructions are a longstanding goal for artificial intelligence (Winograd, 1972). Such agents must *ground* language (Harnad, 1990; Mooney, 2008) by correctly associating words with corresponding referents in their environment. But grounding language is both philosophically (Quine, 1960) and practically challenging: methods to learn such groundings remain an active area of research (Tellex et al., 2020).

[*]Work done while interning at DeepMind. [1]Department of Computer Science, Princeton University, Princeton, New Jersey [2]DeepMind, New York City, United States. Correspondence to: Theodore Sumers <sumers@princeton.edu>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

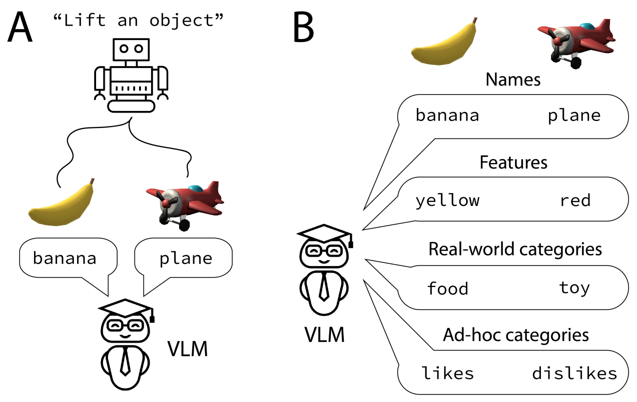


Figure 1. Overview of our approach. **A**: We use a generative VLM to re-label agent trajectories for hindsight experience replay. **B**: Varying the prompt allows us to relabel trajectories along multiple dimensions to teach the embodied agent different tasks.

Embodied language grounding is particularly difficult because training data are scarce. Passive learning from internet data has driven a series of revolutions in natural language processing (Mikolov et al., 2013; Devlin et al., 2018; Brown et al., 2020), but embodied agents must map language into their own idiosyncratic observation and action spaces.

Training data for embodied agents thus typically consist of trajectories paired with linguistic instructions (Tellex et al., 2011; Abramson et al., 2020, *inter alia*) or descriptions (Nguyen et al., 2021; Sharma et al., 2022; Zhong et al., 2022). Providing an agent with aligned behavior and language allows it to learn a mapping between the two. Unfortunately, such data are expensive (typically requiring human annotation), under-specified (a trajectory can correspond to several different instructions or descriptions), and agent-specific (trajectory representations are tightly coupled with the agent architecture).

We propose using large-scale vision-language models (VLMs) to address some of these challenges. Unlike embodied agents, VLMs can be trained on massive internet data (Radford et al., 2021; Ramesh et al., 2022; Alayrac et al., 2022; OpenAI, 2023; Liu et al., 2023). Here, we use the Flamingo VLM (Alayrac et al., 2022) to annotate trajectories, leveraging its internet-derived language grounding to generate training data for the embodied agent (Fig. 1).

Our approach distills the VLM's domain-general language grounding into domain-specific embodied agents, thus ameliorating data cost and scarcity. We use prompting and few-shot learning to control which aspects of a trajectory to relabel, mitigating label under-specification and allowing new task definitions on the fly. Finally, unlike approaches which integrate encoders such as CLIP (Radford et al., 2021) directly into the embodied agent, our method operates purely as data augmentation. This allows us to remain agnostic to changes in the agent architecture and affords substantial interpretability, making interventions and diagnosis easier.

After reviewing related work (Sec. 2), we make the following contributions:

- A novel method using a generative VLM to supervise training of language-conditioned agents (Sec. 3).
- Experiments using our method to flexibly teach new language groundings, including object names (Sec. 5.1), attributes (Sec. 5.2), category membership (Sec. 5.3) and even ad-hoc user preferences (Sec. 5.4).
- An analysis of this imperfect supervision signal, including transferable insight into how different types of noise affect downstream task performance (Sec. 5.5).

Taken together, our work demonstrates that generic language grounding acquired from internet-scale pretraining can be controlled and distilled into embodied agents, allowing us to teach task-specific language groundings without the burden of extensive human supervision.

2. Related Work

Our work studies language grounding within the classic instruction following setting, where an embodied agent is given a natural language instruction and must generate a trajectory satisfying it (Winograd, 1972).[1]

2.1. Learning from Human Interactions

Typical approaches use a human-generated dataset of aligned language and trajectories to learn a mapping between them (Kollar et al., 2010; Tellex et al., 2011; Chen & Mooney, 2011; Artzi & Zettlemoyer, 2013; Mei et al., 2016; Anderson et al., 2018; Blukis et al., 2019; Lynch & Sermanet, 2020; Abramson et al., 2020; Shridhar et al., 2020; Fried et al., 2018, *inter alia*). Language-conditioned reinforcement learning (RL) learns this grounding from trial and error, either assuming an environment-generated reward signal (Misra et al., 2017; Chaplot et al., 2018; Yu et al., 2018), or learning a reward function from aligned language-trajectory data (Bahdanau et al., 2018).

While these works developed agents capable of fulfilling natural language instructions within circumscribed domains, they are limited by training on static environments and scarce, expensive datasets. The resulting learned language groundings are often tightly coupled to the agent's observations and actions, making them inflexible to new objects or concepts. Researchers have explored numerous methods to mitigate this, including data augmentation (Blukis et al., 2020; Chen et al., 2022b), dual-coding memory (Hill et al., 2021), auxiliary language generation objectives (Yan et al., 2022; Bigazzi et al., 2021), or interactive supervision (Kulick et al., 2013; Mohan & Laird, 2014; She et al., 2014; Thomason et al., 2017; Co-Reyes et al., 2018; Chai et al., 2018; Nguyen et al., 2021). Recent approaches instead leverage internet-scale language models to achieve generalization.

2.2. Leveraging Pretrained Models

Several lines of work use language-only knowledge from pretrained models. For example, word embeddings can be used to generalize representation of linguistic instructions (Chen et al., 2020; Li et al., 2019), while language models can be used to break complex instructions into simpler ones (Ahn et al., 2022; Huang et al., 2022a;b; Dasgupta et al., 2022; Singh et al., 2022). Because these approaches use pretrained weights over language only, they can help the agent generalize paraphrases or combinations of existing concepts, but cannot themselves provide groundings for novel visual concepts.

Unlike these language-only models, vision-language models (VLMs) acquire multi-modal grounding from passive internet-scale learning. Contrastive VLMs (e.g. CLIP, Radford et al., 2021) can be used to drive exploration (Tam et al., 2022), construct a semantic representation from visual imagery (Chen et al., 2022a; Singh et al., 2022), or directly as the agent's vision and text encoders, thus inheriting this grounding (Majumdar et al., 2022; Shah et al., 2022; Shridhar et al., 2021; Bucker et al., 2022). However, such pretrained vision encoders are often trained for object recognition and may not encode task-relevant information (e.g., spatial positions, Shridhar et al., 2021). Further, since this information is encoded as hidden vectors, it is nontrivial to determine which representation will be optimal for a particular downstream task (Hsu et al., 2022) or avoid biases arising from their training data (Bommasani et al., 2021). Our approach—using generative VLMs to produce linguistic annotations—is inherently interpretable and allows us to use simple prompts to focus the VLM on task-relevant aspects of the visual scene.

Finally, other approaches have trained new VLMs on internet-scale data to serve as general (Nair et al., 2022) or domain-specific (Fan et al., 2022; Guhur et al., 2021; Hao et al., 2020) visual representations. In contrast, our approach uses an off-the-shelf pretrained VLM to generate task-relevant annotations.

[1]Outside instruction following, a related body of work grounds descriptive language from documents (Branavan et al., 2012; Zhong et al., 2020; 2021) or interactions (Narasimhan et al., 2018; Sumers et al., 2021; Lin et al., 2022) to learn general policies.

2.3. Hindsight Experience Replay

Our approach builds on Hindsight Experience Replay (HER, Andrychowicz et al., 2017) in a language-conditioned setting (Chan et al., 2019). HER's key insight is that early in training, rewards are sparse because agents rarely (if ever) achieve the specified goal. HER densifies rewards by converting these failed trajectories into successful ones by retroactively "relabeling" them, assigning a goal achieved by the agent's behavior.

The central challenge of applying HER to language-conditioned agents is implementing a relabeling function that maps from states to natural language instructions. First, the mapping is not 1:1, as multiple instructions may be compatible with the same state (for example, "Lift a banana" and "Lift something yellow", Fig. 1). Second, the mapping is not known: there is no clear way of converting an agent's state to a linguistic instruction fulfilled by that state.

Prior work has used a reward signal to train a domain-specific relabeling model (Cideron et al., 2020), a mix of image- and human-relabeling (Lynch & Sermanet, 2020), or thousands of human-relabeled trajectories to fine-tune a contrastive VLM which can then be used to select the best instruction from a candidate set (Xiao et al., 2022). We instead use a pretrained generative VLM as the relabeling function, omitting the need for environmental rewards or domain-specific human labelled data. Our approach also offers free-form language generation (rather than selecting from limited options) and on-the-fly task specifications via prompting (allowing us to relabel specific aspects of the trajectory, such that each can have multiple language labels depending on the task specification; Fig. 1B). This allows better use of trajectories.

3. Method

Our approach uses a pretrained generative VLM as the relabeling function for HER (Sec. 2.3). We first formally specify HER. HER assumes that goals $g \in \mathcal{G}$ correspond to predicates $f_g : \mathcal{S} \to \{0,1\}$, so that goals are completed by reaching a set of states. During training, an agent is provided with a goal g and executes a trajectory $\xi = s_0, ...s_T$. Because the goal is rarely accomplished, most trajectories receive no reward: $\forall_{s \in \xi} f_g(s) = 0$. HER solves this by *relabeling* the trajectory, assigning a new goal g' which *was* accomplished by the last state in the sequence: $f_{g'}(s_T) = 1$.[2] The relabeling function m takes a state as input and returns a goal fulfilled by that state $m : \mathcal{S} \to \mathcal{G}$ s.t. $\forall_{\mathrm{s} \in \mathcal{S}} \mathrm{f}_{\mathrm{m(s)}}(\mathrm{s}) = 1$.

In our work, we use a VLM as the relabeling function. We relax the assumption that the environment is fully observable, and consider *observation* sequences $o_1, ..., o_T$ (the agent's visual inputs) generated by an observation function $O : \mathcal{S} \to \Omega$. The VLM serves as a proxy relabeling function $\tilde{m}$, which now takes an observation and returns a goal (a natural language string) satisfied by the state underlying that observation: $\tilde{m} : \Omega \to \mathcal{G}$ s.t. $\forall_{\mathrm{s} \in \mathcal{S}} \mathrm{f}_{\tilde{\mathrm{m}}(\mathrm{O(s)})}(\mathrm{s}) = 1$. Intuitively, this means that we retroactively generate language that describes the agent's actual behavior (Fig. 1A).

This approach is conceptually straightforward and fully compatible with any procedure for training language-conditioned agents, as the original instructions can be directly replaced by the generated ones. Indeed, while HER was developed for reinforcement learning, we use the relabeled trajectories to train with imitation learning instead.

Relative to prior work, our approach makes few assumptions and is highly data efficient. Unlike Cideron et al. (2020) we do not use a reward signal from the environment; and unlike Lynch & Sermanet (2020); Xiao et al. (2022) we do not crowdsource relabeling. However, in settings with a reward signal or preexisting datasets, it would be possible to incorporate such information into our method (e.g., using an annotated dataset to fine-tune the generative VLM).

Finally, while we use a generative VLM, object classification or detection models (Minderer et al., 2022; Kuo et al., 2022; Kirillov et al., 2023) may be used instead. Such models have two important structural limitations: they require *a priori* specification of possible goals (unlike our zero-shot experiments in Sections 5.1-5.3) and cannot be used for ad-hoc categories (Section 5.4). Empirically, we found that the classification / detection paradigm was less effective than our generative approach: substituting the OWL-ViT detection model (Minderer et al., 2022) as a relabeling function yielded substantially worse performance (Appendix B).

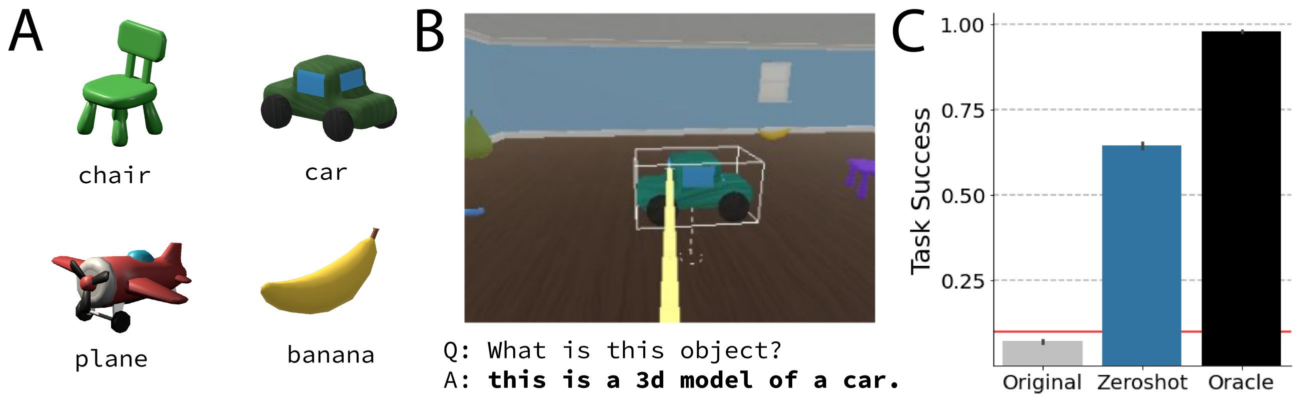
4. Experimental Setup

In this section, we describe our experimental framework employing a VLM to relabel trajectories for HER. Following the classical HER formulation (Andrychowicz et al., 2017) we use a simple task structure which can be re-labeled on the basis of the final observation only. We return to extensions requiring multiple observations in the discussion.

4.1. The Playhouse Environment

We use the Playhouse environment from Abramson et al. (2020), a Unity-based environment with a continuous action space. Each episode takes place in a procedurally-generated home with multiple rooms and a wide range of everyday domestic objects (Fig. 2A). This 3D environment is challenging for RL agents and the VLM, as the agent's egocentric perspective often yields unusual or close-up perspectives on objects (Figs. S3, S4, S5).

[2]Alternative HER formulations relabel randomly sampled states. In our setting, the last state is generally the most informative, but applications to other domains such as vision-language navigation may benefit from such strategies.



Q: What is this object?
A: **this is a 3d model of a car.**

Figure 2. Using a pretrained VLM to teach an agent novel object names (Sec. 5.1). **A**: Four of the ten objects. The agent has never seen these words before, and the VLM is not given any information about what might be present. **B**: We use an open-ended prompt to relabel trajectories, then use the VLM outputs (bold) to retrain the agent. **C**: Results from the original agent ("Original"), after retraining with VLM labels ("Zeroshot"), and an upper bound from retraining on ground truth labels from the environment ("Oracle"). Here and throughout, the red line shows baseline performance (lifting a random object) and error bars show 95% CIs.

4.2. The "Lift" Task

To isolate the effects of relabeling we use the same task structure across our experiments. We chose the "Lift" task as it gives a direct measure of our method's effectiveness: we can evaluate the agent's language grounding on an object-by-object basis.

At the start of each episode, agents are placed in a room with 5-10 objects and instructed to lift a target object. To demonstrate the flexibility of our method, we vary the task specification: using object names ("Lift a plane"; Sec 5.1), attributes ("Lift a red object"; Sec 5.2), categories ("Lift a toy"; Sec 5.3), or preferences ("Lift something John Doe likes"; Sec 5.4). Episodes end when the agent lifts an object, or after 120 seconds.

4.3. Flamingo VLM

We use Flamingo (Alayrac et al., 2022), a state-of-the-art language-generative VLM. Flamingo accepts interleaved images and text as input, and produces text output. This allows us to experiment with both "Zeroshot" prompts (containing only the image to be relabeled and a text prompt) and "Fewshot" prompts (including up to 32 in-context image-text examples). We use the 80B parameter model described by Alayrac et al. (2022) with greedy sampling.

4.4. HER Implementation

We first need an agent that generates structured behaviors that can be interpretably relabelled. We use human-human data to learn a task-agnostic motor policy: e.g., an agent that knows how to *lift* something, but not what a *plane* is. We refer to this as the "original" agent, and train it via behavioral cloning (BC) on the human-human dataset described by Interactive Agents Team (2021); for details on the dataset, agent architecture, and BC implementation, please refer to that work. To ensure the "original" agent lacks task-relevant groundings, we filter out episodes with relevant utterances before performing BC. However, we note that the "original" agent could be generated in any way, such as with RL on a different set of tasks; our approach uses it as a starting point.

To compare the effects of different relabeling functions, we use a batched HER approach (Fig. S1). For each experiment, we generate an initial set of approximately 10,000 trajectories with a generic "Lift an object" instruction (due to implementation details, the actual number varied from 10,000 to 11,500). Across all experiments, around 3% of these initial trajectories timed out as the agent did not lift an object. This was not enough to meaningfully affect results, so for simplicity we discarded them. This is equivalent to assuming the agent can detect a successful grasp, which is reasonable even in robotics (Pinto & Gupta, 2016).

We then use the VLM to relabel the final image in each trajectory. This generates a new annotation describing the agent's actual behavior (Fig. 1A, 2B). We perform light post-processing on the VLM outputs: Flamingo sometimes generates multiple responses (always separated by a newline), so we truncate the response to the first newline and prepend "Lift a ". We then use these VLM-generated strings as the instruction in a second round of BC. Finally, performing BC on the full trajectories is computationally expensive. Preliminary analysis showed that using the full trajectories provides only minor performance gains compared to truncating them to the last 5 seconds; we therefore adopt this truncation throughout.

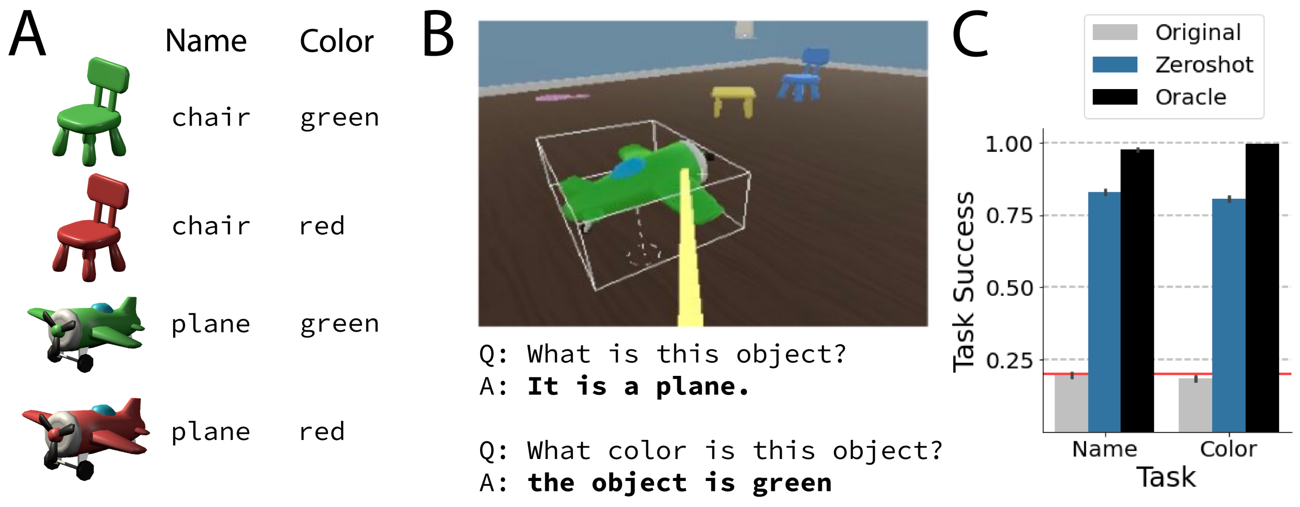


Figure 3. Using a pretrained VLM to flexibly teach object attributes (name or color) from a single set of trajectories (Sec. 5.2). **A**: We randomize color mappings so each object can appear red, green, blue, pink, or yellow. **B**: We again use generic prompts, adding a second color-oriented prompt to obtain color labels. **C**: Results on the "Name" and "Color" tasks from the "Original" agent and after retraining with VLM labels ("Zeroshot") or ground truth labels from the environment ("Oracle").

5. Results

To demonstrate the flexibility of VLMs for relabeling, we vary the goal structure of the "Lift" task. Our first two experiments focus on visible object attributes, using simple "Zeroshot" prompts to teach object names (Sec. 5.1) and attributes (Sec. 5.2). Our third and fourth use "Fewshot" prompts to teach category membership (Sec. 5.3) and finally novel user preferences over objects (Sec. 5.4). We close with an analysis of label noise and task performance (Section 5.5). This highlights an advantage of our approach over contrastive VLMs (Radford et al., 2021): our method produces human-legible language annotations with corresponding confidence scores, allowing us to analyze and filter labels to improve downstream task performance.

5.1. Teaching Object Names

Our first experiment uses relabeling to teach the agent to lift one of 10 objects: a table, a chair, a book, a basketball, a racket, a plane, a car, a banana, a carrot, and a pear (Fig. 2A).

Setup. We begin by training an "original" agent, filtering out any episodes containing one of the 10 target words.[3] We use this agent to generate 10,000 initial trajectories using the generic "Lift a object" instruction in a room with all 10 objects. We provide the VLM with the final image in each trajectory and use a simple QA-style zero-shot prompt: `[IMG_0] Q: What is this object? A:` (Fig. 2B). We then re-train the agent on these new VLM labels.

Results. We test our retrained agent by again placing it in a room with all 10 objects, but now instructing it to lift a specific one (e.g., "Lift a car") and generate 10,000 evaluation trajectories. We first test the "Original" agent to ensure it has no knowledge of the objects. We find that it achieves near-chance performance (7.1% task success), confirming that it does not possess task-relevant language groundings. We then test an "Oracle" agent retrained on ground-truth relabeling from the environment itself. This agent performs near ceiling (97.9% success), confirming there are enough trajectories to teach the task with perfect relabeling. Finally, we find that our VLM-based relabeling method—using only a simple zero-shot prompt and no information about what objects might appear—conveys a significant fraction of the task, resulting in 64.4% success (Fig. 2C). The VLM-retrained agent performs well above chance on all ten objects (Fig. S2).

Intriguingly, we achieve this performance despite substantial noise in the relabeling. VLM-generated strings are relatively low accuracy (only 54.7% contain the canonical object name used in instructions), and frequently contain extraneous words (Fig. 2B, S3; Table S1). We conduct a deeper analysis of label noise and downstream task performance in Sec. 5.5.

5.2. Teaching Object Attributes

Section 5.1 demonstrated that VLMs can teach basic object names. But how controllable is the VLM relabeling? Can we, for example, teach an agent to recognize an object's *attributes* rather than the objects themselves? We now show that the VLM can be used to relabel a single set of trajectories with their *name* or *color* respectively.

Setup. We use a subset of five objects from Section 5.1 (plane, racket, chair, table, and basketball). Previously, these

[3]Due to the tokenizer used in the agent, we filtered out all episodes containing the substring "ball" rather than "basketball."

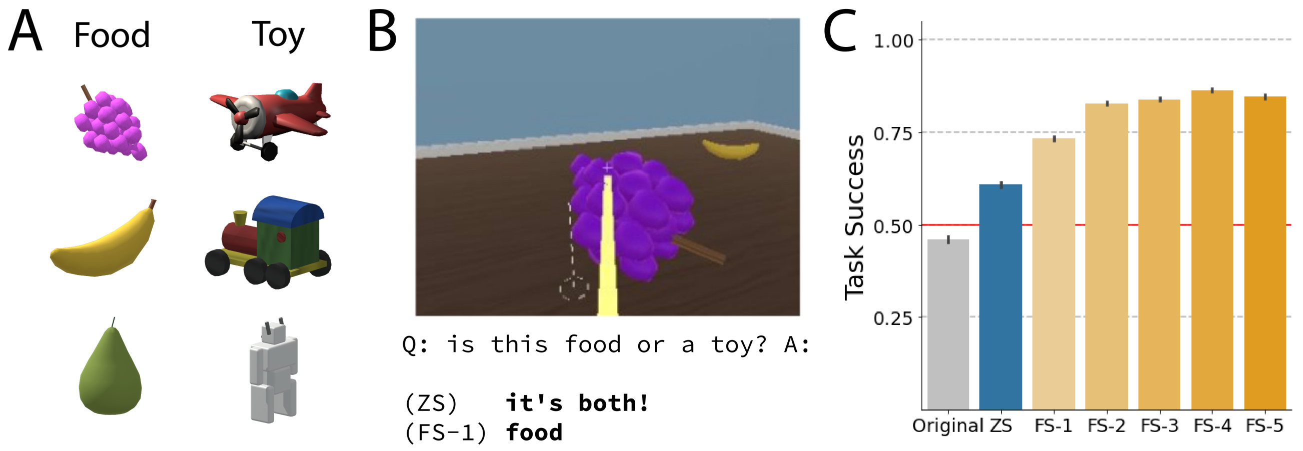


Figure 4. Using a pretrained VLM to teach category structure (Sec. 5.3). **A**: We use a set of 10 objects, 5 "food" and 5 "toys." **B**: We experiment with both "Zeroshot" (ZS) and "Fewshot" (FS) prompting. We vary the number of fewshot examples from each category from one (FS-1 is given 3 examples each of carrots and robots) to all five (FS-5 is given 3 examples each of all 5 objects in each category). **C**: Results for the "Original" agent and after retraining with different relabelings. Performance increases substantially from Zeroshot to FS-2 then plateaus. This suggests that partial information about the category is sufficient for Flamingo to extrapolate to new food and toys.

items' colors were fixed across all episodes. We now render them in different colors (red, green, blue, pink, or yellow). This gives us a total of 25 object-color combinations (green chair, red chair, green plane, etc.; Fig. 3A). As before, we train an original agent filtering out episodes containing any of these object name or color terms. We create a level with all five objects, and randomly assign each color to a object in each episode. We use this task to generate a set of 10,000 trajectories with a generic "Lift an object" prompt.

Now, however, we relabel each trajectory twice (Fig. 3B, S4). One relabeling uses the original prompt from Section 5.1: `[IMG_0] Q: What is this object? A:` . The second introduces a slight variation, adding the word "color": `[IMG_0] Q: What color is this object? A:` . We follow the same procedure and retrain two separate agents: one using the labels generated by the original prompt, and one using the "color" variation.

Results. We again test our agents by placing them in a room with all five objects and instructing them to lift one. However, we now test two forms of instructions: an object name task ("Lift a {plane, racket, chair, table, basketball}") or an object color task ("Lift a {red, green, blue, pink, yellow} object"). We use the same comparisons, checking the original agent and an agent retrained on oracle color and object name labels.

We again find that the original agent performs at chance (19.6% task success on names and 18.4% on colors) and the oracle-relabeled agent performs at ceiling (97.6% on names and 99.5% on color). Our VLM relabeling achieves 83.0% and 80.6% respectively (Fig. 3C).

5.3. Teaching Real-World Categories

Sections 5.1 and 5.2 show the VLM can be used to re-label visual object attributes such as shape and color. But many important properties, such as category membership, are not readily visible. We next test whether our method can be used for tasks depending on such properties.

Setup. We use a set of 10 items, five "food" (pear, banana, carrot, lemon, and grapes) and five "toys" (plane, train, car, robot, dice; Fig. 4A). We train an original agent, filtering out episodes with references to any of these, as well as "food" or "toy." Again, we generate a set of initial trajectories using a "Lift an object" prompt in a room with all 10 objects.

We again use a simple prompt to re-label the trajectories: `[IMG_0] Q: Is this food or a toy? A:` . However, initial results suggested that these categories caused an interaction effect with the 3D rendered graphics: Flamingo recognized that the "food" items were really *toy* food items and frequently labeled them as "toys" or "both" (Fig. 4B, Table S2). This domain shift issue could be obviated by using a different category structure (e.g. food vs furniture), but we instead experimented with few-shot prompting. We generated three example images of the agent lifting each of the ten items and ran a series of relabelings with incrementally more examples. Our "Fewshot-1" prompt gives Flamingo three examples each of one food and one toy (carrots and robots) for a total of 6 in-context examples. "Fewshot-2" added lemons and dice (a total of 12 examples); "Fewshot-3" added planes and bananas (total of 18); "Fewshot-4" added grapes and cars (total of 24), and finally "Fewshot-5" added trains and pears for a total of 30. "Fewshot-5" therefore saw three examples each of all 10 objects. Few-shot relabeling resulted in short VLM genera-

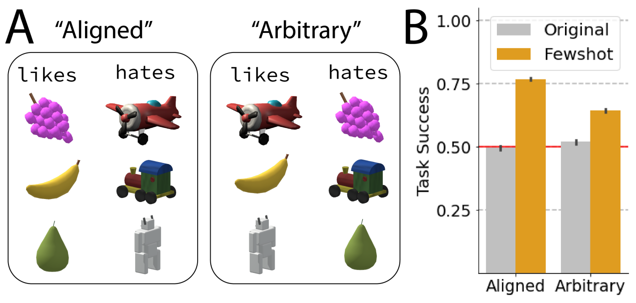


Figure 5. Using a VLM to teach ad-hoc categories (Sec. 5.4). **A**: "Aligned" preferences follow existing category structure, while "Arbitrary" preferences cut across it. **B**: The VLM is able to teach tasks requiring new category structure from fewshot examples, but alignment with existing structure helps.

tions with no extraneous words (Fig. 4B, S5, Table S3).

As before, we re-train agents with each of the resulting label sets. We evaluate them by generating 10,000 rollouts each for "Lift a food" and "Lift a toy" instructions.

Results. As expected, our original agent performs near chance (46.0% task success). Zeroshot relabeling lifts performance above chance to 60.8%. Adding in-context examples provides another substantial boost: "Fewshot-1" lifts performance to 73.3%, and "Fewshot-2" to 82.8% (Fig. 4C). Adding additional examples has only a marginal effect, with performance plateauing around 85%. Notably, even in the "Fewshot-2" condition, Flamingo is readily able to generalize category structure to the remaining three items in each category. This suggests that the few-shot examples help Flamingo adapt to the 3D rendered visuals, while Flamingo inherits information about category membership from its pretraining experience.

5.4. Teaching Ad-hoc Categories

Our first three experiments show that simple prompting and a handful of few-shot examples allow our VLM to recognize and teach canonical properties such as names, colors, and category membership. In general, however, we cannot expect tasks to conform to existing canonical categories. Evidence from psychology suggests that *ad-hoc categories* (Barsalou, 1983) play a crucial role in human cognition, allowing us to create new conceptual groupings on the fly. Such context-dependent categories are often based on usecases ("things to take on a camping trip") or affordances ("things that can be used as firewood in an emergency"). Can our method be used to teach such flexible category structures?

Our final experiment tests the VLM's ability to re-label based on new and arbitrary category structure: here, instantiated as a user's preferences over a set of objects. Such dynamic relabeling would allow individuals to provide personalized task specifications. For example, a user could provide a list of items to bring on a camping trip, or express preferences (or allergies) over food items. The agent could then re-label its previous experience with such items and re-train its policy to learn the new groundings, allowing it to map user-level requests into its action space.

Setup. We aim to test Flamingo's ability to learn (and then teach) new categories via in-context examples. We re-use the 10 objects from Sec. 5.3 but re-formulate the task in terms of user preferences. We introduce two sets of preferences: "Aligned" preferences, which respect existing category structure (John Doe likes food and dislikes toys), and "Arbitrary" preferences, which cut across it (John Doe likes robots, planes, carrots, lemons, and bananas; and dislikes cars, dice, trains, grapes, and pears; Fig. 5A).

We use the 10,000 rollouts generated in Section 5.3 but re-label them with new prompts: `[IMG_0] Q: Would John Doe like this? A:`. We include preambles `John Doe likes food.` for "Aligned" and `John Doe likes robots, planes, carrots, lemons, and bananas.` for "Arbitrary". We use fewshot examples for all 10 items (equivalent to "Fewshot-5" from Sec. 5.3) with "yes" or "no" responses according to the category structure being used. To produce task-appropriate labels, we transform Flamingo's response by mapping "yes" to "an object John Doe likes" and "no" to "an object John Doe hates."

We re-train agents using these labels, and again evaluate by averaging 10,000 rollouts on the two tasks: "Lift something John Doe likes", and "Lift something John Doe hates."

Results. Our original agent performs near chance (50%) for both category structures (Fig. 5B). We find that Flamingo relabeling improves performance for both category structures, with "Aligned" (76.7%) outperforming "Arbitrary" (64.2%). These results demonstrate that Flamingo is aided by, but not solely dependent on, real-world category structure.

It is helpful to compare these results with the "Fewshot-5" results from Section 5.3. These experiments each use the same few-shot examples to relabel the same trajectories, but vary the nature of the category structure. Flamingo does very well with *explicit* real-world category structure ("Food-Toy" in Sec. 5.3; task performance 84.7%); next best with *implicit* real-world category structure ("Aligned" in Sec. 5.4; 76.7%); and modestly with *no* real-world category structure ("Arbitrary" in Sec. 5.4; 64.2%).

Finally, we tested the VLM's ability to generalize categories to new instances by permuting the toys' colors. We found that for both preference structures, the VLM successfully relabeled recolored objects with equivalent accuracy (Appendix C).

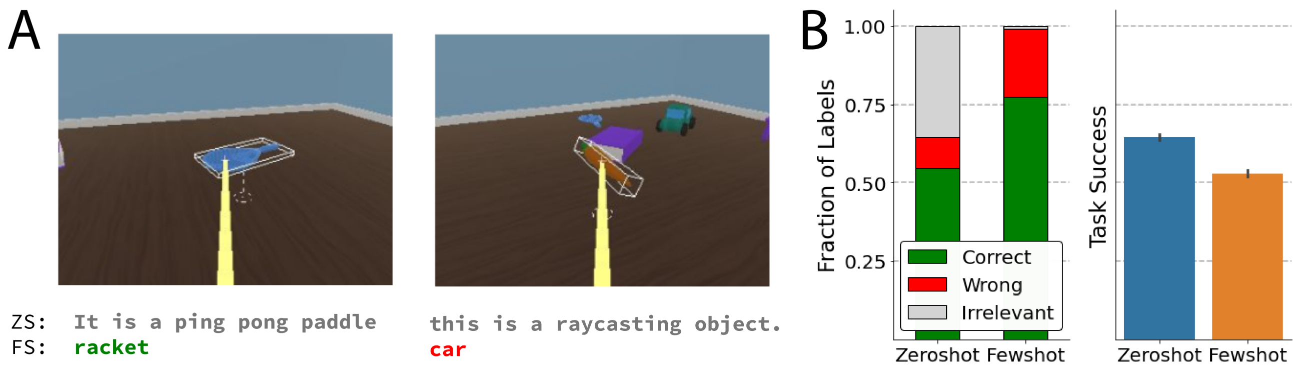


Figure 6. Comparing "Zeroshot" (ZS) and "Fewshot" (FS) relabeling (Sec. 5.5). **A**: Zeroshot typically generates text that reflects the image contents, but is often not task-relevant. Fewshot prompting encourages the VLM to generate one of the 10 object labels. However, when the foreground is challenging it often labels background objects instead. **B**: Label quality and task performance. Zeroshot yields many irrelevant labels; Fewshot instead produces more "Correct" and "Wrong" labels. Agents trained on Fewshot labels perform worse.

5.5. Analyzing the Relabeling Function

In this final section, we use the setup from Section 5.1 to conduct an analysis of Flamingo as a relabeling function.

5.5.1. Zeroshot vs Fewshot Flamingo

We conduct a "Fewshot" relabeling version of "Zeroshot" experiments in Section 5.1, with 32 examples (3-4 examples each of the 10 items in the task) included in context. We find that "Fewshot" relabeling increases accuracy, but—surprisingly—*decreases* downstream task performance. Concretely, 77.2% of "Fewshot" labels contain the canonical object name (compared to 54.7% of "Zeroshot"), yet the "Fewshot"-retrained agent achieves only 52.9% task success (compared to 64.4% for the "Zeroshot" agent; Fig. 6).

A closer look at the labels reveals an important difference. "Zeroshot" labels that are not correct are often *irrelevant* (i.e. they do not contain any of the 10 task-relevant object names). In contrast, incorrect "Fewshot" labels are almost always *wrong* (i.e., they contain a different task-relevant object name). Few-shot prompting encourages Flamingo's generation towards task relevant labels, causing it to "guess" a label when uncertain (Fig. 6B, S3, S7). While "Zeroshot" Flamingo is noisier, these irrelevant labels have little effect on downstream performance. In contrast, "Fewshot" generates wrong task-relevant labels that actively interfere with grounding.

Because many "Zeroshot" relabelings are irrelevant, "Fewshot" is higher *accuracy* ($\frac{\text{\# of correct labels}}{\text{\# of trajectories}}$) but lower *precision* ($\frac{\text{\# of correct labels}}{\text{\# of task-relevant labels}}$). Therefore "Zeroshot" Flamingo produces more *reliable* data: more examples relabeled "Lift a car" will actually reflect the appropriate behavior.

While initial results suggest that "Fewshot" is actually a worse relabeling function, we find that "Fewshot" provides an important benefit: it helps calibrate Flamingo's confidence in its relabeling (Fig. S6). We experiment with filtering out low-confidence labels by progressively dropping the lowest decile. We find that this filtering dramatically increases "Fewshot" precision, but only slightly increases "Zeroshot" precision (Fig. 7A). We filter out the least-confident 50% of the labels and retrain agents on the remaining trajectories. This substantially improves "Fewshot" performance (from 52.9% to 78.6%) while only marginally improving "Zeroshot" (64.4% to 66.1%; Fig. 7B).

5.5.2. Label Properties and Task Performance

These results suggest that relabeling *precision* may be more important than *accuracy* for HER. We quantify this by looking at task success resulting from different label sets.

Formally, we are interested in whether label accuracy or label precision is a better predictor of downstream task performance. We have 10 tasks (each of the individual objects used in Section 5.1, i.e. "Lift a car", "Lift a plane"), and four sets of labels for each (Zeroshot, Zeroshot filtered, Fewshot, and Fewshot filtered). We use a mixed-effects linear regression to predict task success for each of these 40 data points, with fixed effects of label precision and recall, and random effects for each of the 10 tasks. We find that both accuracy and precision are significant, but the effect size of precision is nearly ten times that of accuracy (accuracy: $\beta = .16, t(35.88) = 3.416, p < .01$; precision: $\beta = 1.44, t(35.76) = 11.5, p < 1e-10$; see Table S4). Fig 7C plots regression lines for accuracy and precision respectively. This result yields the valuable general insight that relabeling precision is more important than relabeling accuracy for downstream task performance with HER.

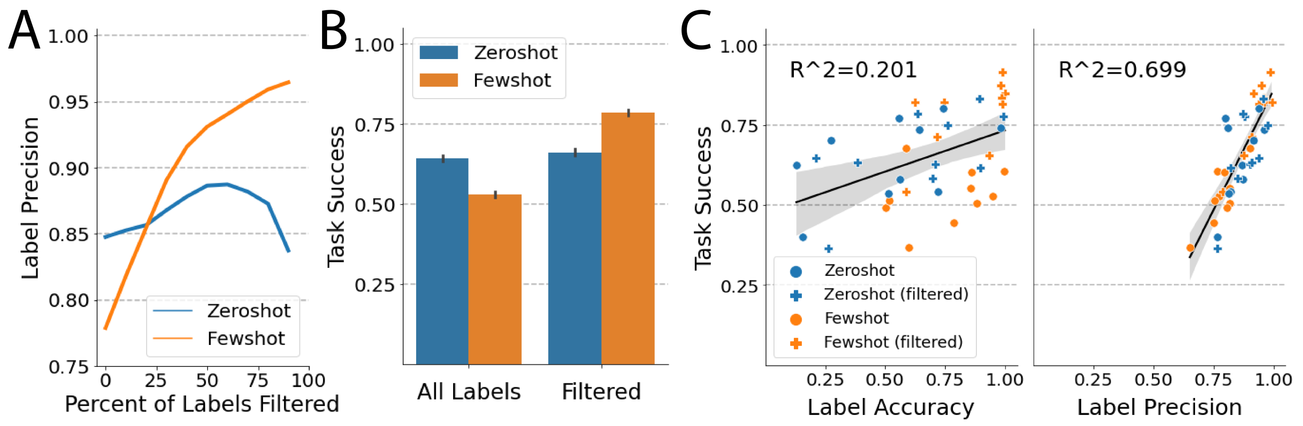


Figure 7. Analysis of label noise and downstream performance (Sec. 5.5). **A**: Filtering out low-confidence labels dramatically increases "Fewshot" label precision, but only marginally improves "Zeroshot." **B**: Filtering out the least confident 50% of labels and re-training agents results in a substantial performance gain for "Fewshot" but not "Zeroshot." **C**: Analysis of label characteristics against downstream task performance on a per-object basis suggests that relabeling precision is more important than accuracy.

6. Discussion

In this work, we used a VLM pretrained on internet data to teach an embodied agent language groundings. We used prompting and fewshot learning to guide the VLM's text generation, focusing it on specific dimensions of the visual stream. This allows us to flexibly distill task-relevant subsets of the VLM's language grounding into the embodied agent.

We note several limitations to our work. First, we focused on English instructions; future work could experiment with pretrained translation models to develop multilingual grounded agents. Second, we demonstrated our method within the classic HER formulation (Andrychowicz et al., 2017) and thus used a task structure that permits relabeling of the final observation only. We additionally used the dataset from Interactive Agents Team (2021) to learn a low-level motor policy before using our method to teach task semantics. Future work could extend our method beyond traditional HER, using a VLM to annotate observation pairs or full videos. Such extensions would facilitate teaching temporally-extended tasks, making our method suitable for training motor policies in addition to task semantics (e.g., teaching both *how* to lift an object and *which* object to lift). Finally, the Flamingo model (Alayrac et al., 2022) used in this work is not publicly available. However, the recently released GPT-4 (OpenAI, 2023) and open-source Prismer models provide comparable capabilities (Liu et al., 2023). Object classification or detection models (Minderer et al., 2022) may also be used to replicate a subset of our method's functionality.

Future work may experiment with other ways to leverage pretrained generative VLMs for embodied agents. Following our offline supervision approach, VLMs could provide a reward signal during training on the basis of vision and text alignment. Alternatively, VLMs could be used directly in the agent. While large model size[4] might mitigate their use for low-level motor policies, they could be used to produce language observations for input to other models (Zeng et al., 2022; Huang et al., 2022b; Dasgupta et al., 2022). Our method may also be straightforwardly applied to vision-language navigation, another popular testbed for embodied agents (Anderson et al., 2018). Overall, the confluence of more naturalistic environments (Shridhar et al., 2020; Savva et al., 2019; Li et al., 2021) with strong and flexible pretrained VLMs (Alayrac et al., 2022; Liu et al., 2023; OpenAI, 2023) makes these models an appealing source of domain-general language groundings. We hope that our method spurs further research leveraging their strengths for embodied agents.

Acknowledgements

We thank Christine Kaeser-Chen, Nathaniel Wong, Fede Carnevale, Alexandre Fréchette, Felix Hill, and the DeepMind NYC team for their advice and assistance.

References

Abramson, J., Ahuja, A., Barr, I., Brussee, A., Carnevale, F., Cassin, M., Chhaparia, R., Clark, S., Damoc, B., Dudzik, A., et al. Imitating interactive intelligence. *arXiv preprint arXiv:2012.05672*, 2020.

Ahn, M., Brohan, A., Brown, N., Chebotar, Y., Cortes, O., David, B., Finn, C., Gopalakrishnan, K., Hausman, K., Herzog, A., et al. Do as I can, not as I say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[4]Our VLM (Alayrac et al., 2022) contains 80B parameters, while our agent (Interactive Agents Team, 2021) contains 57M.

Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., Ring, R., Rutherford, E., Cabi, S., Han, T., Gong, Z., Samangooei, S., Monteiro, M., Menick, J., Borgeaud, S., Brock, A., Nematzadeh, A., Sharifzadeh, S., Binkowski, M., Barreira, R., Vinyals, O., Zisserman, A., and Simonyan, K. Flamingo: a visual language model for few-shot learning. In *Advances in Neural Information Processing Systems*, 2022.

Anderson, P., Wu, Q., Teney, D., Bruce, J., Johnson, M., Sünderhauf, N., Reid, I., Gould, S., and van den Hengel, A. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *CVPR*, pp. 3674–3683, 2018.

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Pieter Abbeel, O., and Zaremba, W. Hindsight experience replay. *Advances in neural information processing systems*, 30, 2017.

Artzi, Y. and Zettlemoyer, L. Weakly supervised learning of semantic parsers for mapping instructions to actions. *Transactions of the Association for Computational Linguistics*, 1:49–62, 2013.

Bahdanau, D., Hill, F., Leike, J., Hughes, E., Hosseini, A., Kohli, P., and Grefenstette, E. Learning to understand goal specifications by modelling reward. In *International Conference on Learning Representations*, 2018.

Barsalou, L. W. Ad hoc categories. *Memory & cognition*, 11:211–227, 1983.

Bigazzi, R., Landi, F., Cornia, M., Cascianelli, S., Baraldi, L., and Cucchiara, R. Explore and explain: Self-supervised navigation and recounting. In *25th International Conference on Pattern Recognition (ICPR)*, pp. 1152–1159, 2021.

Blukis, V., Terme, Y., Niklasson, E., Knepper, R. A., and Artzi, Y. Learning to map natural language instructions to physical quadcopter control using simulated flight. In *CoRL*, 2019.

Blukis, V., Knepper, R. A., and Artzi, Y. Few-shot object grounding and mapping for natural language robot instruction following. *arXiv preprint arXiv:2011.07384*, abs/2011.07384, 2020.

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Branavan, S., Silver, D., and Barzilay, R. Learning to win by reading manuals in a Monte-Carlo framework. *Journal of Artificial Intelligence Research*, 43:661–704, 2012.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T. J., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. *ArXiv*, abs/2005.14165, 2020.

Bucker, A. F. C., Figueredo, L. F. C., Haddadin, S., Kapoor, A., Ma, S., Vemprala, S., and Bonatti, R. LaTTe: Language trajectory transformEr. *ArXiv*, abs/2208.02918, 2022.

Chai, J. Y., Gao, Q., She, L., Yang, S., Saba-Sadiya, S., and Xu, G. Language to action: Towards interactive task learning with physical agents. In *IJCAI*, pp. 2–9, 2018.

Chan, H., Wu, Y., Kiros, J., Fidler, S., and Ba, J. Actrce: Augmenting experience via teacher's advice for multi-goal reinforcement learning. *arXiv preprint arXiv:1902.04546*, 2019.

Chaplot, D. S., Sathyendra, K. M., Pasumarthi, R. K., Rajagopal, D., and Salakhutdinov, R. Gated-attention architectures for task-oriented language grounding. In *AAAI*, 2018.

Chen, B., Xia, F., Ichter, B., Rao, K., Gopalakrishnan, K., Ryoo, M. S., Stone, A., and Kappler, D. Open-vocabulary queryable scene representations for real world planning. *arXiv preprint arXiv:2209.09874*, 2022a.

Chen, D. and Mooney, R. Learning to interpret natural language navigation instructions from observations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 25, pp. 859–865, 2011.

Chen, S., Guhur, P.-L., Tapaswi, M., Schmid, C., and Laptev, I. Learning from unlabeled 3d environments for vision-and-language navigation. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIX*, pp. 638–655. Springer, 2022b.

Chen, V., Gupta, A. K., and Marino, K. Ask your humans: Using human instructions to improve generalization in reinforcement learning. *ICLR*, 2020.

Cideron, G., Seurin, M., Strub, F., and Pietquin, O. Higher: Improving instruction following with hindsight generation for experience replay. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 225–232. IEEE, 2020.

Co-Reyes, J. D., Gupta, A., Sanjeev, S., Altieri, N., Andreas, J., DeNero, J., Abbeel, P., and Levine, S. Guiding policies with language via meta-learning. In *International Conference on Learning Representations*, 2018.

Dasgupta, I., Kaeser-Chen, C., Marino, K., Ahuja, A., Babayan, S., Hill, F., and Fergus, R. Collaborating with language models for embodied reasoning. In *Second Workshop on Language and Reinforcement Learning*, 2022.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. N. BERT: Pre-training of deep bidirectional transformers for language understanding. 2018. URL `https://arxiv.org/abs/1810.04805`.

Fan, L. J., Wang, G., Jiang, Y., Mandlekar, A., Yang, Y., Zhu, H., Tang, A., Huang, D.-A., Zhu, Y., and Anandkumar, A. MineDojo: Building open-ended embodied agents with internet-scale knowledge. *Advances in Neural Information Processing Systems*, 2022.

Fried, D., Hu, R., Cirik, V., Rohrbach, A., Andreas, J., Morency, L.-P., Berg-Kirkpatrick, T., Saenko, K., Klein, D., and Darrell, T. Speaker-follower models for vision-and-language navigation. *Advances in Neural Information Processing Systems*, 31, 2018.

Guhur, P.-L., Tapaswi, M., Chen, S., Laptev, I., and Schmid, C. Airbert: In-domain pretraining for vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1634–1643, 2021.

Hao, W., Li, C., Li, X., Carin, L., and Gao, J. Towards learning a generic agent for vision-and-language navigation via pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13137–13146, 2020.

Harnad, S. The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42(1-3):335–346, 1990.

Hill, F., Tieleman, O., von Glehn, T., Wong, N., Merzic, H., and Clark, S. Grounded language learning fast and slow. In *International Conference on Learning Representations*, 2021.

Hsu, K., Lum, T. G. W., Gao, R., Gu, S. S., Wu, J., and Finn, C. What makes certain pre-trained visual representations better for robotic learning? In *NeurIPS 2022 Foundation Models for Decision Making Workshop*, 2022.

Huang, W., Abbeel, P., Pathak, D., and Mordatch, I. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. *ArXiv*, abs/2201.07207, 2022a.

Huang, W., Xia, F., Xiao, T., Chan, H., Liang, J., Florence, P., Zeng, A., Tompson, J., Mordatch, I., Chebotar, Y., Sermanet, P., Jackson, T., Brown, N., Luu, L., Levine, S., Hausman, K., and Ichter, B. Inner monologue: Embodied reasoning through planning with language models. In *6th Annual Conference on Robot Learning*, 2022b.

Interactive Agents Team, D. Creating multimodal interactive agents with imitation and self-supervised learning. *arXiv preprint arXiv:2112.03763*, 2021.

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., et al. Segment anything. *arXiv preprint arXiv:2304.02643*, 2023.

Kollar, T., Tellex, S., Roy, D., and Roy, N. Toward understanding natural language directions. In *2010 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pp. 259–266. IEEE, 2010.

Kulick, J., Toussaint, M., Lang, T., and Lopes, M. Active learning for teaching a robot grounded relational symbols. In *IJCAI*, pp. 1451–1457. Citeseer, 2013.

Kuo, W., Cui, Y., Gu, X., Piergiovanni, A., and Angelova, A. F-vlm: Open-vocabulary object detection upon frozen vision and language models. *arXiv preprint arXiv:2209.15639*, 2022.

Li, C., Xia, F., Martín-Martín, R., Lingelbach, M., Srivastava, S., Shen, B., Vainio, K., Gokmen, C., Dharan, G., Jain, T., et al. igibson 2.0: Object-centric simulation for robot learning of everyday household tasks. *arXiv preprint arXiv:2108.03272*, 2021.

Li, X., Li, C., Xia, Q., Bisk, Y., Celikyilmaz, A., Gao, J., Smith, N. A., and Choi, Y. Robust navigation with language pretraining and stochastic sampling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 1494–1499, 2019.

Lin, J., Fried, D., Klein, D., and Dragan, A. Inferring rewards from language in context. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8546–8560, Dublin, Ireland, May 2022. Association for Computational Linguistics.

Liu, S., Fan, L., Johns, E., Yu, Z., Xiao, C., and Anandkumar, A. Prismer: A vision-language model with an ensemble of experts. *arXiv preprint arXiv:2303.02506*, 2023.

Lynch, C. and Sermanet, P. Grounding language in play. *arXiv preprint arXiv:2005.07648*, 2020.

Majumdar, A., Aggarwal, G., Devnani, B. S., Hoffman, J., and Batra, D. ZSON: Zero-shot object-goal navigation using multimodal goal embeddings. In *Advances in Neural Information Processing Systems*, 2022.

Mei, H., Bansal, M., and Walter, M. R. Listen, attend, and walk: Neural mapping of navigational instructions to action sequences. In *AAAI*, 2016.

Mikolov, T., Yih, W.-t., and Zweig, G. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pp. 746–751, 2013.

Minderer, M., Gritsenko, A., Stone, A., Neumann, M., Weissenborn, D., Dosovitskiy, A., Mahendran, A., Arnab, A., Dehghani, M., Shen, Z., Wang, X., Zhai, X., Kipf, T., and Houlsby, N. Simple open-vocabulary object detection. In *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part X*, pp. 728–755. Springer-Verlag, 2022.

Misra, D., Langford, J., and Artzi, Y. Mapping instructions and visual observations to actions with reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1004–1015, 2017.

Mohan, S. and Laird, J. Learning goal-oriented hierarchical tasks from situated interactive instruction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28, 2014.

Mooney, R. J. Learning to connect language and perception. In *AAAI*, pp. 1598–1601. Chicago, 2008.

Nair, S., Rajeswaran, A., Kumar, V., Finn, C., and Gupta, A. R3M: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

Narasimhan, K., Barzilay, R., and Jaakkola, T. Grounding language for transfer in deep reinforcement learning. *J. Artif. Int. Res.*, 63(1):849–874, sep 2018. ISSN 1076-9757.

Nguyen, K. X., Misra, D., Schapire, R., Dudík, M., and Shafto, P. Interactive learning from activity description. In *International Conference on Machine Learning*, pp. 8096–8108. PMLR, 2021.

OpenAI. Gpt-4 technical report, 2023.

Pinto, L. and Gupta, A. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3406–3413, 2016. doi: 10.1109/ICRA.2016.7487517.

Quine, W. V. O. *Word & Object*. MIT Press, 1960.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. Hierarchical text-conditional image generation with clip latents. *ArXiv*, abs/2204.06125, 2022.

Savva, M., Kadian, A., Maksymets, O., Zhao, Y., Wijmans, E., Jain, B., Straub, J., Liu, J., Koltun, V., Malik, J., et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9339–9347, 2019.

Shah, D., Osiński, B., Levine, S., et al. LM-Nav: Robotic navigation with large pre-trained models of language, vision, and action. In *6th Annual Conference on Robot Learning*, 2022.

Sharma, P., Torralba, A., and Andreas, J. Skill induction and planning with latent language. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1713–1726, 2022.

She, L., Cheng, Y., Chai, J. Y., Jia, Y., Yang, S., and Xi, N. Teaching robots new actions through natural language instructions. In *The 23rd IEEE International Symposium on Robot and Human Interactive Communication*, pp. 868–873. IEEE, 2014.

Shridhar, M., Thomason, J., Gordon, D., Bisk, Y., Han, W., Mottaghi, R., Zettlemoyer, L., and Fox, D. ALFRED: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10740–10749, 2020.

Shridhar, M., Manuelli, L., and Fox, D. Cliport: What and where pathways for robotic manipulation. *CoRL*, 2021.

Singh, I., Blukis, V., Mousavian, A., Goyal, A., Xu, D., Tremblay, J., Fox, D., Thomason, J., and Garg, A. Progprompt: Generating situated robot task plans using large language models. *arXiv preprint arXiv:2209.11302*, 2022.

Sumers, T. R., Ho, M. K., Hawkins, R. D., Narasimhan, K., and Griffiths, T. L. Learning rewards from linguistic feedback. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 6002–6010, 2021.

Tam, A., Rabinowitz, N. C., Lampinen, A. K., Roy, N. A., Chan, S. C., Strouse, D., Wang, J. X., Banino, A., and

Hill, F. Semantic exploration from language abstractions and pretrained representations. In *Advances in Neural Information Processing Systems*, 2022.

Tellex, S., Kollar, T., Dickerson, S., Walter, M., Banerjee, A., Teller, S., and Roy, N. Understanding natural language commands for robotic navigation and mobile manipulation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 25, pp. 1507–1514, 2011.

Tellex, S., Gopalan, N., Kress-Gazit, H., and Matuszek, C. Robots that use language. *Annual Review of Control, Robotics, and Autonomous Systems*, 3(1), 2020.

Thomason, J., Padmakumar, A., Sinapov, J., Hart, J., Stone, P., and Mooney, R. Opportunistic active learning for grounding natural language descriptions. In *Proceedings of the 1st Annual Conference on Robot Learning (CoRL-17)*, 2017.

Winograd, T. Understanding natural language. *Cognitive psychology*, 3(1):1–191, 1972.

Xiao, T., Chan, H., Sermanet, P., Wahid, A., Brohan, A., Hausman, K., Levine, S., and Tompson, J. Robotic skill acquisition via instruction augmentation with vision-language models. *arXiv preprint arXiv:2211.11736*, 2022.

Yan, C., Carnevale, F., Georgiev, P., Santoro, A., Guy, A., Muldal, A., Hung, C.-C., Abramson, J. S., Lillicrap, T. P., and Wayne, G. Intra-agent speech permits zero-shot task acquisition. In *Advances in Neural Information Processing Systems*, 2022.

Yu, H., Zhang, H., and Xu, W. Interactive grounded language acquisition and generalization in a 2D world. In *ICLR*, 2018.

Zeng, A., Wong, A., Welker, S., Choromanski, K., Tombari, F., Purohit, A., Ryoo, M., Sindhwani, V., Lee, J., Vanhoucke, V., et al. Socratic models: Composing zero-shot multimodal reasoning with language. *arXiv preprint arXiv:2204.00598*, 2022.

Zhong, V., Rocktäschel, T., and Grefenstette, E. RTFM: Generalising to new environment dynamics via reading. In *ICLR*, 2020.

Zhong, V., Hanjie, A. W., Wang, S., Narasimhan, K., and Zettlemoyer, L. SILG: The multi-domain symbolic interactive language grounding benchmark. *Advances in Neural Information Processing Systems*, 34:21505–21519, 2021.

Zhong, V., Mu, J., Zettlemoyer, L., Grefenstette, E., and Rocktäschel, T. Improving policy learning via language dynamics distillation. In *Advances in Neural Information Processing Systems*, 2022.

A. Appendix

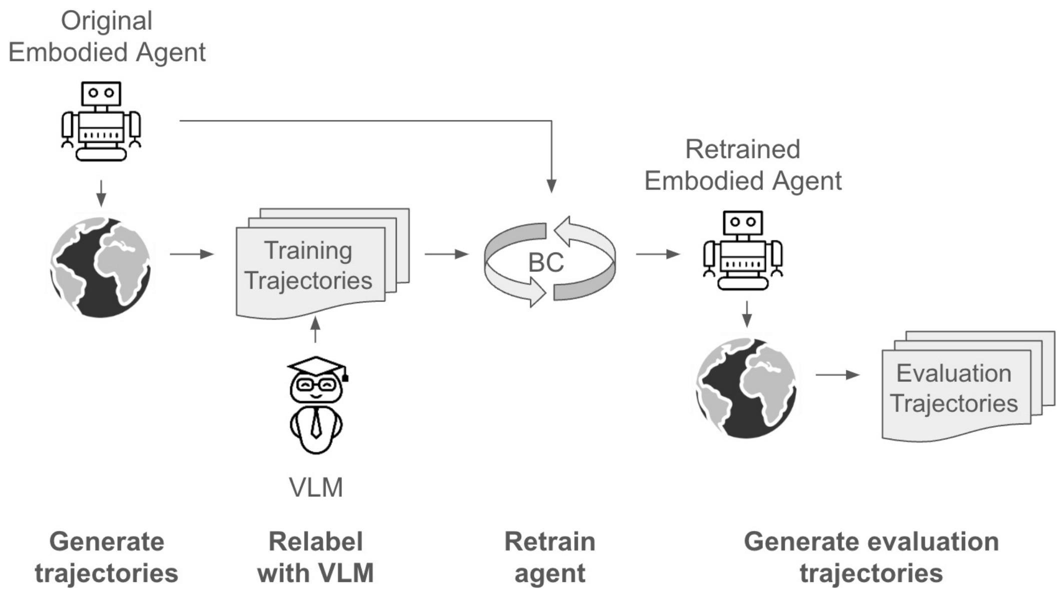


Figure S1. Schematic of our batched HER implementation (Section 4). We generate a batch of trajectories, relabel those trajectories with a VLM, and then re-train the agent with behavioral cloning. We then rollout the retrained agent on target tasks to evaluate performance.

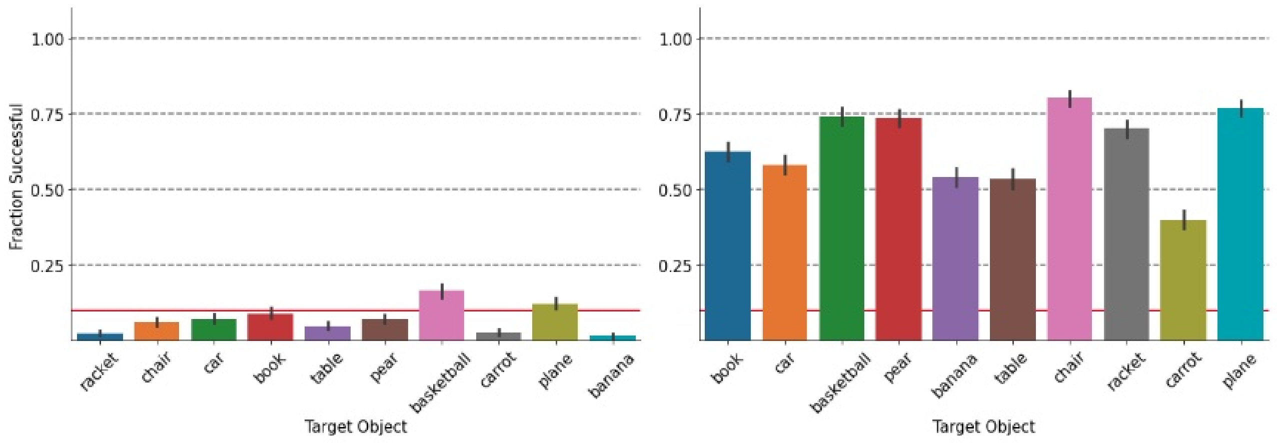


Figure S2. Original and retrained task success per object (Section 5.1).

Zeroshot Caption	Frequency
a basketball	875
a banana	525
it is a basketball	480
a cube	364
a pear	319
a chair	317
it is a banana	303
it is a chair	242
a car	221
it is a cube that is a child of the camera	209
a plane	202
it is a cube that is a child of a camera	198
a box	193
it is a car	172
a table	171
a racket	154
its a basketball	152
it is a table	139
it is a pear	122
it is a basketball hoop	119
a carrot	108
it is a plane	91
it is a cube	88
a tennis racket	83
its a banana	81

Table S1. Top 25 Zeroshot captions (lowercased and punctuation-stripped). There were 11051 total relabeled trajectories (Section 5.1).

Zeroshot Caption	Frequency
its a toy	7704
it is a toy	1022
its both	676
both	400
its a food toy	63
its a toy but its also food	30
its food	22
it is both	20
it is a toy but it is also food	9
its a toy but its also a food	8
its a food	7
its a toy but it can be used as a food	5
its a toy but it is also food	3
food	3
its a toy but its not a toy	2
its food but its also a toy	2
its a toy carrot	2
its food but its not edible	2
its a toy but its a toy that you can eat	2
its a toy but it can be used as food	2
its a food simulator	2
its a toy its a food toy	1
its a toy but it can be used as food if you wan...	1
its a toy that is also food	1
its a food that is also a toy	1
its a toy that looks like food	1
its a toy its a toy	1
it is a food	1
its food for your mind	1
its a toy but it can be used to feed your pet	1
its a toy but its not a toy you can play with	1
its a food but its not edible	1
it is a food toy	1
its a toy but it can be a food too	1
it is a toy but the food is coming soon	1
it is a toy it is not edible	1
its a toy but its not a ball	1

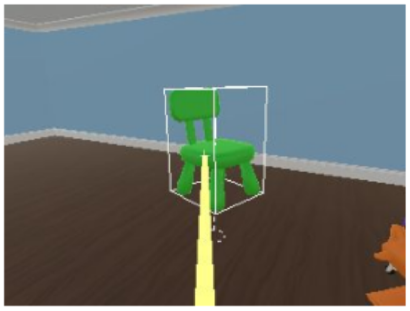
Table S2. All “food or toy” “Zeroshot” captions (lowercased and punctuation-stripped). There were 10002 total relabeled trajectories (Section 5.3).

Fewshot-1 Caption	Frequency
toy	6186
food	3816

Table S3. All “food or toy” “Fewshot-1” captions (lowercased and punctuation-stripped). There were 10002 total relabeled trajectories (Section 5.3).

	Effect	Group	Term	Estimate	Std. Error	Statistic	DOF	P Value
1	fixed		(Intercept)	-0.70	0.11	-6.43	36.50	1.74e-07 ***
2	fixed		Label Precision	1.44	0.13	11.50	35.76	1.43e-13 ***
3	fixed		Label Accuracy	0.16	0.05	3.42	35.88	0.00159 **
4	ran_pars	Task	sd__(Intercept)	0.05				
5	ran_pars	Residual	sd__Observation	0.05				

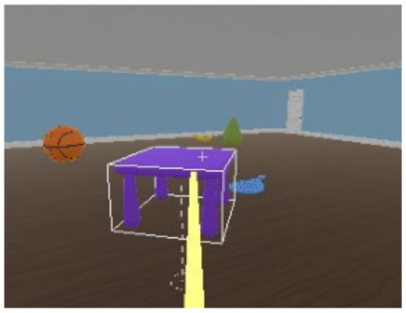
Table S4. Regressing task performance on label accuracy and precision shows that the effects of label precision far outweigh those of accuracy (Section 5.5).



Q: What is this object? A:

ZS: **it is a cube that is used to measure the size of an object.**

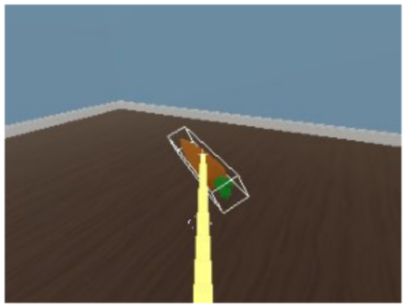
FS: **chair**



Q: What is this object? A:

ZS: **this is a cube with a custom shader that is a simple box with a texture on it.**

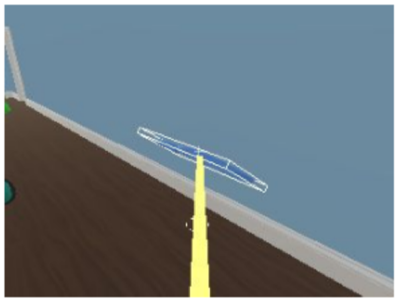
FS: **basketball**



Q: What is this object? A:

ZS: **it is a laser pointer**

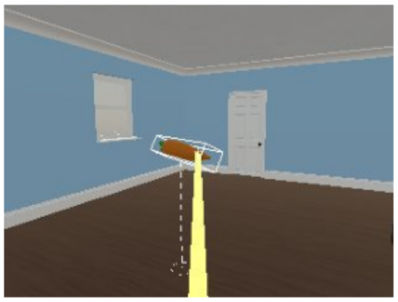
FS: **carrot**



Q: What is this object? A:

ZS: **It is a laser sword.**

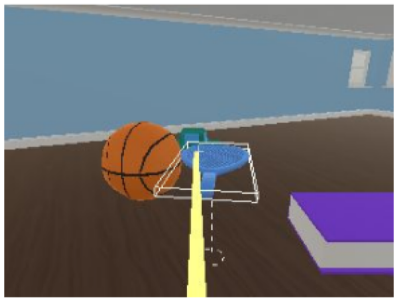
FS: **racket**



Q: What is this object? A:

ZS: **It is a bottle of beer.**

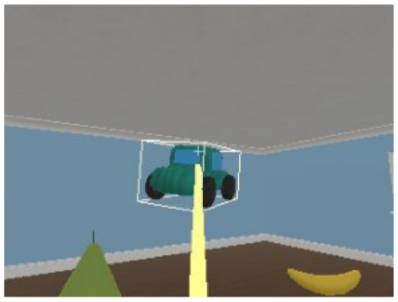
FS: **carrot**



Q: What is this object? A:

ZS: **it is a basketball hoop.**

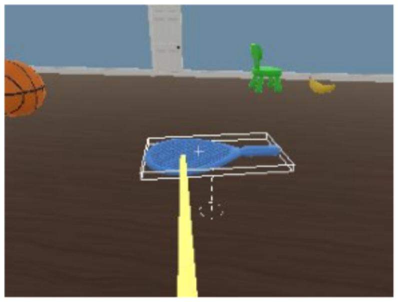
FS: **basketball**



Q: What is this object? A:

ZS: **a tractor**

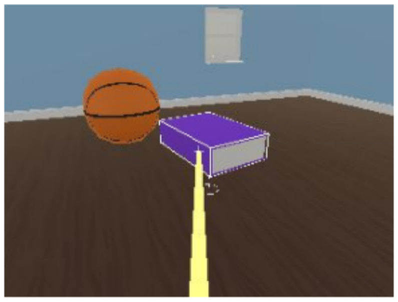
FS: **car**



Q: What is this object? A:

ZS: **a tennis racket**

FS: **racket**

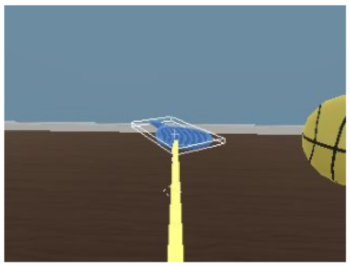


Q: What is this object? A:

ZS: **A box.**

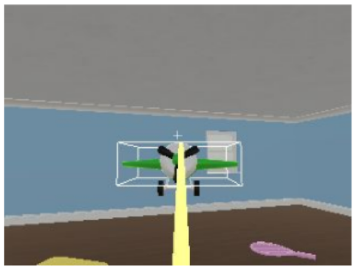
FS: **basketball**

Figure S3. Additional "Zeroshot" (ZS, Section 5.1) and "Fewshot" (FS, Section 5.5) relabeling examples. Regular text is the prompt and bold text is the VLM generation. "Correct" relabelings are green, "Wrong" relabelings are red, and "Irrelevant" text is gray. "Zeroshot" relabeling generally results in reasonable text, but is distracted by unusual visual features (such as the cube that appears around objects, top left; or the yellow ray indicating the agent is grasping something; top right).



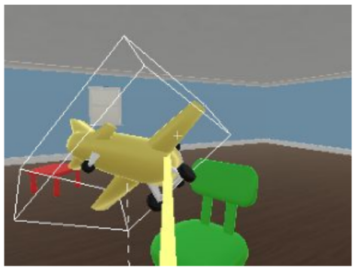
ZS-Shape: It is a 3D model of a **basketball** and a net.

ZS-Color: it's **blue**



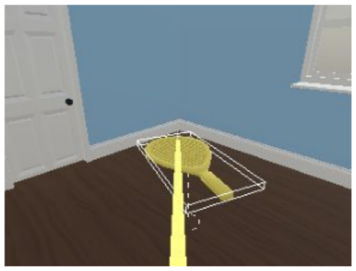
ZS-Shape: it is a **plane** that you can fly in the game.

ZS-Color: **green**



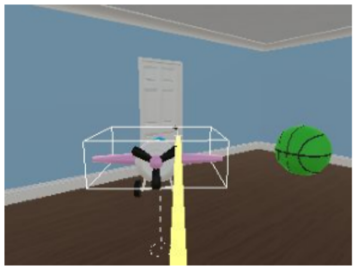
ZS-Shape: it is a **plane** with a propeller.

ZS-Color: the object is **yellow.**



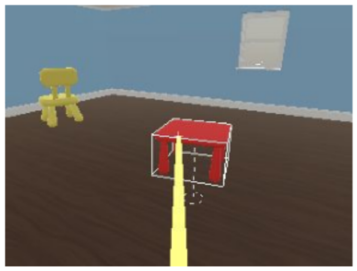
ZS-Shape: a **racket**

ZS-Color: I don't know.



ZS-Shape: a cube

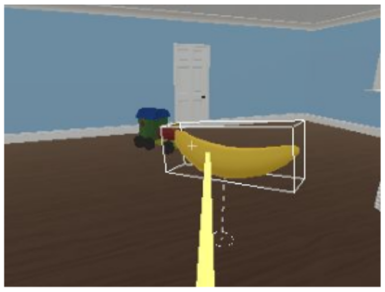
ZS-Color: The color of the object is the color of the light that is reflecting off of it.



ZS-Shape: it is a **chair.**

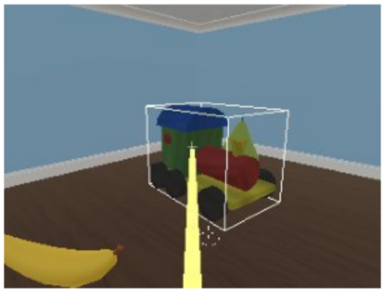
ZS-Color: **red**

Figure S4. Additional examples for Section 5.2. VLM relabelings using zeroshot "names" (top) and "color" (bottom) prompts. Regular text is part of the prompt and bold text is the VLM generation. "Correct" relabelings are green, "Wrong" relabelings are red, and "Irrelevant" text is gray. We conduct parallel VLM-based relabelings of the same initial trajectories in order to teach the agent to recognize *objects* or *colors* respectively.



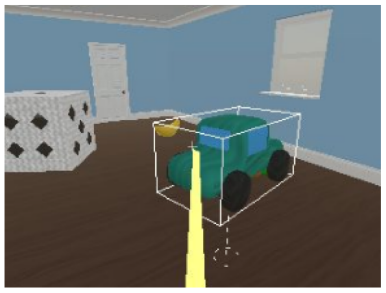
Q: is this food or a toy? A:

ZS: it's a **toy.**
FS-1: **toy**
FS-2: **food**
FS-3: **food**
FS-4: **food**
FS-5: **food**



Q: is this food or a toy? A:

ZS: it's a **toy.**
FS-1: **toy**
FS-2: **toy**
FS-3: **toy**
FS-4: **toy**
FS-5: **toy**



Q: is this food or a toy? A:

ZS: it's a **toy.**
FS-1: **toy**
FS-2: **toy**
FS-3: **toy**
FS-4: **toy**
FS-5: **toy**

Figure S5. Additional examples for Section 5.3. The basic prompt template was the same for all; few-shot Flamingo provided in-context examples of food or toy items. Ironically, zero-shot Flamingo claimed that most food items were either "toys" or some variant of "both", likely due to their 3D rendered appearances (Table S2). Providing in-context examples readily overcame this issue (Table S3).

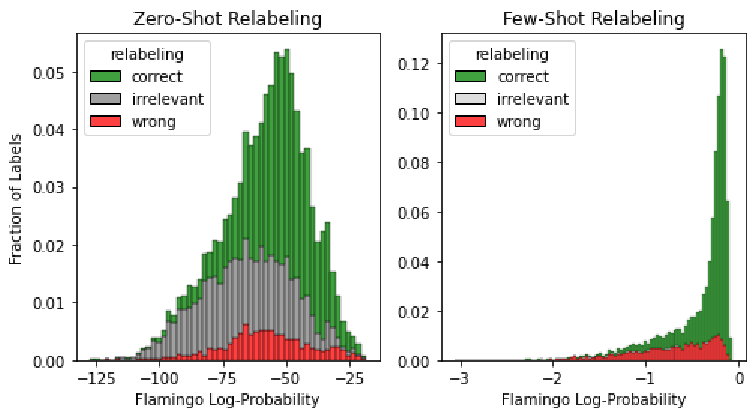


Figure S6. Stacked histograms showing label quality as a function of Flamingo's confidence. For both, wrong or irrelevant labels tend to be low confidence. However, "Fewshot" is better calibrated: virtually all correct labels are high-confidence.

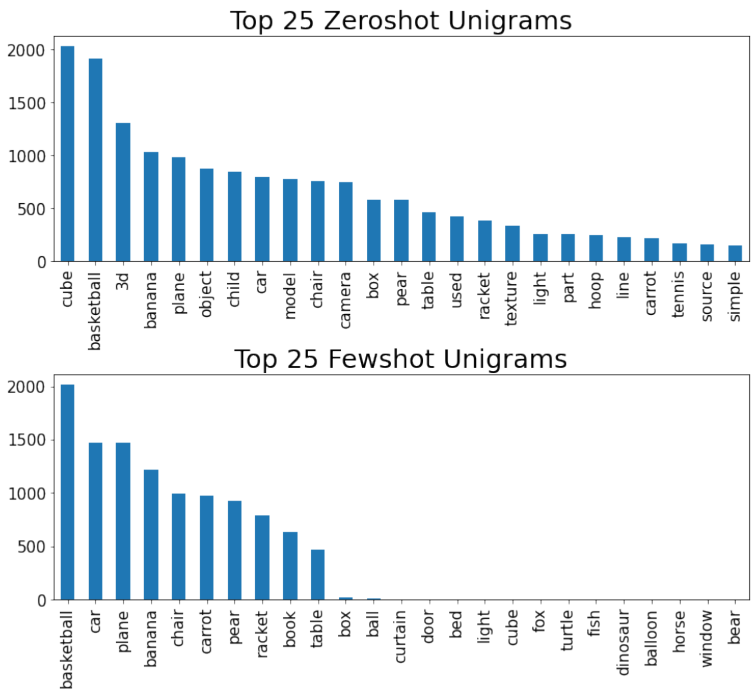


Figure S7. Unigram frequency (lowercased and stopword-filtered) for "Zeroshot" (Sec. 5.1) and "Fewshot" (Sec. 5.5) relabeling. There were 11051 relabeled trajectories.

B. OWL-ViT Comparisons

We compared OWL-ViT (Minderer et al., 2022) based relabeling to Flamingo in our Experiment 1 (Section 5.1). We provided OWL-ViT with the list of 10 objects used in the experiment (plane, basketball, chair, table. . .) and took the highest-confidence detection. We found that OWL-ViT performed poorly: on a subset of 300 trajectories, it predicted the held object only 7.2% of the time, compared to Flamingo's zero-shot 54.8% performance. OWL-ViT only ever predicted a limited subset of objects and achieved extremely low precision on these (Table S5). Flamingo had higher accuracy than OWL-ViT across 9/10 objects, often by wide margins (Table S6).

OWL-ViT Label	Count	Precision
basketball	39	0.10
book	69	0.12
chair	44	0.07
table	140	0.04

Table S5. OWL-ViT consistently predicted a small subset of the 10 object labels, achieving very low precision.

Object	Count	OWL-ViT Accuracy	Flamingo Accuracy
banana	34	0.00	0.76
basketball	47	0.09	1.00
book	27	0.30	0.11
car	23	0.00	0.65
carrot	28	0.00	0.14
chair	24	0.12	0.58
pear	14	0.00	0.64
plane	43	0.00	0.58
racket	34	0.00	0.24
table	18	0.33	0.50

Table S6. Flamingo achieved substantially higher accuracy than OWL-ViT on 9/10 objects.

Our OWL-ViT implementation performed as expected on real-world images, so we hypotheize its poor performance here is due to domain shift from real-world to Unity-generated environments. Notably, Flamingo often produced captions which reflected this domain shift (e.g. "a 3d model of a car", Fig. 2). It may thus be possible to improve OWL-ViT's performance by changing the target labels to reflect the objects' idiosyncratic visual appearances. However, this necessity reflects the brittleness of the object-detection approach, compared to zero-shot generative captioning.

C. Testing Preference Generalization

We investigated whether the system is able to generalize to other instances of preferred objects (Section 5.4). Specifically, we kept the few-shot prompt the same, but varied the color of the objects in the environment by permuting the toys' colors. We then examined the VLM's ability to re-label them. We found that across all 5 toys, for both "aligned" and "arbitrary" preference structures, the VLM readily generalized to the new colors (Table S7). The VLM's accuracy is consistent across objects (Table S8).

Preference Structure	Coloring	Accuracy
Aligned	Canonical	0.91
Aligned	Recolored	0.92
Arbitrary	Canonical	0.71
Arbitrary	Recolored	0.70

Table S7. The VLM readily generalized to relabeling recolored objects with ad-hoc category structures.

preferences	unity_object_name	color	correct
Aligned	Car	Canonical - aquamarine	0.91
Aligned	Car	Recolored - red	0.92
Aligned	Dice	Canonical - white	0.97
Aligned	Dice	Recolored - purple	0.98
Aligned	Plane	Canonical - orange	0.85
Aligned	Plane	Recolored - aquamarine	0.87
Aligned	Robot	Canonical - purple	0.96
Aligned	Robot	Recolored - orange	0.92
Aligned	Train	Canonical - none	0.87
Aligned	Train	Recolored - white	0.92
Arbitrary	Car	Canonical - aquamarine	0.46
Arbitrary	Car	Recolored - red	0.47
Arbitrary	Dice	Canonical - white	0.94
Arbitrary	Dice	Recolored - purple	0.96
Arbitrary	Plane	Canonical - orange	0.85
Arbitrary	Plane	Recolored - aquamarine	0.80
Arbitrary	Robot	Canonical - purple	0.70
Arbitrary	Robot	Recolored - orange	0.64
Arbitrary	Train	Canonical - none	0.59
Arbitrary	Train	Recolored - white	0.55

Table S8. The VLM readily generalized across all objects, achieving comparable accuracy on original and recolored versions.