WHAT IF? COUNTERFACTUAL SIMULATION FOR SAMPLE-EFFICIENT VISUAL REWARD LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Pretrained visual robot reward models capture generic notions of task success, not necessarily a person's specific preferences for how their task should be done. The robot can learn these preferences from user demonstrations, but with only a few demonstrations, a reward model may latch onto spurious correlations without learning what the person actually wants. To better express their preferences, a person could directly specify a *preference axis*—or an aspect of the task that they care about—in language. But a language-conditioned reward model may still reward the wrong behavior when training demonstrations don't fully cover the entire preference axis. Our key idea is to use the person's instruction to cover this axis by imagining what they *don't* want. We introduce COVER, which uses a video world model to generate *counterfactual trajectories*: alternatives that are not preferred along the preference axis but keep the rest of the behavior and scene the same. Comparing the original demonstration against its counterfactual isolates the quality that should make the reward different, without asking the person to demonstrate both versions. We train the reward to score each demonstration higher than its corresponding counterfactual, alongside learning from the demonstrations directly. Across four simulated and two real-robot tasks, COVER learns rewards that capture user preferences better than language-conditioned baselines under the same demonstration budgets, matching the strongest baseline's accuracy with up to $10\times$ fewer demonstrations. On a real robot, selecting training demonstrations with these rewards yields policies that follow both stated preferences of each task more closely than training on unfiltered demonstrations.

1 INTRODUCTION

Pretrained visual reward models can generalize across scenes, objects, and task instructions (Liang et al., 2026; Lee et al., 2026), but they typically reflect generic criteria learned from broad data, like estimates of task progress or success (Zhang et al., 2025; Ma et al., 2025; Du et al., 2023). People, meanwhile, have specific *preferences* for how their tasks should be done (Peng et al., 2024a; Poddar et al., 2024). In the example in Fig. 1, a pretrained reward may recognize progress as the robot carries the kettle towards the stove; the user, however, may care about *how* it gets there: whether the robot carries it high or low, keeps it upright or lets it tilt, and passes near or far from a pan of eggs. The robot can learn these preferences by fine-tuning its reward on user demonstrations, but this often requires hundreds of demonstrations (Zakka et al., 2022; Zhao et al., 2023; Chi et al., 2024).

Unfortunately, when the robot only has a few demonstrations to learn from, many reward functions can fit the data equally well (Ng et al., 2000). In our kettle example, the user may carry the kettle high while also keeping it upright and close to the pan. The robot could then learn to reward uprightness or proximity, even if the user actually only cares about height. In image space, these accidental associations are further exacerbated by visual attributes, like background appearance or occlusions. The robot may therefore latch onto spurious correlations in the demonstrations without learning the intended preference.

Natural language lets the user directly specify a *preference axis*—a feature of the behavior that they care about, e.g., carry height in "Carry the kettle high!". Knowing this axis should help the robot distinguish the intended reward from other explanations of the demonstrations. But when the demonstrations never show that axis varying independently of the others, a reward model condi-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

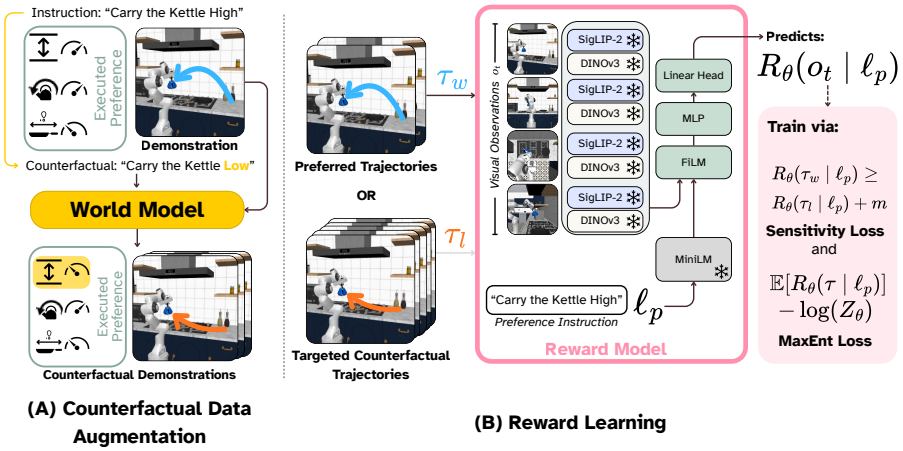


Figure 1: **COVER overview.** (a) A fine-tuned video world model generates counterfactual trajectories from the demonstration's first frame, conditioned on a caption about the desired change along a preference axis (here, carry *low*). (b) The reward model: each camera's frames are encoded by frozen SigLIP-2 and DINOv3 and concatenated per frame. Visual embedding and MiniLM embedding $\mathbf{e}_\ell$ of the preference instruction ℓ_p pass into a FiLM layer, 2-layer MLP, and linear head to calculate a scalar reward at time t. We use the sensitivity loss to order each winner above its loser by margin m. We add sensitivity loss to MaxEnt IRL loss for reward learning.

tioned on the instruction may still latch onto the wrong axis. We can use language for more informed supervision by specifying how the reward should change across visual observations.

Our key idea is to use the person's instruction to imagine what they *don't* want. Rather than ask them to demonstrate it (which is expensive) or execute it on the robot (which is potentially dangerous), we generate a dispreferred behavior along that axis while keeping everything else the same. In the kettle example, the robot imagines a rollout with a low carry but the same uprightness, pan clearance, and visual scene. Because the imagined alternative matches the demonstration on everything other than carry height, comparing the two trajectories singles out height as the reason for a different reward. We call these imagined alternatives *counterfactual trajectories*, taking inspiration from counterfactual simulation in cognitive science and policy learning (Peng et al., 2024a; 2023), where people judge causes by imagining a scene with a particular aspect changed (Gerstenberg et al., 2021).

We introduce COVER (**CO**unterfactual **V**isual r**E**ward lea**R**ning), a framework that learns a language-conditioned visual reward model from demonstrations and generated counterfactual trajectories. We fine-tune a video world model on unlabeled video trajectories with captions from a vision-language model (VLM). By conditioning on a demonstration's initial observation and a description that differs only along the user's stated preference axis, the world model generates the counterfactual. We then train the reward to score each demonstration above its counterfactual, alongside learning from the demonstrations directly.

We evaluate COVER on four simulated and two real robot tasks, and find substantial improvements over baselines that share the same reward architecture, instruction conditioning, and demonstration budget but trained without targeted counterfactual negatives. With $N{=}10$ demonstrations per preference, COVER ranks first on all four simulated tasks, and it doubles the accuracy over the standard MaxEnt IRL (Finn et al., 2016) baseline on CLOSEFRIDGE. COVER is also data-efficient: on COFFEESERVEMUG, with only 2 demonstrations per preference it already matches the accuracy the strongest baseline reaches with 20, a $10\times$ reduction in demonstrations. Additionally, we find that generating counterfactuals targeted to violate the stated user preference outperforms using random background trajectories or another preferences' demonstrations as counterfactual negatives. Finally, on a real 7DoF Franka robot, we use the learned rewards to filter trajectories for personalized policy training and show that only our method improves the ground-truth preference return over uniform policy trained on all trajectories without filtering.

Our contributions are: 1) a language-conditioned visual reward learning framework that learns personalized rewards from a few demonstrations; 2) a world-model technique generating counterfactual trajectories without human supervision; and 3) real-world policy improvement with these rewards.

2 Related Work

Image-based reward models. A growing line of work derives scalar reward signals from image observations, including task- and goal-conditioned visual rewards obtained from demonstrations or action-free videos (Ma et al., 2023a;b; Sontakke et al., 2023), language-grounded rewards and progress estimators based on vision-language models (VLMs) (Mahmoudieh et al., 2022; Du et al., 2023; Rocamonde et al., 2024; Ma et al., 2025), and large-scale general-purpose reward models trained across diverse robot tasks and embodiments (Liang et al., 2026; Lee et al., 2026). Many of these approaches generalize across tasks, scenes, and objects, but use language mainly to specify the task and lack explicit constraints for identifying user-specific preferences from a few demonstrations.

Language-grounded abstractions for data-efficient reward learning. Beyond selecting a task, language can specify which aspects of behavior matter, and prior work exploits this by constructing preference-relevant abstractions. Existing methods mask named state features (Hwang et al., 2026b; Abi Nader et al., 2026), build task- or preference-relevant visual masks using foundation segmentation models (Kirillov et al., 2023; Carion et al., 2026; Peng et al., 2024a;b), or learn preference-relevant representations from human feedback, such as preference-aligned visual representations (Tian et al., 2024) and language-aligned rationale embeddings (Hwang et al., 2026a); see Bobu et al. (2024) for a broader survey. These methods commit to a fixed level of abstraction for reward learning, which can limit the class of preferences that can be captured by the reward function. We instead use language to generate counterfactual training pairs, allowing the reward to learn from full observations.

Contrastive learning in robotics. A contrastive objective is defined by the relation its pairs encode (Oord et al., 2018). In robot learning, these relations are typically symmetric: temporally close frames (Sermanet et al., 2018; Nair et al., 2023), augmented views (Laskin et al., 2020), and trajectories humans judge similar (Bobu et al., 2023; Dennler et al., 2025; Tian et al., 2024) are pulled together. A preference, however, is an ordering, not an equivalence. Preference-based reward learning therefore ranks pairs of trajectories (Bradley & Terry, 1952; Christiano et al., 2017; Lee et al., 2021; Hejna et al., 2024) or sets of demonstrations (Brown et al., 2019), but human pairwise supervision requires a large number of comparisons, and randomly sampled pairs may differ along many factors, allowing rewards to exploit spurious cues. Noise injection (Brown et al., 2020) removes the annotation cost, but induces generic degradation rather than variation along a stated preference. COVER's sensitivity pairs instead provide automatic, controlled ordering supervision: counterfactual generation changes only a language-specified preference axis.

World models for robot learning. Prior work commonly uses learned dynamics and video models either as predictive simulators for planning or training (Hou et al., 2026; Ha & Schmidhuber, 2018; Hafner et al., 2023; Bruce et al., 2024; Parker-Holder et al., 2025; Assran et al., 2025), or as scalable video generators trained on robotics-heavy corpora (NVIDIA et al., 2025) or large-scale video dataset (Yang et al., 2025; Kong et al., 2024; OpenAI, 2025; Google DeepMind, 2025). Recent work (Wu et al., 2023; Liu et al., 2024) edit source clips under instructions while preserving their content. We use a video world model to vary a language-specified preference axis while preserving other behavior and scene factors, thereby generating ordering supervision for reward learning.

3 Method

Our goal is to efficiently learn robot policies that satisfy individual users' preferences when performing a given task. Existing preference-learning methods face two fundamental challenges. First, they require substantial user data to learn what the user cares about: demonstrations mainly show desirable behavior, and comparison-based methods need hundreds of labeled pairs. Second, this data must be produced on the physical robot; in particular, eliciting what the user does *not* want would require demonstrating or executing undesirable behavior in the real world. Our algorithm, COVER, addresses these challenges by using a world model to generate counterfactual trajectories. These generated counterfactual trajectories augment limited behavioral data and enable the robot to elicit information about undesirable behaviors without requiring execution in the physical environment.

We first review maximum-entropy IRL to explicate why demonstrations alone leave a preference underspecified (§3.1). We then introduce the *sensitivity loss* that induces ordering constraints on the reward based on generated counterfactuals (§3.2). Finally, we describe how the counterfactuals are produced by a fine-tuned video world model (§3.3).

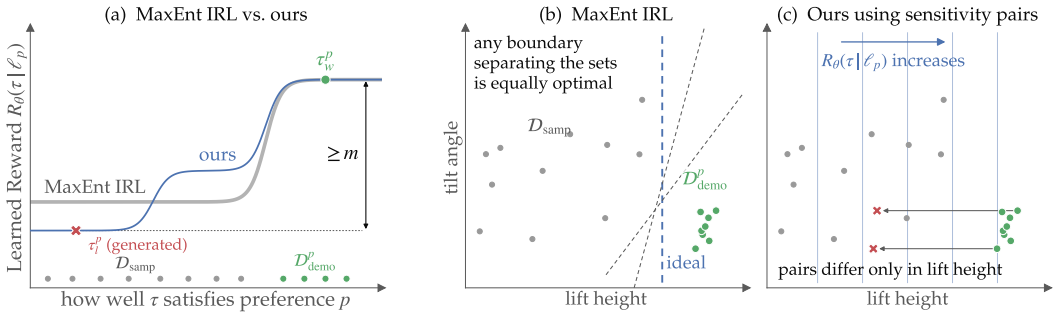


Figure 2: **The identifiability gap** (schematic). (a) MaxEnt IRL only requires demonstrations $\mathcal{D}^p_{\text{demo}}$ to score above background trajectories $\mathcal{D}_{\text{samp}}$, so a reward that is constantly low across everything undemonstrated (gray) is optimal. A sensitivity pair (τ_w^p, τ_ℓ^p) enforces a margin m (Eq. 2), selecting a reward that increases with the preference (blue). (b) The same gap in the plane of a relevant feature (lift height) and an irrelevant one (tilt): any separating boundary is equally optimal. (c) Generated losers τ_ℓ^p differ from winners only in lift height, pinning the reward's increase to the preference axis.

3.1 Preliminaries: Maximum-Entropy IRL and the Identifiability Gap

We denote a trajectory $\tau = (o_1, o_2, \ldots, o_T)$ as sequence of T observations, where each observation consists of multi-camera images (e.g., third-person, side, and wrist views) of the robot. We assume that a person implicitly evaluates the utility of a trajectory based on an internal reward function, $R(\tau)$. We approximate this reward function from data using a neural network parameterized by θ, and we denote this model as $R_\theta(\tau)$. The problem of learning a user's preference can be formulated as finding the parameters of $R_\theta(\tau)$ that maximize the likelihood of the user's observable behavior, such as the demonstrations they provide. Real-world preferences often depend on multiple *axes* of comparisons, i.e., different aspects of how the task is performed. For example, a user may care about "speed" or "height" of a trajectory. We refer to a general preference axis as p.

Suppose a user can provide a small dataset of expert demonstrations $\mathcal{D}^p_{\text{demo}} = \{\tau_i\}$, where all demonstrations correspond to an instruction, ℓ_p, that describes the preference axis (*e.g.* "carry the kettle high"). The objective of reward learning is to learn a reward R_θ that inputs (τ, ℓ_p) and outputs the cumulative reward over the image sequence.

Maximum-entropy IRL (Ziebart et al., 2008) aims to learn such a reward from demonstrations by modeling the demonstrator as noisily sampling from a Gibbs distribution, $q_\theta(\tau) \propto \exp R_\theta(\tau)$. Using this model, we fit R_θ by maximizing the likelihood of the observed demonstrations:

$$\max_\theta \; \mathbb{E}_{\tau \sim \mathcal{D}^p_{\text{demo}}}[R_\theta(\tau \mid \ell_p)] - \log Z_\theta, \qquad Z_\theta = \int \exp R_\theta(\tau \mid \ell_p)\, d\tau. \tag{1}$$

Following guided cost learning (Finn et al., 2016), we model R_θ with a neural network and estimate the intractable $\log Z_\theta$ from *background trajectories* $\mathcal{D}_{\text{samp}} = \{\tau_j\}$. We use unlabeled trajectories that do not correspond to specific preferences as background trajectories.

Under each preference axis instruction ℓ_p, Eq. (1) only requires $\mathcal{D}^p_{\text{demo}}$ to score above $\mathcal{D}_{\text{samp}}$. With few demonstrations in high-dimensional image space, many incidental features can discriminate between the two sets (*e.g.*, object placements, background details, joint poses), so the learned reward may not capture the property that ℓ_p describes. Furthermore, all non-demonstrated trajectories are treated the same way by Eq. (1), regardless of how close they are to satisfying the user's preference; a trajectory that is too slow is identical to a trajectory that is both too slow and too low.

Just asking for additional preferred demonstrations does not resolve this issue. Without observing trajectories that are *dispreferred* by the user, a learned reward function cannot determine what aspects of a trajectory cause that trajectory to score lower than a preferred one. To learn what features of the trajectory are meaningful for user preferences, we posit that reward models should learn from *controlled* comparisons. We aim to generate counterfactual trajectories that violate the user's preference along a particular preference axis, ℓ_p, while matching a demonstration in $\mathcal{D}^p_{\text{demo}}$ in everything else. Using targeted counterfactual trajectories constrains the learned reward along the preference axis that is undetermined when only positive examples are observed (see Fig. 2).

3.2 SENSITIVITY LOSS: COUNTERFACTUAL NEGATIVES AS ORDERING CONSTRAINTS

We formulate counterfactual generation (§3.3) as a method that produces pairs of trajectories (τ_w^p, τ_l^p) that only differ along the preference axis p, holding all other trajectory features same. These counterfactual pairs are the required data structure used in our *sensitivity loss*.

Sensitivity loss. We call the pair (τ_w^p, τ_l^p) a *sensitivity pair*, with winner τ_w^p and loser τ_l^p. This pair encourages a reward model to discriminate between trajectories that differ along a particular preference axis, p, and thus results in a learned reward that is *sensitive* to the given preference axis. We refer to the dataset of all sensitivity pairs as $\mathcal{P}_{\text{sens}}^p$. The learning objective is a hinge loss with margin m on the predicted-reward difference,

$$\mathcal{L}_{\text{sens}} = \mathbb{E}_{(\tau_w^p, \tau_\ell^p) \sim \mathcal{P}_{\text{sens}}^{(p)}} \big[\max\big(0,\ m - (R_\theta(\tau_w^p \mid \ell_p) - R_\theta(\tau_\ell^p \mid \ell_p))\big)\big]. \tag{2}$$

Each pair provides an example that varies along the preference axis that positive demonstrations can leave undetermined (§3.1, Fig. 2). The total objective is

$$\mathcal{L} = \mathcal{L}_{\text{IRL}} + \lambda_{\text{sens}} \cdot \mathcal{L}_{\text{sens}}, \tag{3}$$

where λ_{sens} trades off the sensitivity objective against the maximizing the demonstration likelihood, and m sets how far apart the pair must score. Theoretically, adding sensitivity loss to MaxEnt IRL loss can be interpreted as solving a constrained maximum-entropy problem: Eq. (2) is the penalty form of adding pairwise ordering constraints to Eq. (1) (see App. B for detailed analysis). We use hinge loss because it enforces the ordering and nothing more: a pair already separated by m contributes no gradient, so a loser is pushed below the corresponding winner but not toward $-\infty$. Swapping the hinge for the Bradley–Terry logistic loss changes results only marginally (App. E.2).

3.3 COUNTERFACTUAL GENERATION WITH A VIDEO WORLD MODEL

In principle, the robot could collect sensitivity pairs directly, by asking the user to demonstrate each counterfactual trajectory that violates the preference. This would require users to provide twice as much data, and executing counterfactual behavior on a robot is potentially unsafe. We instead generate the pairs automatically with a world model to both reduce effort and encourage safety.

We generate sensitivity pairs using a video world model g_ϕ. The world model is conditioned on the initial frame o_1 of a demonstration and a *structured caption* c that describes for each preference axis, which of three levels (low, moderate, or high) the behavior exhibits (e.g., "the kettle is carried *low* above the stovetop, held *roughly upright*, passing at a *moderate* distance from the fry pan"), and outputs a trajectory video $v = g_\phi(o_1, c)$. Unlike the user's instruction ℓ_p, which names only the one axis they care about, c specifies every axis; this is what lets the generator change one axis while holding the others fixed. Training the world model has two steps: (1) generating the caption c from background trajectory videos using VLM-based rankings, and (2) fine-tuning a pre-trained video world model on the labeled videos.

Generating captions. Absolute judgments of geometry provided by VLMs are often uncalibrated, but relative judgments between examples are more reliable (Wang et al., 2024). We thus generate captions for each video using pairwise video comparisons along preference axes. For each preference axis, the VLM answers K pairwise comparisons on trajectories. To select distinct trajectories, we encode them using the same vision encoders used for reward learning to generate a latent embedding, and sample trajectories that are far apart in this latent space. The binary VLM judgements are aggregated by fitting a Bradley–Terry model, $P(\tau_i \succ \tau_j) = \sigma(s_i - s_j)$, where s_i is the fitted score of trajectory τ_i. From the fitted scores, we split each axis into terciles (cut at the 33rd/66th percentiles), labeled low/moderate/high. Each trajectory's per-axis buckets determine its caption c_i by filling the corresponding axis phrases of the structured caption, yielding the world-model fine-tuning dataset $\mathcal{D}_{\text{WM}} = \{(\tau_i, c_i)\}$ (e.g., on the tilt axis the low/moderate/high buckets correspond with "held perfectly upright", "held roughly upright with a slight tilt", "tilted steeply").

Fine-tuning. We fine-tune a pre-trained Cosmos-Predict-2.5-2B (NVIDIA et al., 2025) on $\mathcal{D}_{\text{WM}}$ using a rectified-flow velocity loss (Liu et al., 2023). The fine-tuning corpus is all of $\mathcal{D}_{\text{WM}}$, i.e., every captioned background trajectory of $\mathcal{D}_{\text{samp}}$, including clips that happen to satisfy the announced preference: the generator must observe both extremes of every axis to be steerable toward either bucket at generation time. No preference label or user instruction enters this stage; every clip arrives with its machine-written caption only, so including preference-satisfying clips leaks no supervision about which axis the user cares about.

Sensitivity pairs. To construct the loser for a demo $\tau_w^p \in \mathcal{D}_{\text{demo}}^p$, we condition the world model on the demo's initial frame o_1 and a caption whose phrase for the target axis p is set to the bucket opposing the preference, while every other axis $q \neq p$ is set to the moderate bucket. The generated video $\tau_\ell^p = g_\phi(o_1, c)$ is dispreferred along p by construction, so (τ_w^p, τ_ℓ^p) joins $\mathcal{P}_{\text{sens}}^p$ in Eq. (2).

4 EXPERIMENTS

We address three research questions. **RQ1**: Do counterfactual trajectories improve reward learning over using demonstrations alone? **RQ2**: Does our approach for generating targeted counterfactuals outperform other methods for selecting the sensitivity pairs? **RQ3**: Do rewards learned with counterfactual trajectories improve real-world policy learning?

4.1 EXPERIMENTAL SETUP

Simulation Tasks and Preferences. We evaluate on four kitchen manipulation tasks in RoboCasa (Nasiriany et al., 2026): KETTLEBOILING, OPENCABINET, CLOSEFRIDGE, and COFFEESERVEMUG, each with three preference axes, as shown in Fig. 3. The twelve axes span motion style (e.g., carrying the kettle upright), spatial clearance and placement (e.g., keeping the kettle far from the pan, placing the mug near the counter edge), and temporal ordering (e.g., which door opens or closes first); per-axis definitions are in Table 3 (App. F.1).

Real Robot Experiment. We additionally evaluate two manipulation tasks on a physical Franka Research 3 robot: MUG TRANSPORT, carrying a mug across a table past a laptop, and WINE POUR, pouring a bottle of wine into a wineglass; see Fig. 7 and Table 3. Real-robot evaluation extends beyond reward learning to policy learning via top-M *reward-filtered behavior cloning*.

Reward model architecture. The reward $R_\theta(\tau \mid \ell_p)$ encodes each camera's frames with frozen SigLIP 2 (Tschannen et al., 2025) and DINOv3 (Siméoni et al., 2026), conditions on frozen MiniLM (Wang et al., 2020) embedding of ℓ_p via FiLM (Perez et al., 2018), and averages the per-frame outputs of a small trainable MLP head into a scalar trajectory reward (details in App. F.5).

Counterfactual generation. For each task we fine-tune Cosmos-Predict-2.5-2B (NVIDIA et al., 2025) on $\mathcal{D}_{\text{WM}}$ and generate losers from initial frames of each demonstration. We fine-tune the world model for 1k steps on 8×H100 GPUs. Implementation details along with quantitative and qualitative analyses of the generated counterfactuals are in App. D (Table 2, Fig. 9).

Baselines. We share the same reward architecture, optimizer, schedule, and instructions across all baselines. The most naive baseline is **SFT** that assigns 1 as reward labels for expert demonstrations and is learned with softplus margin loss. **MaxEnt IRL** maximizes the likelihood of demonstrations using only the MaxEnt IRL loss. **Pixel-Mask + Raw RGB** tests visual abstraction instead of generating counterfactuals, masking preference-relevant pixels by segmenting the manipulated object and the objects referenced by the preference instruction (Peng et al., 2024a). We use SAM-3 (Carion et al., 2026) to obtain mask and concatenate that with raw RGB input. **Masking loss** adapts Masked IRL (Hwang et al., 2026b) to the visual domain: using the same world model as ours, it generates trajectories that agree with a demonstration on the target preference axis but differ elsewhere, and replaces the sensitivity loss with an invariance loss that pulls their rewards together (App. C). In contrast, **Sensitivity loss (ours)** generates counterfactual negatives that violate the preference, and trains the reward model using the sensitivity loss, as described in Sec. 3.2. Beyond these controlled baselines, App. E evaluates additional supervision sources: a pretrained foundation reward model (Lee et al., 2026), both zero-shot and fine-tuned, VLM pairwise comparisons over background trajectories, and the hinge vs. Bradley–Terry form of the sensitivity loss.

Evaluation metric. For all tasks, we report *pairwise ranking accuracy*, the ratio of evaluation trajectory pairs correctly ordered along each preference axis using the learned reward. For real robot experiments, we also measure the *average return* of policy rollouts. Details are in App. F.7.

4.2 RQ1: DO COUNTERFACTUAL NEGATIVES IMPROVE REWARD LEARNING?

This section tests whether the difference between a preferred trajectory and a dispreferred counterfactual trajectory learns the preference axis a user cares about. Fig. 4 compares our method against baselines on four RoboCasa tasks. Given $N{=}10$ demonstrations per preference, with the sensitiv-

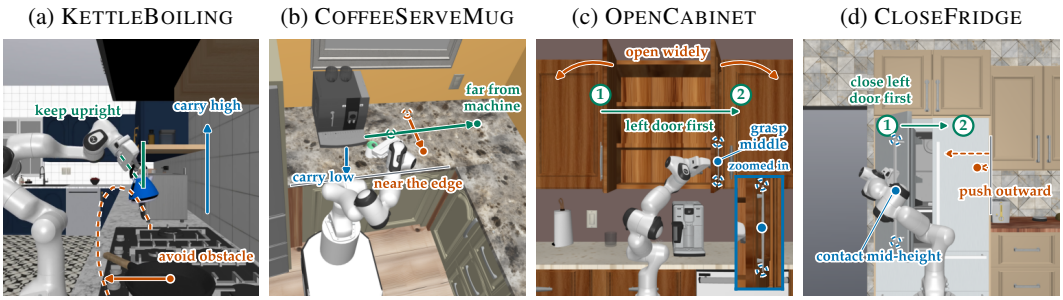


Figure 3: **Simulated tasks and preferences in RoboCasa.** (a) carrying the kettle from counter to stove; (b) picking up a mug from the coffee machine and placing it on the countertop; (c) opening both cabinet doors; (d) closing both fridge doors. Solid and dashed markers indicate preferred and less preferred behaviors, respectively.

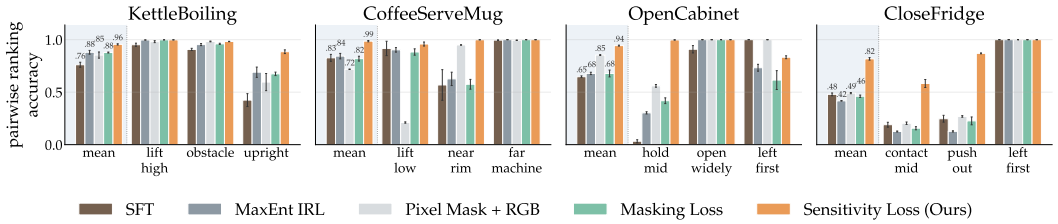


Figure 4: **Reward Performance on 4 RoboCasa Tasks.** Each task trains a single preference-language-conditioned reward model. Bars show the average pairwise ranking accuracy across all preferences, and the per-preference accuracies. Error bars show ± 1 standard error over 5 seeds.

ity loss on counterfactual pairs, the average pairwise ranking accuracy reaches 0.955 on KETTLEBOILING, 0.986 on COFFEESERVEMUG, 0.943 on OPENCABINET, and 0.817 on CLOSEFRIDGE, ranking first on every task.

Training on demonstrations is prone to spurious correlations. In general, baselines fail most on subtle preferences such as keeping the kettle upright in KETTLEBOILING, holding the middle of the handle in OPENCABINET, and contacting the door far from the hinge in CLOSEFRIDGE. Compared to preferences with strong visual cues, these preferences are hard to capture from good demonstrations alone. In contrast, our method generates counterfactual negatives and uses the sensitivity loss to make the reward model learn them.

Counterfactual pairs improve performance at scale. With the generated counterfactual pairs, our method consistently outperforms baselines by at least 12% in average pairwise ranking accuracy on COFFEESERVEMUG across demonstration budgets (Fig. 5). Our method's accuracy increases from 0.872 at $N{=}2$ to 0.998 at $N{=}20$ and stays above every baseline at each N, while the strongest baseline (MaxEnt IRL) plateaus below 0.88 and Pixel-Mask + Raw RGB never exceeds 0.72. Notably, with 2 demonstrations per preference our method already matches the accuracy MaxEnt IRL reaches with 20 (0.872 vs. 0.875): the same performance with $10\times$ fewer demonstrations.

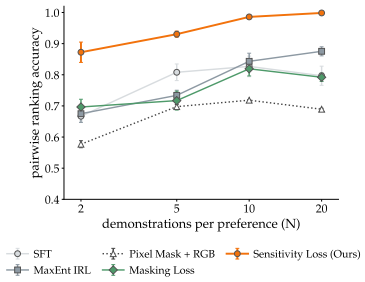
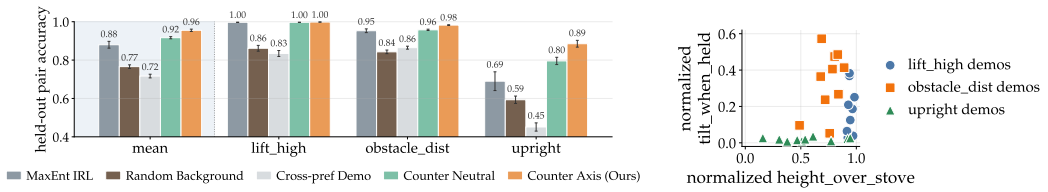


Figure 5: **Scaling N on COFFEESERVEMUG**: pairwise ranking accuracy vs. N. Mean $\pm$ standard error over 5 seeds.

4.3 RQ2: HOW SHOULD THE COUNTERFACTUAL TRAJECTORIES BE CONSTRUCTED?

Throughout the method and RQ1, the loser of every sensitivity pair is a *counterfactual negative*: a video generated by the world model with the target preference axis captioned to oppose the preference. This section validates that design choice by ablating how we obtain losers. All variants have the same winners as the demonstrations, use the same sensitivity loss formulation, and differ only in how the losers are constructed. We conduct this experiment on the KETTLEBOILING task.



(a) Pairwise Ranking Accuracy on Loser Variants

(b) Demonstrations GT Rewards

Figure 6: **Loser-type ablation on KETTLEBOILING** (N=10). (a) Per-preference pairwise ranking accuracy; every variant shares the same winners and differs only in the sensitivity-pair losers. Mean ± 1 standard deviation over 5 seeds. (b) Each preference's top-10 demonstrations, plotted on the ground-truth features of two of the three axes, min–max normalized over the trajectory pool.

Loser constructions. We compare five types of loser constructions. Suppose the user instructs "carry the kettle high" and demonstrates high carries. *Counter Axis* (ours) is the construction of §3.3: the world model gets demonstration's first frame and prompted with the target axis captioned to oppose the preference ("the kettle is carried *low* above the stovetop"), all other axes moderate. *Counter Neutral* uses the same generator but captions the target axis as moderate (a medium-height carry), asking whether generation must push past neutral to the opposing extreme. *Cross-pref Demo* skips generation and reuses the top-N demonstrations of a different preference (*e.g.*, the kettle-upright demonstrations, whose carry height is uncontrolled) as losers, asking whether already collected demonstrations from an unrelated preference can replace generation. *Random Background* draws losers uniformly from the unlabeled background trajectories, which vary on every axis at once, asking whether any non-expert trajectory is enough. *None* is the MaxEnt IRL baseline with no sensitivity pairs.

The loser must oppose the preference axis. Fig. 6 shows that both *Counter Neutral* and *Counter Axis* outperform *None*, the MaxEnt IRL baseline, whereas *Random Background* and *Cross pref Demo* perform worse than MaxEnt IRL in all preferences. This implies that constructing a loser that is different in the preference axis to be learned is important. Comparing *Counter Neutral* and *Counter Axis*, both have high pairwise ranking accuracy; *Counter Axis* marginally outperforms *Counter Neutral*, which indicates that increasing the gap between the winner and loser along the preference axis makes their difference clearer and easier to learn. Interestingly, generating losers without considering a specific preference axis results in *worse* reward models than standard MaxEnt IRL. *Random Background* and *Cross-pref* fall below the MaxEnt IRL baseline, because the margin we enforce is satisfied by whatever cue separates winners from losers, opening these alternative approaches to learning spurious correlations. This issue becomes most obvious on the `upright` preference. Inspecting the training data as shown in Fig. 6b, we found that `obstacle dist` demonstrations generally have higher lift height compared to `upright` demos. This means that the reward model may have learned `lift high` based on the sensitivity loss that samples winners from `upright` demos and losers from `obstacle dist` demos. Quantitatively, using cross-preference demonstrations as losers result in 0.451 accuracy for `upright` preference. The results imply that generating the loser on the opposing side of the target preference axis is important and no given cross-preference demonstrations or background trajectories can be used as losers directly.

4.4 RQ3: DOES THE LEARNED REWARD IMPROVE REAL-WORLD POLICY LEARNING?

On two real world manipulation tasks, we demonstrate how the learned reward can help learn a personalized policy. Our two tasks are: *mug transport*, the robot carries a mug across a table on which a laptop rests, with preferences to carry the mug high and to avoid going over the laptop; and *wine pour*, the robot pours from a wine bottle into a glass, with preferences over pour height and tilt angle. See Fig. 7 for illustration of the preferences.

Bar plots in Fig. 7 show that the reward models trained with our sensitivity loss achieve the highest pairwise ranking accuracy on all four preferences (0.84/0.99 on *mug transport*, 0.79/0.86 on *wine pour*), exceeding MaxEnt IRL and Pixel-Mask + Raw RGB by at most 0.40. Interestingly, SFT failed on `lift high` preference below 0.4, although the same preference was easily learnable in simulation experiments. This was mainly because we used a single positive demonstration for each preference axis in the *mug transport* task. The two demonstrations—one for each preference—differed in terms of distance to the laptop but had similar heights while carrying the mug.

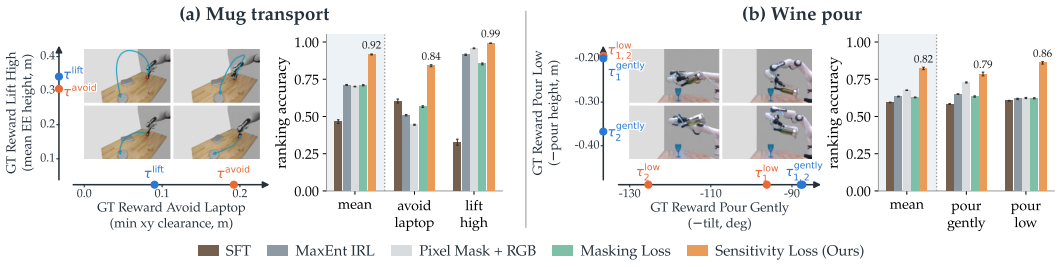


Figure 7: **Real-robot reward learning on the two Franka tasks.** (a) Mug transport (N=1 demonstration per preference); (b) wine pour (N=2). *Left* of each panel: task setup; markers show the demonstrations' ground-truth rewards, arrows the preferred direction. *Right*: pairwise ranking accuracy on held-out real episodes; mean ± 1 standard error over 10 seeds.

Wine pour is a slightly more complex task, so we used $N = 2$. Still, one of the `pour gently` demonstrations poured wine as low as the `pour low` demonstrations, and one of the `pour low` demonstrations tilted the wine bottle to a medium-low angle which was close to `pour gently` demonstrations. Since each counterfactual loser is generated from its own demonstration along the announced axis, the ordering signal survives this overlap: the sensitivity loss reaches 0.82 accuracy while every baseline stays at or below 0.72.

To learn personalized policies that correspond to each preference, we filter a set of unlabeled trajectories (149 trajectories for mug transport that vary carry height and distance to the laptop; 98 for wine pour that vary pour height and tilt angle) into the top-20 with the highest rewards, and train a state-based diffusion policy (Chi et al., 2025) (App. F.4). The policy-learning stage uses only the reward's scores over images, never over individual preference labels. Each policy is evaluated by the ground-truth preference return of rollouts on the robot from randomly sampled initial configurations per preference (20 rollouts for *mug transport*, 10 for *wine pour*).

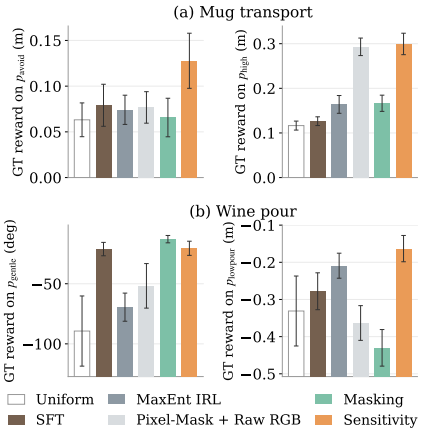


Figure 8: **Real-robot policy learning.** Bars show ground truth returns of rollouts from policies trained with top $M = 20$ episodes for each preference, mean ± 1 standard error.

Results in Fig. 8 show that the pattern of reward learning performance in Fig. 7 transfers to closed-loop behaviors. The uniform policy passes close over the laptop (average planar distance 0.063 m) and carries low (0.116 m). Filtering with the sensitivity-loss reward is the only condition that raises the ground-truth preference return in both *mug transport* preferences. Pixel-Mask + Raw RGB also learned lift high policy but marginally improved over uniform baseline on avoid laptop policy, as in Fig. 7. The transfer also holds on *wine pour*: sensitivity-loss filtering achieves the lowest pour height on `pour low` (0.16 m vs. 0.33 m for uniform) while matching Masking and SFT on `pour gently` (13–21° mean apex tilt vs. 89° for uniform). Masking and Pixel-Mask + Raw RGB fail to improve `pour low`, falling at or below the uniform baseline.

5 Conclusion

We present COVER, which learns a language-conditioned visual reward from a few demonstrations by imagining what the user does *not* want: a fine-tuned video world model generates counterfactuals opposing the stated preference axis, and a sensitivity loss scores each demonstration above its counterfactual, alongside MaxEnt IRL. COVER ranks first on all four simulated tasks and all four real-robot preferences, its filtered policies alone improving the ground-truth return on every preference; sensitivity pairs help only when the losers differ along the target axis. One limitation is that COVER trains the generator and reward per task over a fixed vocabulary of preference axes, and generation fidelity bounds what the reward can learn. Future directions include free-form preference axes, stronger video generators, and reward uses beyond demonstration filtering.

AI USE STATEMENT

In this work, we used generative AI tools to generate synthetic datasets, to run repetitive experiments, and to tune hyperparameters: a frontier VLM produces the pairwise-ranking captions used to fine-tune the counterfactual world model (§3.3), a use that is part of the method itself and is described and evaluated in the paper (App. D), and LLM-based assistants helped implement experiment tooling and follow protocols under author direction. We have not used generative AI tools to develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, clean and reformat datasets, or interpret results; assistance with translation and support for qualitative and thematic data analysis are not applicable to this work. Additionally, we used generative AI tools to create or modify scientific figures, suggest changes on some paragraphs to improve readability, along with validation from authors. We have reviewed all AI-assisted work: LLM-generated code was verified and tested for correctness by the authors, generated counterfactuals are evaluated quantitatively for fidelity, and all claims and prose were checked and revised by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work does not involve human subjects, personal data, or crowdsourced annotation: all demonstrations are either scripted in simulation or collected on robot hardware by the authors, and the datasets we release contain no personally identifiable information. Counterfactual generation is applied only to these demonstrations and only for reward model training. Learned rewards can inherit biases or artifacts of the underlying generative models; we bound this risk by quantifying generation fidelity and by evaluating every reward against ground-truth preference orderings before it is used for policy learning, and all real-robot rollouts were executed under direct author supervision. We foresee no negative societal impacts specific to this work beyond those general to robot learning.

REPRODUCIBILITY STATEMENT

Section 4.1 and the Appendix specify tasks and preference axes (App. F.1), data generation (App. F.2), camera configurations (App. F.3), the real-robot setup including policy training (App. F.4), loss implementations (App. A, C), training hyperparameters and seeds (App. F.6), and the evaluation protocol, including the trajectory-level held-out splits and no-repeat expert pooling we recommend as defaults. Code and dataset will be released to reproduce all reported numbers.

REFERENCES

Jordan Abi Nader, David Lee, Nathaniel Dennler, and Andreea Bobu. QuickLAP: Quick language-action preference learning for autonomous driving agents. In *Robotics: Science and Systems (RSS)*, 2026.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

Andreea Bobu, Yi Liu, Rohin Shah, Daniel S Brown, and Anca D Dragan. Sirl: Similarity-based implicit representation learning. In *Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction*, pp. 565–574, 2023.

Andreea Bobu, Andi Peng, Pulkit Agrawal, Julie A Shah, and Anca D Dragan. Aligning human and robot representations. In *Proceedings of the 2024 ACM/IEEE international conference on human-robot interaction*, pp. 42–54, 2024.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Daniel Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In *International conference on machine learning*, pp. 783–792. PMLR, 2019.

Daniel S Brown, Wonjoon Goo, and Scott Niekum. Better-than-demonstrator imitation learning via automatically-ranked demonstrations. In *Conference on robot learning*, pp. 330–359. PMLR, 2020.

Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. In *Forty-first international conference on machine learning*, 2024.

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris Coll-Vinent, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. Sam 3: Segment anything with concepts. In *International conference on learning representations*, volume 2026, pp. 138846–138923, 2026.

Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. In *Proceedings of Robotics: Science and Systems (RSS)*, 2024.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

Nathaniel Dennler, Stefanos Nikolaidis, and Maja Matarić. Contrastive learning from exploratory actions: Leveraging natural interactions for preference elicitation. In *2025 20th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pp. 778–788. IEEE, 2025.

Yuqing Du, Ksenia Konyushkova, Misha Denil, Akhil Raju, Jessica Landon, Felix Hill, Nando de Freitas, and Serkan Cabi. Vision-language models as success detectors. In Sarath Chandar, Razvan Pascanu, Hanie Sedghi, and Doina Precup (eds.), *Proceedings of The 2nd Conference on Lifelong Learning Agents*, volume 232 of *Proceedings of Machine Learning Research*, pp. 120–136. PMLR, 22–25 Aug 2023.

Chelsea Finn, Sergey Levine, and Pieter Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *International conference on machine learning*, pp. 49–58. PMLR, 2016.

Tobias Gerstenberg, Noah D Goodman, David A Lagnado, and Joshua B Tenenbaum. A counterfactual simulation model of causal judgments for physical events. *Psychological review*, 128(5): 936, 2021.

Google DeepMind. Veo 3: Generating video with audio. Technical report, `https://deepmind.google/models/veo/`, 2025.

David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. *Advances in neural information processing systems*, 31, 2018.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Joey Hejna, Rafael Rafailov, Harshit Sikchi, Chelsea Finn, Scott Niekum, W Bradley Knox, and Dorsa Sadigh. Contrastive preference learning: Learning from human feedback without reinforcement learning. In *International Conference on Learning Representations*, volume 2024, pp. 18770–18798, 2024.

Bohan Hou, Gen Li, Jindou Jia, Tuo An, Xinying Guo, Sicong Leng, Haoran Geng, Yanjie Ze, Tatsuya Harada, Philip Torr, et al. World model for robot learning: A comprehensive survey. *arXiv preprint arXiv:2605.00080*, 2026.

Minjune Hwang, Yigit Korkmaz, Daniel Seita, and Erdem Biyik. Causally robust reward learning from reason-augmented preference feedback. In *The Fourteenth International Conference on Learning Representations*, 2026a.

Minyoung Hwang, Alexandra Forsey-Smerek, Nathaniel Dennler, and Andreea Bobu. Masked irl: Llm-guided reward disambiguation from demonstrations and language. *IEEE International Conference on Robotics and Automation, ICRA*, 2026b.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *2023 IEEE/CVF international conference on computer vision (ICCV)*, pp. 3992–4003. IEEE, 2023.

Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.

Michael Laskin, Aravind Srinivas, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In *International conference on machine learning*, pp. 5639–5650. PMLR, 2020.

Kimin Lee, Laura M Smith, and Pieter Abbeel. Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 6152–6163. PMLR, 18–24 Jul 2021.

Tony Lee, Andrew Wagenmaker, Karl Pertsch, Percy Liang, Sergey Levine, and Chelsea Finn. Roboreward: General-purpose vision-language reward models for robotics. *arXiv preprint arXiv:2601.00675*, 2026.

Anthony Liang, Yigit Korkmaz, Jiahui Zhang, Minyoung Hwang, Abrar Anwar, Sidhant Kaushik, Aditya Shah, Alex S. Huang, Luke Zettlemoyer, Dieter Fox, Yu Xiang, Anqi Li, Andreea Bobu, Abhishek Gupta, Stephen Tu, Erdem Biyik, and Jesse Zhang. Robometer: Scaling general-purpose robotic reward models via trajectory comparisons. In *Robotics: Science and Systems 2026*, 2026.

Shaoteng Liu, Yuechen Zhang, Wenbo Li, Zhe Lin, and Jiaya Jia. Video-p2p: Video editing with cross-attention control. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8599–8608. IEEE, 2024.

Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023.

Yecheng Jason Ma, Vikash Kumar, Amy Zhang, Osbert Bastani, and Dinesh Jayaraman. Liv: Language-image representations and rewards for robotic control. In *International Conference on Machine Learning*, pp. 23301–23320. PMLR, 2023a.

Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. VIP: Towards universal visual reward and representation via value-implicit pre-training. In *The Eleventh International Conference on Learning Representations*, 2023b.

Yecheng Jason Ma, Joey Hejna, Chuyuan Fu, Dhruv Shah, Jacky Liang, Zhuo Xu, Sean Kirmani, Peng Xu, Danny Driess, Ted Xiao, et al. Vision language models are in-context value learners. In *International Conference on Learning Representations*, volume 2025, pp. 33984–34009, 2025.

Parsa Mahmoudieh, Deepak Pathak, and Trevor Darrell. Zero-shot reward specification via grounded natural language. In *International Conference on Machine Learning*, pp. 14743–14752. PMLR, 2022.

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. In Karen Liu, Dana Kulic, and Jeff Ichnowski (eds.), *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 892–909. PMLR, 14–18 Dec 2023.

Soroush Nasiriany, Sep Nasiriany, Abhiram Maddukuri, and Yuke Zhu. Robocasa365: A large-scale simulation framework for training and benchmarking generalist robots. In *International Conference on Learning Representations*, volume 2026, pp. 98643–98667, 2026.

Andrew Y Ng, Stuart Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, pp. 2, 2000.

NVIDIA, Arslan Ali, Junjie Bai, Maciej Bala, Yogesh Balaji, Aaron Blakeman, Tiffany Cai, Jiaxin Cao, Tianshi Cao, Elizabeth Cha, Yu-Wei Chao, Prithvijit Chattopadhyay, Mike Chen, Yongxin Chen, Yu Chen, Shuai Cheng, Yin Cui, Jenna Diamond, Yifan Ding, Jiaojiao Fan, Linxi Fan, Liang Feng, Francesco Ferroni, Sanja Fidler, Xiao Fu, Ruiyuan Gao, Yunhao Ge, Jinwei Gu, Aryaman Gupta, Siddharth Gururani, Imad El Hanafi, Ali Hassani, Zekun Hao, Jacob Huffman, Joel Jang, Pooya Jannaty, Jan Kautz, Grace Lam, Xuan Li, Zhaoshuo Li, Maosheng Liao, Chen-Hsuan Lin, Tsung-Yi Lin, Yen-Chen Lin, Huan Ling, Ming-Yu Liu, Xian Liu, Yifan Lu, Alice Luo, Qianli Ma, Hanzi Mao, Kaichun Mo, Seungjun Nah, Yashraj Narang, Abhijeet Panaskar, Lindsey Pavao, Trung Pham, Morteza Ramezanali, Fitsum Reda, Scott Reed, Xuanchi Ren, Haonan Shao, Yue Shen, Stella Shi, Shuran Song, Bartosz Stefaniak, Shangkun Sun, Shitao Tang, Sameena Tasmeen, Lyne Tchapmi, Wei-Cheng Tseng, Jibin Varghese, Andrew Z. Wang, Hao Wang, Haoxiang Wang, Heng Wang, Ting-Chun Wang, Fangyin Wei, Jiashu Xu, Dinghao Yang, Xiaodong Yang, Haotian Ye, Seonghyeon Ye, Xiaohui Zeng, Jing Zhang, Qinsheng Zhang, Kaiwen Zheng, Andrew Zhu, and Yuke Zhu. World simulation with video foundation models for physical ai. *arXiv preprint arXiv:2511.00062*, 2025.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

OpenAI. Sora 2 system card. Technical report, `https://openai.com/index/sora-2-system-card/`, 2025.

Jack Parker-Holder, Shlomi Fruchter, et al. Genie 3: A new frontier for world models. *Google DeepMind Blog*, 4, 2025.

Andi Peng, Aviv Netanyahu, Mark K Ho, Tianmin Shu, Andreea Bobu, Julie Shah, and Pulkit Agrawal. Diagnosis, feedback, adaptation: A human-in-the-loop framework for test-time policy adaptation. In *International conference on machine learning*, pp. 27630–27641. PMLR, 2023.

Andi Peng, Andreea Bobu, Belinda Z Li, Theodore R Sumers, Ilia Sucholutsky, Nishanth Kumar, Thomas L Griffiths, and Julie A Shah. Preference-conditioned language-guided abstraction. In *Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, pp. 572–581, 2024a.

Andi Peng, Ilia Sucholutsky, Belinda Li, Theodore Sumers, Thomas L Griffiths, Jacob Andreas, and Julie Shah. Learning with language-guided state abstractions. In *International Conference on Learning Representations*, volume 2024, pp. 38711–38744, 2024b.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Sriyash Poddar, Yanming Wan, Hamish Ivison, Abhishek Gupta, and Natasha Jaques. Personalizing reinforcement learning from human feedback with variational preference learning. *Advances in Neural Information Processing Systems*, 37:52516–52544, 2024.

Juan Rocamonde, Victoriano Montesinos, Elvis Nava, Ethan Perez, and David Lindner. Vision-language models are zero-shot reward models for reinforcement learning. In *International Conference on Learning Representations*, volume 2024, pp. 28446–28463, 2024.

Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, Sergey Levine, and Google Brain. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 1134–1141. IEEE, 2018.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seung Eun Yi, Michael Ramamonjisoa, Francisco Massa, Daniel HAZIZA, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Herve Jegou, Patrick Labatut, and Piotr Bojanowski. DINOv3. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856.

Sumedh Sontakke, Jesse Zhang, Séb Arnold, Karl Pertsch, Erdem Bıyık, Dorsa Sadigh, Chelsea Finn, and Laurent Itti. Roboclip: One demonstration is enough to learn robot policies. *Advances in Neural Information Processing Systems*, 36:55681–55693, 2023.

Thomas Tian, Chenfeng Xu, Masayoshi Tomizuka, Jitendra Malik, and Andrea Bajcsy. What matters to you? towards visual representation alignment for robot learning. In *International Conference on Learning Representations*, volume 2024, pp. 29609–29628, 2024.

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, et al. Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in neural information processing systems*, 33:5776–5788, 2020.

Yufei Wang, Zhanyi Sun, Jesse Zhang, Zhou Xian, Erdem Biyik, David Held, and Zackory Erickson. RL-VLM-f: Reinforcement learning from vision language foundation model feedback. In *Forty-first International Conference on Machine Learning*, 2024.

Jay Zhangjie Wu, Yixiao Ge, Xintao Wang, Stan Weixian Lei, Yuchao Gu, Yufei Shi, Wynne Hsu, Ying Shan, Xiaohu Qie, and Mike Zheng Shou. Tune-a-video: One-shot tuning of image diffusion models for text-to-video generation. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7589–7599. IEEE, 2023.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. In *International Conference on Learning Representations*, volume 2025, pp. 83048–83077, 2025.

Kevin Zakka, Andy Zeng, Pete Florence, Jonathan Tompson, Jeannette Bohg, and Debidatta Dwibedi. Xirl: Cross-embodiment inverse reinforcement learning. In *Conference on Robot Learning*, pp. 537–546. PMLR, 2022.

Jiahui Zhang, Yusen Luo, Abrar Anwar, Sumedh Anand Sontakke, Joseph J Lim, Jesse Thomason, Erdem Biyik, and Jesse Zhang. Rewind: Language-guided rewards teach robot policies without new demonstrations. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 460–488. PMLR, 27–30 Sep 2025.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.016.

Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pp. 1433–1438. Chicago, IL, USA, 2008.

A IRL Loss: Implementation

Both terms of Eq. (1) are estimated on minibatches of B trajectories drawn half from $\mathcal{D}^p_{\text{demo}}$ and half from $\mathcal{D}_{\text{samp}}$ (B=8: 4 demonstrations and 4 background trajectories; the demonstration side is smaller when fewer than 4 distinct experts are available). With labels $y_j = +1$ for demonstrations and $y_j = -1$ for background trajectories,

$$\mathcal{L}_{\text{IRL}} \;=\; \frac{1}{B}\sum_{j=1}^{B} \text{softplus}\big(-y_j\, R_\theta(\tau_j)\big) \;+\; \lambda_Z\, \tau_T \log \frac{1}{B}\sum_{j=1}^{B} \exp\big(R_\theta(\tau_j)/\tau_T\big), \tag{4}$$

where the first term is a smooth margin standing in for the likelihood term of Eq. (1), raising reward on demonstrations and lowering it on background trajectories, and the second is the batch estimate of $\log Z_\theta$ at temperature $\tau_T = 1$, downweighted by $\lambda_Z = 0.01$.

B Constrained Maximum-Entropy Reading of the Sensitivity Loss

Under the trajectory distribution $q_\theta(\tau \mid \ell_p) = \exp R_\theta(\tau \mid \ell_p)/Z_\theta$ of §3.1, reward differences equal log-probability ratios: both trajectories of a sensitivity pair share the same instruction ℓ_p and hence the same normalizer Z_θ, so

$$\log \frac{q_\theta(\tau^p_w)}{q_\theta(\tau^p_\ell)} = \big(R_\theta(\tau^p_w \mid \ell_p) - \log Z_\theta\big) - \big(R_\theta(\tau^p_\ell \mid \ell_p) - \log Z_\theta\big) = R_\theta(\tau^p_w \mid \ell_p) - R_\theta(\tau^p_\ell \mid \ell_p). \tag{5}$$

Requiring each winner to score above its loser by margin m therefore constrains the learned *distribution*: the winner must be at least e^m times more likely than the loser. Adding these constraints to the MaxEnt objective of Eq. (1) gives the constrained problem

$$\max_\theta\; \mathbb{E}_{\tau\sim\mathcal{D}^p_{\text{demo}}}[R_\theta(\tau \mid \ell_p)] - \log Z_\theta \quad \text{s.t.} \;\; \log \tfrac{q_\theta(\tau_w)}{q_\theta(\tau_\ell)} \geq m \;\; \forall(\tau_w, \tau_\ell) \in \mathcal{P}^{(p)}_{\text{sens}}. \tag{6}$$

A standard way to solve such an optimization problem is to move each constraint into the objective as a penalty that is zero when the constraint holds and grows with its violation. For Eq. (6) that penalty is exactly the hinge of Eq. (2): $\max\big(0, m - (R_\theta(\tau_w \mid \ell_p) - R_\theta(\tau_\ell \mid \ell_p))\big)$ is zero when the pair satisfies its ordering constraint and increases linearly with violation. Minimizing the total loss of Eq. (3) therefore solves Eq. (6) with penalty weight λ_{sens}, and pairs already separated by m contribute no gradient. The sensitivity loss thus does not change the form of the MaxEnt solution: the optimal distribution remains Gibbs in the reward, with the ordering constraints removing rewards that misorder any sensitivity pair.

C Masking Loss: Invariance Pairs and Analysis

Invariance pairs. Following Masked IRL (Hwang et al., 2026b), an *invariance pair* (τ_a, τ_b) agrees on the preference axis named by ℓ_p but differs in everything else; requiring $R_\theta(\tau_a \mid \ell_p) \approx R_\theta(\tau_b \mid \ell_p)$ forces the reward to ignore that variation. The world model of §3.3 can generate these pairs without additional training: for target preference p, every generation whose caption perturbs a *different* axis $q \neq p$ is paired with its own source trajectory. The two videos agree on axis p, whose caption phrase is unchanged, while differing along q, so under ℓ_p they show behavior of equal standing in a different visual context.

Loss. Writing $\mathcal{P}^{(p)}_{\text{inv}}$ for the set of invariance pairs of preference p, the loss is a symmetric MSE pulling the paired rewards to equality,

$$\mathcal{L}_{\text{mask}} = \mathbb{E}_{(\tau_a,\tau_b)\sim\mathcal{P}^{(p)}_{\text{inv}}}\big[\big(R_\theta(\tau_a \mid \ell_p) - R_\theta(\tau_b \mid \ell_p)\big)^2\big]. \tag{7}$$

The **Masking loss** baseline of §4.1 replaces the sensitivity term of Eq. (3) with this invariance term, optimizing $\mathcal{L} = \mathcal{L}_{\text{IRL}} + \lambda_{\text{mask}} \cdot \mathcal{L}_{\text{mask}}$ with λ_{mask}=1.

Why invariance cannot identify the axis. Invariance pairs only remove preference-irrelevant directions from the reward's dependence; they can never select the axis to attend to. Each sensitivity pair, by contrast, constrains the reward along the preference axis, pinning down what demonstrations and invariance pairs leave undetermined.

D WORLD MODEL TRAINING AND GENERATION FIDELITY

Fine-tuning recipes. We fully fine-tune Cosmos-Predict-2.5-2B (NVIDIA et al., 2025) on the background trajectories, each rendered as a 2×2 four-camera tile (512×512), for 1,000 steps on $8\times$H100 GPUs. Fine-tuning captions come from the VLM ranking pipeline of §3.3.

Generator fidelity. We evaluate the generator before any reward is trained: per preference axis in KETTLEBOILING task, we measure the ratio of generated counterfactual pairs whose prompted change is realized, judged by a Bradley-Terry judge fit on ground-truth-ordered held-in trajectory pairs (validation accuracy 0.96–1.00). On KETTLEBOILING the preference axes related with distances are rendered faithfully while `upright` reaches only 0.69 (Table 1).

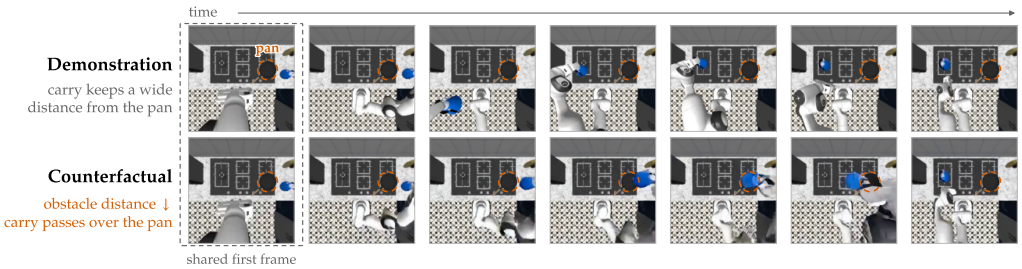
Table 1: **World-model fidelity on KETTLEBOILING**: fraction of generated counterfactual pairs whose prompted change is realized in the video.

	`lift_high`	`obstacle_dist`	`upright`
Realized rate	0.92	0.85	0.69

Human evaluation. The Bradley–Terry judge above measures only whether the prompted *target-axis* change is realized; the counterfactual premise additionally requires that everything else stay the same. We therefore evaluate off-axis preservation with a human evaluation on KETTLEBOILING and CLOSEFRIDGE (Table 2). For each preference axis, a rater viewed 10 demonstration–counterfactual pairs side by side, each generation seeded on the demonstration's first frame and prompted to flip only the target axis, and judged (i) *edit success*: whether the prompted target-axis change is realized, and (ii) *off-axis preservation*: whether no other aspect of the behavior or scene visibly changed. Edit success is 0.70–1.00 per axis, and most failures are preservation failures rather than missed edits: `contact_mid_height` on CLOSEFRIDGE realizes the prompted change in 0.80 of pairs but preserves the remaining axes in only 0.40, while the KETTLEBOILING axes preserve at 0.90 and also maintain other axes preserved over 0.70 rate.

Table 2: **Human evaluation on world model counterfactual generation** of generated counterfactuals on KETTLEBOILING and CLOSEFRIDGE: per axis, the fraction of 10 demonstration–counterfactual pairs whose prompted target-axis change is realized (edit success) and whose other behavior and scene aspects are unchanged (off-axis preservation).

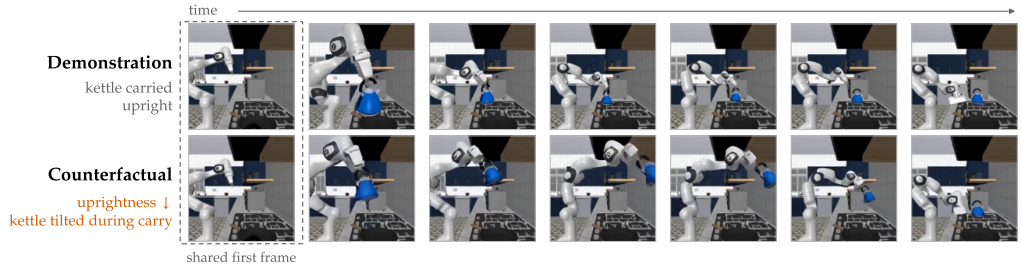
Task	Axis	Edit success	Off-axis preservation
KETTLEBOILING	`lift_high`	0.80	0.90
	`obstacle_dist`	1.00	0.90
	`upright`	0.80	0.90
	mean	0.87	0.90
CLOSEFRIDGE	`contact_mid_height`	0.80	0.40
	`push_outward`	1.00	0.80
	`close_left_first`	0.70	0.80
	mean	0.83	0.67



(a) `obstacle_dist`: wide clearance from the pan → carry passes over the pan (top camera).



(b) `lift_high`: kettle carried high over the stove → kettle kept at counter level (front camera).



(c) `upright`: kettle carried upright → kettle tilted during the carry (side camera).

Figure 9: **Qualitative counterfactual generations on KETTLEBOILING**, one example per preference axis. Each panel shows a demonstration (top row) and a counterfactual generated by the fine-tuned world model (bottom row), seeded on the demonstration's first frame (dashed box) and prompted to flip only the target axis. One camera view is shown per axis.

Qualitative examples. Fig. 9 shows one demonstration–counterfactual pair per KETTLEBOILING preference axis. In each pair the generation is seeded on the demonstration's first frame and prompted with a caption that flips only the target axis: the carry path passes over the pan instead of keeping a wide distance (`obstacle_dist`), the kettle stays at counter level instead of being lifted high (`lift_high`), and the kettle is tilted during the carry instead of being held upright (`upright`), while the scene, task, and remaining axes follow the demonstration. Fig. 10 shows the same construction on the real-robot MUG TRANSPORT task: conditioned on a real demonstration's first frame, the world model keeps the mug close to the table instead of carrying it high (`place_mug_high`) and routes the carry directly over the laptop instead of keeping a wide clearance (`avoid_laptop`).

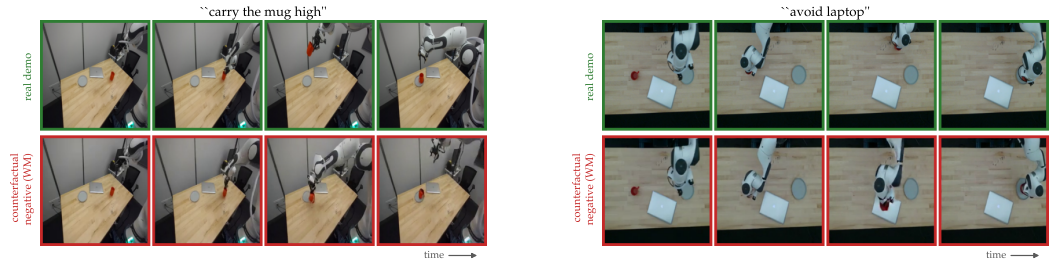


Figure 10: **Qualitative counterfactual generations on the real-robot MUG TRANSPORT task**, one example per preference axis. Each panel shows a real demonstration (top row, green) and the counterfactual negative generated by the fine-tuned world model (bottom row, red), conditioned on the demonstration's first frame and prompted to oppose the target axis: the high carry becomes a low carry (`place_mug_high`, third-person camera) and the wide-clearance carry passes directly over the laptop (`avoid_laptop`, top camera).

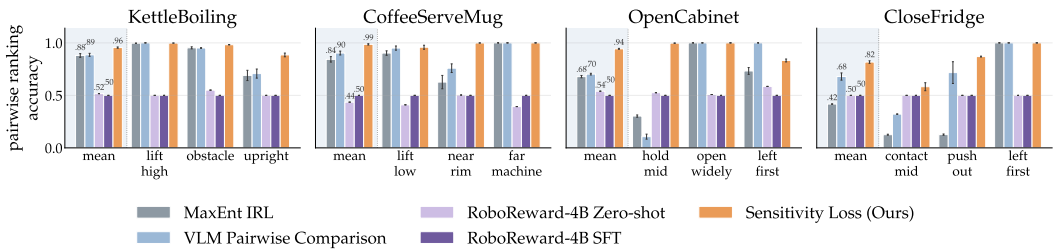


Figure 11: **Alternative supervision sources vs. counterfactual sensitivity pairs** at $N{=}10$ demonstrations per preference: VLM pairwise comparisons on background trajectories, and a pretrained foundation reward model (RoboReward-4B) zero-shot and fine-tuned on the same N demonstrations as the other baselines, evaluated on the same held-out pairs as Fig. 4. The VLM pairwise comparison loss trains the identical reward model with the identical hinge objective; only the pair source differs. The RoboReward conditions use the model's native discrete score, with tied pairs receiving 0.5 credit. The leftmost group in each panel is the mean over that task's preferences. Error bars show ± 1 standard error over 5 seeds; the RoboReward bars show none (zero-shot is deterministic, and the fine-tuned model collapses to a constant score; see text).

E ADDITIONAL EXPERIMENTS

E.1 VLM PAIRWISE COMPARISONS AND A PRETRAINED FOUNDATION REWARD MODEL AS ALTERNATIVE SUPERVISION

A natural reference-free alternative to generating counterfactual negatives is to ask a VLM to compare trajectories directly. For each preference axis, we query the same frontier VLM used for world-model captioning (App. D) with pairwise comparisons between randomly sampled held-in trajectories (eight comparisons per trajectory, presentation order randomized), phrasing each question along the stated preference. Every answered comparison is kept as one preference pair, and the resulting pairs are construct $\mathcal{P}^{p}_{\text{sens}}$. Training is otherwise identical to COVER: the same reward architecture and instruction conditioning, the same demonstrations and MaxEnt IRL term, and the same sensitivity loss parameters. Only the source of pairs changes: VLM-labeled pairs of background trajectories replace the world model generated counterfactual pairs.

Fig. 11 reports the average pairwise ranking accuracy at $N{=}10$. VLM pairwise supervision improves over MaxEnt IRL on every task, most visibly on CLOSEFRIDGE ($0.417 \rightarrow 0.679$), but underperforms against our method on all four tasks. The gap tracks how reliably the VLM can judge each axis from trajectory videos: agreement between the raw comparisons and ground truth is high on axes with salient visual evidence (0.94 on KETTLEBOILING lift height, 0.80–0.91 on the COFFEESERVEMUG axes, 0.78–0.87 on CLOSEFRIDGE contact height and push position), lower

on subtler geometry (0.63–0.72 on KETTLEBOILING obstacle clearance and tilt), and at or below chance on the OPENCABINET handle-grasp position (0.40) and the CLOSEFRIDGE door-order axis (0.49). On OPENCABINET the resulting reward accordingly stays at the MaxEnt IRL level performance. Counterfactual generation avoids this failure mode: rather than requiring an annotator to rank real trajectories along a subtle axis, it *constructs* the winner and loser to differ along that axis, so the supervision remains targeted even where the VLM is not perfect to judge real videos.

Pretrained foundation reward model. We additionally test whether a pretrained robot reward model can supply the preference signal directly. RoboReward-4B (Lee et al., 2026) is a Qwen3-VL-4B model fine-tuned to output a discrete progress score from 1 to 5 given a rollout video and a task instruction. We evaluate it zero-shot by phrasing each preference as the task instruction under the model's original scoring rubric, and we also fine-tune it with its native SFT objective—cross-entropy on the score token—using LoRA (rank 16) on the same $N{=}10$ demonstrations per preference as all other baselines, each labeled with score 5 (perfect completion); no negatives or additional labels are provided. Scores are read out as the model's discrete answers, and tied pairs receive 0.5 credit. On COFFEESERVEMUG, the RoboReward conditions are evaluated on a fixed 150-pair-per-axis subsample of the held-out pairs (per-clip scoring cost); the other tasks use the full sets.

Zero-shot, RoboReward assigns a nearly constant score per axis—its rubric judges end-of-episode progress from the final state, whereas our preferences are properties of the path—so pairwise accuracy stays at chance (0.44–0.54 pooled). Fine-tuning on demonstrations alone collapses the model entirely: with every training target equal to 5 there is no negative signal, the fine-tuned model assigns score 5 to every held-out trajectory, and every evaluation pair ties, yielding 0.500 on all four tasks and every preference axis. The from-scratch **SFT** baseline of Fig. 4 degrades more gracefully under the same demos-as-positives supervision because its continuous reward still ranks trajectories by proximity to the demonstrations, whereas the discrete five-level rubric offers no such gradation. Extra annotation only partially repairs this: in a variant with twice the budget ($2N$ direct labels per preference: the top-10 trajectories by ground truth labeled 5, the bottom-10 labeled 1), the fine-tuned model recovers axes with salient visual evidence but remains at chance on fine geometry (`upright` 0.490 on KETTLEBOILING; `far_from_machine` 0.509 on COFFEESERVEMUG).

E.2 SENSITIVITY-LOSS FORM: HINGE VS. BRADLEY–TERRY

As an ablation study on how to implement the sensitivity loss, we swap the hinge loss of Eq. (2) for the Bradley–Terry logistic loss $-\log\sigma\big(R_\theta(\tau_w \mid \ell_p) - R_\theta(\tau_\ell \mid \ell_p)\big)$ on KETTLEBOILING, keeping the counterfactual pairs and reward model specifics. Figure 12 reports per-preference held-out accuracy under the full objective (IRL + sensitivity loss) at demonstration budgets $N \le 20$. The hinge is moderately better on `lift_high` at small and intermediate budgets (0.905 vs. 0.800 at $N{=}10$), and the two forms are at parity on `obstacle_dist` and on `upright`, which stays near its generation ceiling (0.62–0.66) under either loss. A pairs-only regime that trains on the same pairs without the IRL term shows the same picture: the two losses match on all three axes, and accuracy is nearly flat in N because that regime is supervised by the generated pairs alone (`lift_high` is already at 0.97 with a single demonstration, `obstacle_dist` rises modestly, $0.845 \to 0.890$, and `upright` stays near 0.59–0.62). We therefore keep the hinge, which also admits the constrained maximum-entropy interpretation in App. B.

F EXPERIMENTAL SETUP DETAILS

F.1 TASKS AND PREFERENCE AXES

Table 3 defines the preferred behavior along every preference axis, for the four simulated RoboCasa tasks and the two real Franka tasks of §4.1.

F.2 DATA GENERATION IN SIMULATION

Data. Every simulated-task trajectory descends from a *single* scripted base demonstration per task per scene in RoboCasa365 (Nasiriany et al., 2026) dataset. For KETTLEBOILING we sample 1,700 counterfactual variants of the base demonstration by Jacobian-based end-effector deformation, of

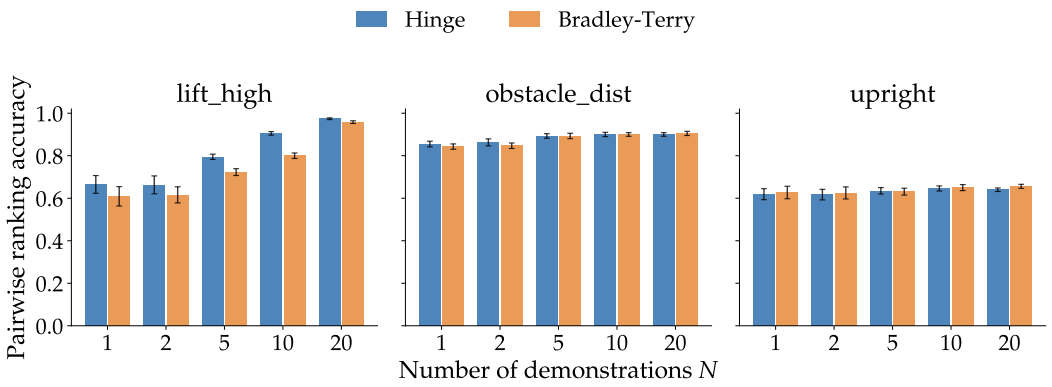


Figure 12: **Hinge vs. Bradley–Terry (BT) sensitivity loss** on KETTLEBOILING: held-out pairwise ranking accuracy per preference axis under the full training objective (IRL + sensitivity loss), at demonstration budgets $N \leq 20$. Bars show the mean over 10 seeds; error bars are ± 1 standard error.

Table 3: **Preference axes.** Preferred behavior per axis for the simulated and real tasks.

Task	Axis	Preferred behavior
KETTLEBOILING	`lift_high`	larger clearance above the stovetop during the counter-to-stove carry
	`obstacle_dist`	kettle kept farther from the pan on the non-target burner
	`upright`	kettle carried upright with minimal tilt
COFFEESERVEMUG	`lift_low`	served mug carried low, close to the counter
	`near_rim`	placement closer to the counter edge
	`far_from_machine`	placement farther from the coffee machine along the counter
OPENCABINET	`hold_handle_middle`	grasp near the handle midpoint, not its ends
	`open_widely`	both doors opened to a wider final angle
	`open_left_first`	trajectories ordered by which door begins opening first
CLOSEFRIDGE	`contact_mid_height`	contact near the middle of the handle
	`push_outward`	contact placed toward the hinge side of the door panel
	`close_left_first`	trajectories ordered by which door is closed first
MUG TRANSPORT	`avoid_laptop`	carry path with larger horizontal clearance from the laptop
	`place_mug_high`	mug carried higher above the table
WINE POUR	`pour_low`	pouring from a lower height
	`pour_gently`	pouring with a smaller tilt angle

which 980 survive physics filters; every survivor completes the task and differs only in motion style. COFFEESERVEMUG follows the same recipe on five base episodes in distinct kitchen scenes (1,001 variants each); OPENCABINET and CLOSEFRIDGE use stitched kinematic edits of their base demonstrations (151 variants each).

F.3 CAMERAS IN SIMULATION EXPERIMENTS

Camera configurations for the simulated tasks are task-specific: KETTLEBOILING uses four fixed cameras (`top`/`front`/`right`/`side`), including two mutually orthogonal vertical views that disambiguate carry height from kettle tilt, COFFEESERVEMUG four fixed external cameras

(top/front/left_back/right_back), and OPENCABINET/CLOSEFRIDGE a single front camera consumed as a two-panel (wide plus zoomed) tile.

F.4 REAL-ROBOT SETUP

Hardware and cameras. Both real tasks run on a Franka Research 3 robot observed by four RGB cameras: a third-person view, a wrist-mounted camera, a left-side view, and a top-down view. The reward model encodes the four camera streams as in App. F.5; the world model consumes and generates the four views as a single 2×2 tiled video, as in KETTLEBOILING.

Demonstrations and unlabeled pools. MUG TRANSPORT uses $N=1$ demonstration per preference and WINE POUR $N=2$ (§4.4). The unlabeled pools of §4.4 carry no preference labels: for MUG TRANSPORT, 149 episodes (49 episodes from the demonstration-collection sessions plus 100 scripted-planner episodes on a 10×10 grid over average carry height, 0.10–0.42 m, and laptop clearance); for WINE POUR, 98 scripted-planner episodes on a 10×10 grid over pour height (0.19–0.42 m) and bottle tilt (89–126°; two grid cells were kinematically unreachable).

Ground-truth measurements. Ground-truth preference values are computed from recorded end-effector trajectories and measured object poses, and are used for evaluation only. For MUG TRANSPORT, the carry phase is the segment with the gripper closed; carry height is the mean end-effector height above the table over this segment, and laptop clearance is the minimum planar distance between the carry path and the laptop footprint. For WINE POUR, both axes are measured at the pour apex: the pour height and the tilt of the bottle's long axis from vertical.

Counterfactual generation. For each real task we fine-tune the same Cosmos-Predict-2.5-2B backbone on the task's captioned episodes rendered as 2×2 four-camera tiles (the recipe of App. D) and generate counterfactuals from the demonstrations' first frames; a compliance filter removes generations whose prompted change is not realized.

Policy training. Reward-filtered behavior cloning selects, per preference, the top-M ($M=20$) pool episodes ranked by the learned reward (scores standardized within each of the 10 reward seeds, then averaged). On the selected episodes we train a state-based diffusion policy (Chi et al., 2025) per method and preference: observations are 2 steps of the 10-D end-effector state (position, 6D rotation, gripper width), actions are 7-D, prediction horizon 16 with 8 executed actions per step, DDPM with 100 diffusion steps, batch size 256, learning rate 10^{-4} with cosine schedule, 301 epochs; both tasks share this configuration. Error bars in Fig. 8 are computed over on-robot rollouts of a single trained policy per method and preference (§4.4).

F.5 REWARD MODEL ARCHITECTURE

The input to the reward model is multiple camera observations, task instructions, and preference instructions. Each camera's frames are encoded by frozen SigLIP 2-base (Tschannen et al., 2025) and DINOv3-ViT-B/16 (Siméoni et al., 2026), and instructions by a frozen language encoder MiniLM (Wang et al., 2020). Visual embeddings are concatenated across camera views, passed to a FiLM layer (Perez et al., 2018) conditioned on the language embedding, then a trainable 2-layer MLP and a linear head that outputs a scalar reward r_t per timestep; the trajectory reward is the average $R_\theta(\tau) = \frac{1}{T}\sum_t r_t$.

F.6 REWARD TRAINING DETAILS

Averaging per-timestep rewards keeps R_θ invariant to episode length, so the Gibbs weight $\exp R_\theta(\tau)$ and the fixed margin m of Eq. (2) compare trajectories by preference satisfaction rather than duration. We train for 50 epochs with batch size 8 and learning rate 10^{-4}. The sensitivity loss weight is $\lambda_{\text{sens}}=1$ on KETTLEBOILING/COFFEESERVEMUG and 3 on the door tasks. All reported reward results are averaged over 5 random seeds per simulated task and 10 seeds per real task; each simulated-task seed trains a single reward model jointly on the task's three preference axes, and error bars show ±1 standard error unless a figure states otherwise.

F.7 EVALUATION DETAILS

Pairwise ranking accuracy is computed over fixed per-axis pair sets in which the two trajectories have a clear ground-truth gap along the announced preference axis; the same pair sets are shared by all methods and seeds. In simulation, KETTLEBOILING uses 442/340/500 pairs for `lift_high`/`obstacle_dist`/`upright`, both endpoints of every pair lying in its 108-trajectory held-out subset; OPENCABINET, CLOSEFRIDGE, and COFFEESERVEMUG use 400 pairs per axis, with both endpoints of every COFFEESERVEMUG pair lying in the held-out kitchen scene. On the real robot, *mug transport* uses 210 pairs for carry height and 350 pairs for laptop clearance, drawn from the 49 demonstration-session episodes; *wine pour* uses 167 pairs for pour height and 162 pairs for tilt, drawn from the 98 trajectories. For all tasks, pairs involving a preference's own demonstrations are excluded from that preference's evaluation set.