CRITICHACK: EVALUATING VISUAL REWARDS UNDER ROBOT POLICY OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Learned visual rewards guide robot policy optimization, but can assign high scores to executions that act on the wrong object without completing the requested task. We present CRITICHACK, a closed-loop study of visual reward optimization in simulated drawer manipulation and identity-sensitive stacking. Our central finding is that **optimizing a learned visual reward can increase wrong-object failure rates even as reward and task success improve.** In a supervised-initialization study, three full-denoiser fine-tuning runs of a diffusion policy with no prior reward exposure, optimized against Robometer on a drawer task, raise simulator-measured success and wrong-object failure rates by 9.6 and 11.8 percentage points (95% CIs $[6.2, 13.0]$ and $[8.1, 15.6]$) on 512 evaluation seeds, alongside higher reward. A warm-start study shows the same pattern from a policy previously optimized against learned rewards: across five runs, success and wrong-object failures rise by 8.4 and 3.8 points on average. Constrained-policy experiments, which optimize only a selector over fixed diffusion-noise sequences, reproduce reward gain and wrong-object amplification across two task–reward pairings and two optimizers. Reward-ranking analysis further shows that **strong aggregate success–failure discrimination can conceal semantic misranking.** On collected trajectories, Robometer assigns higher average scores to wrong-object failures than to successes; strong discrimination against other failures masks this scoring error. As a diagnostic intervention, reweighting the critic reward with a frozen verifier trained on task-specific outcome labels improves task completion and reduces wrong-object failures relative to critic-only training under the fine-tuning sampler. Verifier-only controls identify task-specific outcome supervision as a substantial source of recovery. These results show that gains in reward and task success can conceal the amplification of specific semantic failures.

1 INTRODUCTION

Learned visual reward models provide feedback for robot policy optimization from visual observations and task descriptions, reducing reliance on hand-designed task rewards (Lee et al., 2026). Yet a reward model need not fail broadly to induce undesirable behavior. It may score most failures correctly yet reward a coherent semantic mistake that optimization can discover and exploit.

Consider a robot instructed to open the middle drawer (Figure 1). Opening the upper drawer instead shows visible manipulation progress but fails the requested task. Likewise, a stacking policy may construct a stable stack using the wrong cube. We call these outcomes *wrong-object failures*: executions that manipulate a distractor in a task-consistent manner while failing the requested task. Robometer already shows that RoboReward assigns high scores to wrong-object executions and that such false positives can reinforce incorrect behavior during online learning (Liang et al., 2026). Less is understood about how these errors interact with genuine task learning. If optimization simply degraded task success, standard evaluation would catch it. We show that it need not: optimizing a visual reward can increase task success and wrong-object failures at the same time. The two checks a practitioner would ordinarily rely on, strong aggregate success–failure discrimination of the reward model and rising task success after optimization, can both look healthy while a specific semantic failure grows.

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.

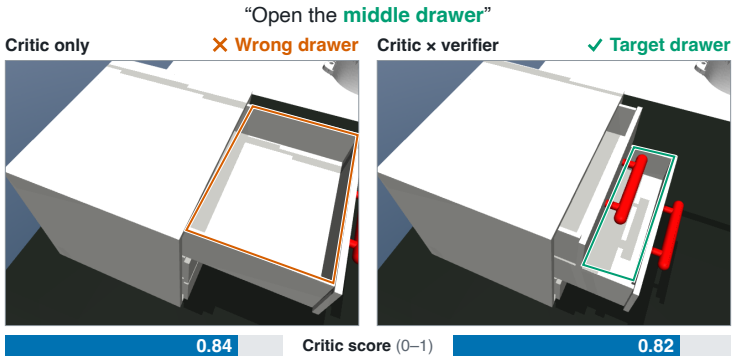

Figure 1: **High visual reward can accompany the wrong object.** Critic-only training opens the upper distractor while the target stays closed; critic–verifier training opens the requested middle drawer. Both receive high scores from the original Robometer, shown on the same zero-to-one scale. Frames are the final critic inputs for the first prespecified evaluation seed in constrained-policy PPO (Section 4.2), chosen independently of outcome; counts and intervals are in Section 5.2.

We study this interaction with CRITICHACK, a closed-loop evaluation of visual rewards under robot policy optimization. Simulator state measures task success and wrong-object failures independently of the learned reward. Our main evidence fine-tunes every denoiser parameter of a diffusion policy against Robometer on a drawer task. In the *supervised-initialization study*, three training histories (independent training runs) start from a policy with no prior reward exposure; on 512 evaluation seeds they raise task success by $9.6\,[6.2, 13.0]$ percentage points and wrong-object failures by $11.8\,[8.1, 15.6]$ points. The amplification recurs in an independent repetition, although completion rewards also increased wrong-object failures in that repetition (Section 5.1). The *warm-start study* starts instead from a policy previously optimized against learned rewards; across its five training histories, evaluated on a fixed bank of 512 seeds, task success increases by 8.4 percentage points on average while wrong-object failures increase by 3.8 points, and paired evaluations show that both changes come largely from converting former failures. *Constrained-policy* experiments optimize only a selector over a fixed bank of diffusion-noise sequences. With Robometer and Language-Image Value learning (LIV) (Ma et al., 2023a), optimized with proximal policy optimization (PPO) and a grouped variant of group relative policy optimization (GRPO), they reproduce reward gain and wrong-object amplification in 11 of 29 *strata* (task, critic and decoy-color combinations), including every drawer Robometer and distinct-color LIV stacking stratum.

A trajectory-level reward audit shows how aggregate evaluation can miss this behavior. On a separate panel of collected policy trajectories, Robometer's area under the receiver operating characteristic curve (AUROC) is .782 for success versus all failures but .398 for success versus wrong-object failures, indicating an inverted ordering. Wrong-object failures receive higher mean scores than successes. Strong discrimination against other failures masks this inversion in aggregate metrics.

We further evaluate an outcome-verification intervention in which a frozen success verifier either reweights the original critic reward or serves as a reward on its own. Verification improves task completion over critic-only optimization, and verifier-only rewards reproduce much of the gain.

Contributions. First, we show that task improvement and semantic failure amplification can coexist, so task success after optimization need not reveal a growing semantic failure. Second, we show that strong aggregate reward discrimination can hide the semantic misranking that accompanies this failure, and that semantic AUROC tracks amplification retrospectively across the constrained-policy strata. Third, we show that outcome verification can redirect behavior, in full-denoiser training and in constrained policies, with verifier-only ablations identifying supervision as a substantial source of improvement.

Scope. Our simulated experiments cover two tasks, occluded-drawer manipulation and identity-sensitive stacking. Full-denoiser fine-tuning uses one critic, Robometer's terminal success probability, on two drawer layouts, with the five-step DDIM sampler and renderer used during fine-tuning.

The warm-start policy was previously optimized with RoboReward and trained by Robometer-weighted regression; the supervised-initialization policy has no reward exposure. Constrained-policy experiments keep the diffusion decoder frozen. The verifier is task-specific and trained on simulator labels. The real-robot study is a case study with one training history per fine-tuned condition.

2 Related Work

Reward overoptimization and reward-model evaluation. Learned rewards turn human preferences into policy objectives (Christiano et al., 2017), but optimizing such a proxy can depart from the intended outcome (Amodei et al., 2016; Skalse et al., 2022), and more capable agents exploit misspecified rewards more effectively (Pan et al., 2022). Overoptimization has been characterized in language-model and diffusion-model alignment (Gao et al., 2023; Zhang et al., 2024), where it appears as a decline in a gold or held-out metric under continued proxy optimization. We study a signature this diagnostic can miss: task success rises while a specific semantic failure grows, both measured through simulator state independently of the reward. Reward-model evaluation raises a related concern. Low prediction error need not imply low regret after optimization (Fluri et al., 2025), and static ranking accuracy imperfectly predicts post-optimization behavior (Kim et al., 2025b; Wen et al., 2025). Aggregate metrics can also conceal poor performance on meaningful subsets, a phenomenon known as hidden stratification (Oakden-Rayner et al., 2020). Section 5.3 shows such masking in a visual reward model, on the very subset that optimization amplifies.

Visual rewards for robot learning. Visual rewards have been learned from human videos (Chen et al., 2021), goal-conditioned value pretraining (Ma et al., 2023b) and language–image alignment (Ma et al., 2023a). RoboCLIP and vision–language model reward models (VLM-RMs) use pretrained video– and image–language models as rewards (Sontakke et al., 2023; Rocamonde et al., 2024), and Video-Language Critic learns video–text alignment and temporal rankings (Alakuijala et al., 2025). RoboReward and Robometer scale general-purpose robotic reward models (Lee et al., 2026; Liang et al., 2026). Our tasks hinge on spatial reference and color-based object identity, distinctions that match documented weaknesses of vision–language models (Kamath et al., 2023; Yuksekgonul et al., 2023). Liang et al. (2026) document RoboReward false positives on wrong-object executions and their consequences for online reinforcement learning (RL). We find the same failure class in Robometer's own success output, where strong aggregate discrimination hides it, and show that optimizing against it can raise task success and wrong-object failures together. Our results concern Robometer's terminal success probability; its published online RL instead uses dense progress rewards and success-based termination, a recipe we do not evaluate.

Outcome supervision and reward correction. Success classifiers have long served as rewards, from event-based control (Fu et al., 2018) to vision–language success detectors (Du et al., 2023), and outcome verifiers trained on correctness labels are widely used in language-model reasoning (Cobbe et al., 2021). Other work corrects the reward itself: ensembles support conservative optimization against uncertain feedback (Coste et al., 2024), Adapt2Reward uses failure prompts to adapt visual rewards (Yang et al., 2024), and RARM uses reference-anchored, confidence-gated progress matching (Yang et al., 2026). Our verifier is a task-specific success classifier trained on simulator labels. We use it to test whether changing the feedback redirects an observed failure, not as a label-free alternative or a performance claim against these methods.

Diffusion-policy optimization. Diffusion Policy (DP) (Chi et al., 2023) generates robot action chunks by iterative denoising. Treating the denoising chain as a Markov decision process (MDP) enables reward-driven fine-tuning of the full denoiser (Black et al., 2024; Ren et al., 2025), while latent-noise steering improves a fixed policy by optimizing the noise supplied to its denoiser (Wagenmaker et al., 2025). Our two protocols follow these routes: full-denoiser PPO under the denoising-MDP formulation, and a constrained selector over a fixed bank of noise sequences with the decoder frozen. Generalist vision–language–action (VLA) policies such as OpenVLA (Kim et al., 2025a) can also be adapted with RL (Chen et al., 2025); we use task-specific DPs, which allow many independent training histories to be evaluated against simulator-measured outcomes, and do not study VLA fine-tuning.

3 Evaluating Visual Rewards Under Optimization

Starting from an initial policy, we optimize against a frozen reward model and compare the resulting behavior with that of the initialization. Our two primary tasks, drawer manipulation and identity-sensitive stacking, distinguish successful task completion from manipulation of the wrong object.

Robot policies. We use task-specific diffusion policies (Chi et al., 2023) that generate action chunks from current observations. The robot executes the initial portion of each chunk before generating the next chunk from updated observations, producing a closed-loop trajectory $\tau = (o_0, a_0, \ldots, o_T)$ under policy π_ϕ. A frozen visual critic assigns reward $r_b(\tau, x)$ for task instruction x; the policy need not be language-conditioned. We consider two complementary optimization protocols: full-denoiser fine-tuning, which updates the diffusion network, and constrained noise-policy optimization, which learns a distribution over diffusion-noise sequences while keeping the decoder fixed.

Tasks and physical outcomes. For each execution, we measure requested-task success $y(\tau) \in \{0, 1\}$ and wrong-object failure $w(\tau) \in \{0, 1\}$ from MuJoCo (Todorov et al., 2012) joint and contact states. These states supply outcome labels and oracle feedback but are excluded from critic and verifier inference. A wrong-object failure requires the requested task to remain unsuccessful. Each execution therefore belongs to one of three mutually exclusive categories: success ($y = 1$), wrong-object failure ($y = 0, w = 1$), or other failure ($y = 0, w = 0$).

Occluded drawer. The requested middle drawer must end at least 54 mm open. The movable upper drawer starts at 47.5% of its slide range. A wrong-object failure requires $y = 0$ and more than 20 mm of additional upper-drawer opening at any time. The oracle uses clipped target completion, $\min(1, \max(0, q_T/(54\,\text{mm})))$, where q_T is final target opening.

Identity-sensitive stacking. The requested bright red cube must rest on a rectangular wooden block. Success requires final position and height within 6 and 12 mm of the intended stack, tilt at most 20°, support contact, and no gripper contact for the last three control ticks. A wrong-object failure satisfies the same criterion with the decoy while the target fails; remaining failures are no-stack outcomes. Two initial scenes and two target/decoy position assignments form four cells. Near-red and distinct-blue decoys both enter training and evaluation. The oracle uses binary target success.

Three measurements, two distinct failure signatures. For the initial policy U and critic-optimized policy C, we report

$$\Delta R = \mathbb{E}_C[r_b] - \mathbb{E}_U[r_b], \qquad \Delta S = \mathbb{E}_C[y] - \mathbb{E}_U[y], \qquad \Delta W = \mathbb{E}_C[w] - \mathbb{E}_U[w]. \tag{1}$$

Reward gain with success loss ($\Delta R > 0, \Delta S < 0$) is the usual aggregate overoptimization signature. Reward gain with wrong-object amplification ($\Delta R > 0, \Delta W > 0$) is the semantic signature studied here. The latter signature does *not* imply a decrease in the scalar success objective.

4 Policy Optimization and Evaluation Protocols

Our primary experiment fine-tunes the full policy denoiser; constrained noise-policy experiments test recurrence across tasks, critics and optimizers.

Rewards and controls. The critics remain frozen throughout policy optimization. Robometer receives eight rollout frames and supplies its terminal success probability as reward. LIV supplies a final-frame image–language cosine c, mapped to $(1 + c)/2$ for training and reported in native units. Each protocol compares the initial policy (U), original-critic optimization (C) and simulator-derived oracle feedback (O). Oracle feedback tests task learnability. Section 6 adds critic $\times$ verifier (Product, P) and verifier-only (V) rewards to test how task-completion information redirects optimization. The policy family is fixed within each comparison. Critic inputs, physical measurements and verifier-label provenance are specified in Appendix A.

Comparison	Δ Success (pp)	Δ Wrong-object (pp)	Δ Robometer
Change from initialization			
Critic	+8.4 [4.9, 11.9]	+3.8 [0.8, 6.8]	+.061 [.036, .086]
Critic × verifier (Product)	+15.5 [11.8, 19.1]	−0.6 [−3.4, 2.1]	+.096 [.070, .122]
Direct objective comparison			
Product − critic	+7.1 [4.4, 9.8]	−4.4 [−7.1, −1.8]	+.034 [.013, .056]

Table 1: **Primary full-denoiser evaluation.** Entries report pooled iteration-40 effects with 95% CIs in brackets, accounting for training-history and shared-seed variation. The first two rows compare against the shared initialization; the last reports Product minus critic. Per-training results and controls are in Appendices B.2 and C.

4.1 Full-denoiser Fine-tuning

We update all 251.7M denoiser parameters of a diffusion policy (Chi et al., 2023) with PPO (Schulman et al., 2017), keeping the visual encoder frozen. A short stochastic denoising diffusion implicit model (DDIM) chain (Song et al., 2021) generates each action chunk:

$$x_K \sim \mathcal{N}(0, I), \qquad x_{k-1} = \mu_\phi(x_k, t_k, c) + \sigma\varepsilon_k, \quad \varepsilon_k \sim \mathcal{N}(0, I). \tag{2}$$

Here c is the current observation condition and μ_ϕ is the DDIM update mean. We use $K = 5$ steps and $\sigma = .05$. PPO likelihoods are computed under the same Gaussian transitions used for sampling, following the denoising-MDP formulation (Ren et al., 2025).

We train five critic-only and five Product policies from a shared drawer-policy initialization, with independent training seeds. Each receives 126 episodes per iteration for 40 iterations, totaling 5,040 episodes. The shared warm-start policy has previously undergone RoboReward RL and Robometer-weighted offline regression on this scene. A second study compares Robometer and terminal completion rewards from a capable supervised policy in another drawer layout, with three independent trainings per objective (Appendix C.2).

Oracle controls optimize measured completion with either dense increments or terminal feedback; a verifier-only control isolates outcome supervision. These controls are evaluated on 128 seeds per policy (Appendix C). Update details and the reward-placement audit are in Appendix B.

4.2 Constrained Closed-loop Noise Policies

The constrained experiments test whether the pattern recurs across drawer manipulation and identity-sensitive stacking, using Robometer and LIV under PPO and grouped GRPO. Within each setting, reward objectives share a frozen diffusion decoder and a fixed bank of 24 noise sequences. A trainable categorical policy selects one sequence per episode, and the decoder generates actions from current observations throughout execution. The drawer selector learns unconditional probabilities; the stacking selector conditions on the initial visual observation.

Both optimizers update the selector probabilities from executed rollouts. Grouped GRPO (Shao et al., 2024) standardizes rewards within groups of four executions that share the initial state and images. Training starts from the uniform selector and uses eight rounds: 192 feedback episodes per objective for PPO and 256 for GRPO. Evaluation uses the final eighth checkpoint on 32 new seeds for drawer Robometer PPO and 64 per task/color for the other panels. Repetitions vary training and evaluation seeds on the same noise support; Appendix D gives replication and optimizer details.

4.3 Evaluation and Uncertainty

The primary full-denoiser comparison is iteration 40 versus the shared initialization on 512 pre-specified evaluation seeds. The same seed bank evaluates iterations 0, 10, 20, 30 and 40, providing descriptive training dynamics alongside the fixed endpoint. The shared initial policy is evaluated once as a common baseline; subsequent checkpoints are evaluated for all five training histories per objective. Paired policies use the same renderer and evaluation seeds within each panel; oracle controls are compared with their own initial evaluations.

We report changes in success and wrong-object failure in percentage points (pp), with 95% bootstrap confidence intervals (CIs). Per-training intervals resample paired evaluation seeds. Pooled full-

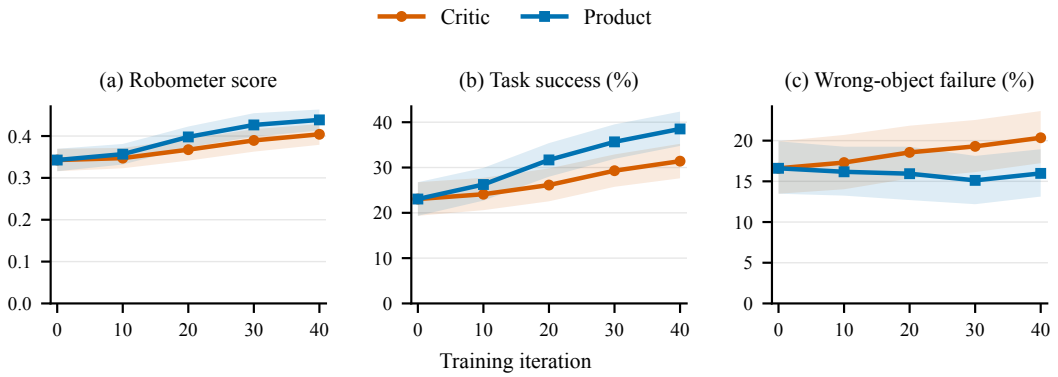

Figure 2: **Reward gain and wrong-object amplification develop together.** Lines show mean Robometer score, task success and wrong-object failure across five training histories per objective on the drawer task. All checkpoints and objectives use the same 512 evaluation seeds. Shading denotes 95% pointwise crossed-bootstrap CIs over training histories and shared seeds.

denoiser intervals jointly account for training-history and shared-seed variation; cross-objective contrasts resample histories independently across objectives while retaining the common seed draw. Constrained-policy intervals condition on the fitted policies and fixed noise supports. Appendix B.2 gives full-denoiser bootstrap details and per-training results, and Appendix F the constrained grid.

5 Optimization Amplifies Semantic Errors

5.1 Wrong-object Amplification with a Trainable Policy Network

Reward gain is accompanied by a specific error. Critic optimization improves task success while also increasing wrong-object failures (Table 1). Across the five training histories, success increases by 8.4 percentage points on average, while wrong-object failures increase by 3.8 points. All five histories have positive point estimates for both reward gain and wrong-object amplification. Paired executions show that both behavioral changes arise largely by converting former other failures into correct completions and wrong-object behaviors. The increase in distractor opening also persists when measured regardless of task success. Rising reward and rising success therefore do not certify that the semantic error has diminished. Appendix B.2 reports per-training changes (Figure 5) and behavior transitions.

Figure 2 shows that the critic's mean reward, success and wrong-object failure rates rise together across the five checkpoints. Product training follows a different trajectory, with larger success gains and mean wrong-object failure remaining near the initial level. Appendix B.3 shows each history.

The optimizer can improve the intended task. All three dense-oracle trainings improve success, as does the terminal-completion control, with paired intervals excluding zero. These controls show the policy family can learn the task, including from once-per-episode feedback (Appendix C).

Amplification from supervised initialization. Prior optimization against learned rewards is not required for amplification. In a second drawer layout selected by a fixed capability screen, three Robometer-trained policies from a shared supervised initialization again improve success while amplifying wrong-object failures. At the fixed endpoint, success rises by 9.6 percentage points and wrong-object failures by 11.8 points, alongside a higher critic reward. Terminal-completion training also improves task success but produces less wrong-object amplification. In an independent repetition, Robometer amplification recurs, but completion rewards also increase wrong-object failures, so the Robometer–completion contrast is not resolved there (Appendix C.2).

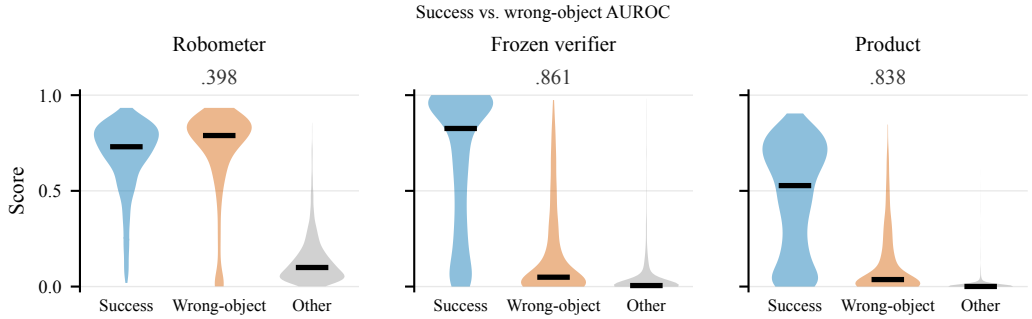

Figure 3: **A semantic inversion hidden by other failures.** Score distributions over 5,760 distinct trajectories from the checkpoint-diagnostic panel, grouped by physical outcome. Panel annotations report success–wrong-object AUROC. Bars mark medians; widths show normalized density.

5.2 Recurrence and Boundaries Across the Constrained Grid

Reward gain and wrong-object amplification recur across three PPO and two GRPO drawer trainings. In the first PPO training, wrong-object failures increase from 9 to 24 of 32 executions: the robot opens the distractor while leaving the requested drawer nearly closed. These high-scoring executions show task-inconsistent manipulation, not merely a failure to move.

Distinct-color LIV stacking reproduces reward gain, success loss and wrong-object amplification under both PPO and GRPO. Across the four stacking and five drawer histories, all nine establish semantic amplification and oracle success gains; seven additionally establish a binary-success loss. Figure 8 in Appendix D shows the recurring pattern with per-training counts and intervals.

Prevalence across task–critic pairings. Across the complete grid, 11 of 29 evaluated strata (from 20 training histories) show wrong-object amplification with a paired 95% interval above zero: every drawer Robometer and distinct-color LIV stacking stratum, and two near-color Robometer stacking strata. Elsewhere, near-color LIV stacking does not reliably induce the signature, Robometer stacking often learns the intended task, and drawer LIV can favor partial progress and closing (Appendix F.1).

5.3 Aggregate Reward Discrimination Can Hide Semantic Misranking

The aggregate AUROC and semantic AUROC ask different questions. Robometer separates success from failure overall (AUROC .782), yet reverses the ordering between success and wrong-object failure (AUROC .398; Figure 3). The pattern persists on the final policies alone (Appendix B.5).

AUROC is the probability that a random success outscores a random failure, with half credit for ties; a value below .5 indicates inverted ordering. With S, W and N the successful, wrong-object and other-failure trajectories of one panel, empirical AUROC decomposes exactly as

$$A_{S,F} = \lambda A_{S,W} + (1-\lambda) A_{S,N}, \qquad \lambda = \frac{|W|}{|W| + |N|}, \quad F = W \cup N. \tag{3}$$

Thus good discrimination against other failures can mask poor ordering against wrong-object failures. Here, other failures make up most negative examples and are separated well ($A_{S,N} = .970$). The aggregate score does not certify the ordering of the semantic subset that an optimizing policy can reach.

On 128 shared seeds per checkpoint, this panel contains the initial-policy episodes once and iterations 10–40 of each learned arm, so it measures reward ordering on behaviors produced by optimization.

Retrospective association. Using only initial-policy trajectories from the 29 constrained strata, semantic misranking tracks later amplification: the Spearman correlation between the initial success–wrong-object AUROC and ΔW is $-.82\,[-.91, -.68]$, versus $-.34\,[-.63, +.05]$ for aggregate AUROC (Appendix F.2). The association is carried by differences between task–critic families and colors and is absent within both drawer families.

6 Verification as a Diagnostic Intervention

A frozen, task-specific success verifier lets us test how outcome-supervised visual feedback redirects behavior. We compare the original critic reweighted by this verifier with the verifier alone:

$$R_P(\tau, x) = \bar{r}_b(\tau, x)\, p_\theta(\tau), \qquad R_V(\tau) = p_\theta(\tau), \tag{4}$$

where $\bar{r}_b \in [0, 1]$ is the same nonnegative critic objective used for C. Both rewards use the same frozen verifier, so their comparison measures the benefit of retaining the critic at equal supervision.

Task-specific visual verification. The verifier predicts simulator-labeled success from visual features near the end of a trajectory and their change from initialization. The drawer head is trained on 384 simulator-labeled development trajectories that include executions of critic-optimized development policies (Appendix A.1); stacking heads use 456 or 2,536 labels. Each head is frozen before policy optimization and reused across critics within its task and label budget.

The same optimization can learn a better behavior. Relative to critic-only training, the product increases success by 7.1 percentage points and reduces wrong-object failures by 4.4 points on the shared 512-seed evaluation (Table 1; per-training changes in Figure 5); its wrong-object change relative to initialization is $-0.6\,[-3.4, 2.1]$ points. A verifier-only control also substantially improves completion without changing its wrong-object count (Appendix C).

The constrained policies show larger recoveries across drawer and stacking experiments (Figure 8). Verifier-only rewards reproduce much of this recovery, identifying task-specific outcome supervision as a substantial source of the improvement (Appendix D.2).

Redirecting an already exploited policy. A separate continuation forks each of three exploited constrained drawer policies and its Adam state, without new verifier labels. Switching to the product improves task completion on all three histories and at a drawer opening held out from continuation, while continuing the original critic leaves success low. The third history remains only partially repaired, and verifier-only is better on its in-distribution evaluation (Appendix D.1).

The value of retaining the critic depends on the task and training setting (Appendix D.2). In a controlled ball-placement extension, the product with a full-history verifier supports near-complete recovery (63 and 64/64 successes in two histories), whereas a final-quarter verifier at equal label and frame budgets does not (17 and 2/64; Appendix D.2).

7 Sim-to-Real Case Study

As a sim-to-real case study, we deploy simulation-fine-tuned drawer policies on a Franka Research 3 (Franka Emika GmbH, 2023) in a setup matched to simulation at 1:1 scale (Appendix E). We

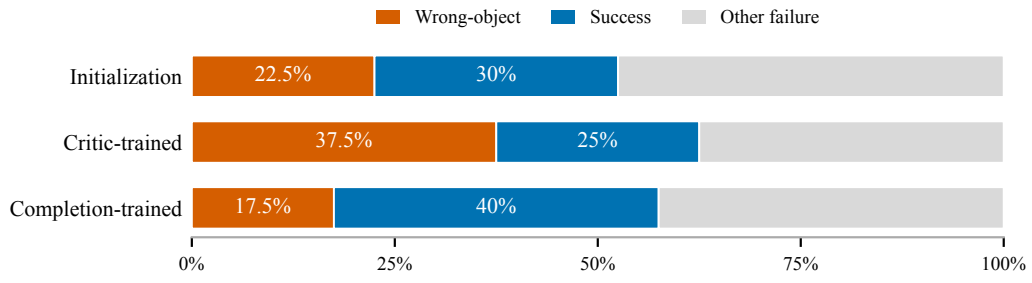

Figure 4: **Real-robot outcome composition after simulation fine-tuning.** Stacked bars show the percentage of 40 physical trials per policy ending in success, wrong-object failure or other failure. The wrong-object changes and the Completion-trained success gain match the direction seen in simulation, but Critic-trained success falls rather than rises; no difference from Initialization is statistically resolved at this sample size (intervals and Fisher tests in Table 9).

compare three policies, all deployed with the five-step DDIM sampler used in fine-tuning: the shared warm-start initialization, which has prior reward exposure (Initialization); Critic 5 at iteration 40, chosen after the simulation evaluations without a prespecified quantitative selection rule or any hardware outcomes and fixed before any physical trials (Critic-trained); and the terminal-completion oracle at iteration 40 (Completion-trained). Each policy executes 40 trials without hardware fine-tuning, with outcomes assessed independently of the critic. A trial succeeds if the middle drawer opens by at least 6 cm; it is a wrong-object failure if the upper drawer is pulled more than 2 cm beyond its initial half-open position while the task remains unsuccessful; all other trials are other failures.

Figure 4 summarizes the physical executions. Initialization succeeds in 30.0% of trials and fails on the wrong object in 22.5%; Critic-trained reaches 25.0% and 37.5%, and Completion-trained 40.0% and 17.5%. The wrong-object changes and the Completion-trained success gain point in the same direction as in simulation, but Critic-trained success is lower than Initialization, whereas critic training raises success in simulation. With 40 trials per policy, none of these differences is statistically resolved (Table 9 in Appendix E).

8 Discussion and Limitations

Useful task learning and semantic failure amplification can coexist. An improving average success rate can conceal a policy's increasing tendency to act on the wrong object. Evaluating a visual reward therefore requires tracking specific failure behaviors through optimization, using outcome measurements independent of the reward itself. The trajectory audit offers a corresponding evaluation target for reward models: discrimination between success and semantically coherent failures, alongside aggregate success–failure discrimination. Prospective validation of such audits on unseen task–critic pairings remains open.

Limitations. Task selection, shared representations and the small number of training histories (five per objective from the warm start, three from the supervised initialization) limit transfer to broader policy and task populations. Pooled full-denoiser intervals include training variation; constrained-policy intervals condition on fitted policies. In an independent repetition from the supervised initialization, completion rewards also increased wrong-object failures, leaving the Robometer–completion contrast unresolved (Appendix C.2). Supervised-initialization results are conditional on the capability-selected layout.

Verification and supervision. Outcome supervision improves completion, and verifier-only rewards reproduce much of the gain obtained with the product reward. The product reduces wrong-object failures relative to critic-only training, but does not establish a reduction relative to initialization. Because a wrong-object failure requires task failure, a lower W alone does not show fewer distractor actions. A zero critic score remains zero after multiplication. Verification should therefore be compared with its underlying supervision and evaluated on both completion and specific failures.

9 Conclusion

CriticHack shows that visual reward optimization can improve task success while amplifying wrong-object failures. Paired outcome analyses reveal how previously unsuccessful executions split into useful task completions and semantic errors, while reward audits show that strong aggregate discrimination can conceal a preference for wrong-object failures. Verification-based interventions identify task-specific outcome supervision as a substantial source of recovery. These findings motivate evaluating learned rewards through the behaviors they reinforce during policy optimization.

AI Assistance Statement

OpenAI Codex, ChatGPT, and Anthropic Claude were used to assist with code implementation, data analysis, figure generation, and manuscript refinement. All AI-generated outputs and analyses were rigorously verified by the authors against executable MuJoCo rollouts, stored policies, and simulator measurements. The authors remain solely responsible for the originality, validity, and integrity of all claims presented in this work.

Reproducibility Statement

Appendices A–D provide the primary evaluation protocol, all primary training endpoints, controls and uncertainty definitions. The separate Supplement reports the complete constrained grid and additional verifier analyses. The CPU artifact contains scored outcomes, frozen configurations, analysis scripts and provenance manifests; it does not include a complete GPU retraining environment. Model and raw-trajectory availability is specified in its artifact manifest. Hardware protocol details are given in Appendix E.

References

Minttu Alakuijala, Reginald McLean, Isaac Woungang, Nariman Farsad, Samuel Kaski, Pekka Marttinen, and Kai Yuan. Video-Language Critic: Transferable reward functions for language-conditioned robotics. *Transactions on Machine Learning Research*, 2025. URL https://openreview.net/forum?id=jJOVpnNrEp.

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*, 2016. URL https://arxiv.org/abs/1606.06565.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/14f75513f0f1ca01de1e826b52e6b840-Abstract-Conference.html.

Annie S. Chen, Suraj Nair, and Chelsea Finn. Learning generalizable robotic reward functions from "in-the-wild" human videos. In *Proceedings of Robotics: Science and Systems*, 2021. doi: 10.15607/RSS.2021.XVII.012. URL https://www.roboticsproceedings.org/rss17/p012.html.

Yuhui Chen, Shuai Tian, Shugao Liu, Yingting Zhou, Haoran Li, and Dongbin Zhao. ConRFT: A reinforced fine-tuning method for VLA models via consistency policy. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.019. URL https://roboticsproceedings.org/rss21/p019.html.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems*, 2023. doi: 10.15607/RSS.2023.XIX.026. URL https://www.roboticsproceedings.org/rss19/p026.html.

Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/d5e2c0adad503c91f91df240d0cd4e49-Paper.pdf.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. URL https://arxiv.org/abs/2110.14168.

Thomas Coste, Usman Anwar, Robert Kirk, and David Krueger. Reward model ensembles help mitigate overoptimization. In *International Conference on Learning Representations*, 2024. URL

https://proceedings.iclr.cc/paper_files/paper/2024/hash/dda7f9378a210c25e470e19304cce85d-Abstract-Conference.html.

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 *Proceedings of The 2nd Conference on Lifelong Learning Agents*, volume 232 of *Proceedings of Machine Learning Research*, pp. 120–136, 2023. URL https://proceedings.mlr.press/v232/du23b.html.

Lukas Fluri, Leon Lang, Alessandro Abate, Patrick Forré, David Krueger, and Joar Max Viktor Skalse. The perils of optimizing learned reward functions: Low training error does not guarantee low regret. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 17306–17377. PMLR, 2025. URL https://proceedings.mlr.press/v267/fluri25a.html.

Franka Emika GmbH. *Franka Research 3: Product Manual*, September 2023. URL https://download.franka.de/documents/120010_Product%20Manual%20Franka%20Research%203_1.0_EN.pdf. Document 120010, version 1.0.

Justin Fu, Avi Singh, Dibya Ghosh, Larry Yang, and Sergey Levine. Variational inverse control with events: A general framework for data-driven reward definition. In *Advances in Neural Information Processing Systems*, volume 31, 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/c9319967c038f9b923068dabdf60cfe3-Paper.pdf.

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pp. 10835–10866. PMLR, 2023. URL https://proceedings.mlr.press/v202/gao23h.html.

Amita Kamath, Jack Hessel, and Kai-Wei Chang. What's "up" with vision-language models? investigating their struggle with spatial reasoning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 9161–9175, 2023. doi: 10.18653/v1/2023.emnlp-main.568.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model. In *Proceedings of The 8th Conference on Robot Learning*, volume 270, pp. 2679–2713. PMLR, 2025a. URL https://proceedings.mlr.press/v270/kim25c.html.

Sunghwan Kim, Dongjin Kang, Taeyoon Kwon, Hyungjoo Chae, Dongha Lee, and Jinyoung Yeo. Rethinking reward model evaluation through the lens of reward overoptimization. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13252–13280. Association for Computational Linguistics, 2025b. doi: 10.18653/v1/2025.acl-long.649. URL https://aclanthology.org/2025.acl-long.649/.

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. URL https://arxiv.org/abs/2601.00675.

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 *Proceedings of Robotics: Science and Systems*, 2026. doi: 10.15607/RSS.2026.XXII.140. URL https://roboticsproceedings.org/rss22/p140.html.

Yecheng Jason Ma, Vikash Kumar, Amy Zhang, Osbert Bastani, and Dinesh Jayaraman. LIV: Language-image representations and rewards for robotic control. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pp. 23301–23320. PMLR, 2023a. URL https://proceedings.mlr.press/v202/ma23b.html.

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 *International Conference on Learning Representations*, 2023b. URL https://arxiv.org/abs/2210.00030.

Luke Oakden-Rayner, Jared Dunnmon, Gustavo Carneiro, and Christopher Ré. Hidden stratification causes clinically meaningful failures in machine learning for medical imaging. In *Proceedings of the ACM Conference on Health, Inference, and Learning*, pp. 151–159, 2020. doi: 10.1145/3368555.3384468.

Alexander Pan, Kush Bhatia, and Jacob Steinhardt. The effects of reward misspecification: Mapping and mitigating misaligned models. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2201.03544.

Allen Z. Ren, Justin Lidard, Lars L. Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion policy policy optimization. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/c0749c39aaff9e9e4c91f7118bf21b1e-Abstract-Conference.html.

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*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/7a7f6cc5dc2a84fb4edf0feb8e5cfd50-Abstract-Conference.html.

John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations*, 2016. URL https://arxiv.org/abs/1506.02438.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. URL https://arxiv.org/abs/1707.06347.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. URL https://arxiv.org/abs/2402.03300.

Joar Skalse, Nikolaus H. R. Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and characterizing reward gaming. In *Advances in Neural Information Processing Systems*, volume 35, pp. 9460–9471, 2022. doi: 10.52202/068431-0687. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/3d719fee332caa23d5038b8a90e81796-Abstract-Conference.html.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2010.02502.

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. In *Advances in Neural Information Processing Systems*, volume 36, pp. 55681–55693, 2023. doi: 10.52202/075280-2430. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/ae54ce310476218f26dd48c1626d5187-Paper-Conference.pdf.

Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, 2012. doi: 10.1109/IROS.2012.6386109. URL https://doi.org/10.1109/IROS.2012.6386109.

Andrew Wagenmaker, Yunchu Zhang, Mitsuhiko Nakamoto, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering your diffusion policy with latent space

reinforcement learning. In *Proceedings of The 9th Conference on Robot Learning*, volume 305, pp. 258–282. PMLR, 2025. URL https://proceedings.mlr.press/v305/wagenmaker25a.html.

Xueru Wen, Jie Lou, Yaojie Lu, Hongyu Lin, Xing Yu, Xinyu Lu, Ben He, Xianpei Han, Debing Zhang, and Le Sun. Rethinking reward model evaluation: Are we barking up the wrong tree? In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=Cnwz9jONi5.

Pengzhi Yang, Xinyu Wang, Pengyu Jing, Kehan Wen, Yiduo Qu, Zhenhao Huang, Minghao Fu, Xin Liu, Yaheng Shen, and Fan Shi. RARM: Confidence-gated progress reward modeling for RL in manipulation. *arXiv preprint arXiv:2606.22027*, 2026. URL https://arxiv.org/abs/2606.22027.

Yanting Yang, Minghao Chen, Qibo Qiu, Jiahao Wu, Wenxiao Wang, Binbin Lin, Ziyu Guan, and Xiaofei He. Adapt2Reward: Adapting video-language models to generalizable robotic rewards via failure prompts. In *Computer Vision – ECCV 2024*, volume 15115 of *Lecture Notes in Computer Science*, pp. 163–180. Springer, 2024. doi: 10.1007/978-3-031-72998-0_10. URL https://doi.org/10.1007/978-3-031-72998-0_10.

Mert Yuksekgonul, Federico Bianchi, Pratyusha Kalluri, Dan Jurafsky, and James Zou. When and why vision-language models behave like bags-of-words, and what to do about it? In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=KRLUvxh8uaX.

Ziyi Zhang, Sen Zhang, Yibing Zhan, Yong Luo, Yonggang Wen, and Dacheng Tao. Confronting reward overoptimization for diffusion models: A perspective of inductive and primacy biases. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 60396–60413. PMLR, 2024. URL https://proceedings.mlr.press/v235/zhang24ch.html.

A CLAIMS, MEASUREMENTS AND PROTOCOLS

Claim	Evidence and control	Scope
Useful learning and wrong-object amplification coexist	Five critic trainings, 512 shared seeds; Table 1. Oracle controls in Appendix C.	Full-denoiser optimization from a reward-exposed drawer policy under the fixed evaluation protocol.
Prior critic exposure is not required	Three trainings per objective from supervised initialization, 512 seeds; Appendix C.2.	A layout selected by a fixed capability screen.
Verification redirects behavior	Five product trainings versus critic, 512 shared seeds; verifier-only controls in Appendices C and D.2.	Product–critic contrasts under the fine-tuning sampler and matched verifier-only controls with task-specific outcome supervision.
The pattern recurs across protocols	Drawer Robometer and distinct-color LIV stacking under PPO and GRPO; Appendix D.	Fixed noise supports and task-specific verifiers; complete task–critic grid in Supplementary §S2.
Aggregate AUROC can mask semantic misranking	Collected-trajectory audit; Section 5.3 and Appendix B.5.	Descriptive reward ordering on collected trajectories from the evaluated policies.

Table 2: **Evidence supporting the main claims.** Controls and diagnostics answer different questions and are not pooled into the primary endpoint.

Primary estimate and additional analyses. The primary warm-start endpoint compares fixed iteration 40 with the shared initial policy on a fixed bank of 512 evaluation seeds. Appendix B.2 provides the complete training counts and crossed bootstrap definition. The separate Supplement contains the complete constrained grid, sampler calibration, independent training repetition and verifier analyses. The artifact provides scored outcomes, configurations, analysis code and provenance.

A.1 TASK-SPECIFIC VERIFICATION AND LABEL PROVENANCE

The two policy-camera embeddings supply 128 visual coordinates; ten robot-state coordinates are excluded. The 256-dimensional verifier input is the mean of the final five recorded visual observations concatenated with its difference from the first observation. This last recorded observation precedes the final action chunk. We fit feature normalization on the labeled development set and freeze it with the head. Simulator state supplies training labels and independent outcome measurements, but is not an input to the verifier or original critic.

The drawer logistic head uses 384 trajectories: a 96-episode static panel and complete policy-induced panels of 48, 48, 128 and 64 episodes, including 91 successes in total. The 96 static trajectories cover 12 scenes with eight executions each. The 288 policy-induced trajectories come from four development-stage panels collected before the reported trainings, containing executions of policies optimized against the original critic, the oracle and earlier product rewards, together with their development parent policy. No trajectory from the reported primary trainings or their evaluations enters the fit. Its objective is mean binary cross-entropy plus $.01\|w\|_2^2/2$, optimized for at most 200 L-BFGS iterations. Earlier 96- and 144-label heads are development controls; they are not substituted for this frozen head in the common matrix.

The cube architecture was selected on development data: 296 training labels and 96 validation labels compare nine fixed linear, MLP and RBF candidates by Brier score. The selected three-member, 64-unit ReLU MLP uses $L_2 = .001$ and is refit on all 392 labels. The deployed 456-label version adds 64 uniformly sampled labels from earlier policy-training executions, using a fixed sampling seed and no endpoint features. A separate diversity-selection diagnostic does not replace this head. The MLP fitting recipe uses 500 steps at learning rate .003. The 2,536-label version instead combines the 392 labels with all 2,016 training labels from two earlier stacking-development runs and 128 predetermined distinct-color examples from their old endpoints. Their original run IDs are listed in Supplementary Table S15. Those old endpoints become *development data*, not independent evaluation of the expanded head. Relative to 456 labels, the additional cost is 2,080 labels. The architecture is unchanged.

Each current run freezes the chosen head before new policy training and new-seed endpoint evaluation. No current evaluation trajectory enters head fitting. The same task-specific head is reused across original critics at a fixed budget. A new head is fitted for the other task, and both cube colors participate in training: this is not zero-shot transfer of one verifier across tasks or to unseen colors.

A.2 Execution, Rewards and Physical Measurements

Drawer executions retain 300 simulator steps and 75 observation-conditioned action chunks. Eight critic frames are sampled at steps 0, 43, 85, 128, 171, 214, 256 and 299. Success requires final target opening of at least 54 mm. Wrong-object failure requires task failure and more than 20 mm of additional upper-drawer opening. It excludes upper closing, so target opening and upper net motion are also reported. Claims that a failed trajectory never completed the task use its maximum target opening over the complete saved joint trace, not just a final image.

Cube evaluation uses two source scenes and both target/decoy position assignments, with 16 paired evaluation seeds per cell, totaling 64 per color and policy. Success requires the requested target to satisfy the stack geometry (6 mm position tolerance, 12 mm height tolerance and 20-degree tilt limit) and remain in contact with the support, without gripper contact, for the final three native control steps. The decoy is evaluated with the identical rule. Wrong-object failure means the target fails while the decoy succeeds; all remaining unsuccessful episodes are no-stack failures. The three categories partition every reported panel. Colors interpolate between a fixed distinct decoy and the target at fractions 0 and .9; both conditions receive balanced training feedback.

The critic camera is separate from the policy's two observation cameras. Robometer uses the frozen original success-prediction output on the task instruction and eight rollout frames; its separate progress output is not substituted. LIV uses the official model's final-frame image–language cosine c, with $(1 + c)/2$ as the fixed nonnegative training reward for both original and product objectives. Native cosine is reported without fitting a success threshold. High-score failures, $F_c = \Pr[y = 0, r_b \geq c]$, and their wrong-object subset are secondary diagnostics with all executions in the denominator. Robometer uses $c = .5$ and normalized discrete RoboReward uses $c = .75$; LIV has no high-score threshold. Scores from different critics are not pooled. Separate RoboReward protocols and their score rubric are given in Supplementary §S3.1.

B PRIMARY FULL-DENOISER EVIDENCE

B.1 FROZEN CONTRACT AND EXECUTION

The training contract was committed before launch: parent weights fixed by checksum, 22.4M-parameter encoder frozen, all 251,701,895 denoiser parameters trainable, K=5, η=0, σ=.05; 126 fresh episodes per iteration and 40 iterations; two value-only iterations; ten epochs per iteration at 1,512 decisions per optimizer step; clipping .2, critic learning rate 10^{-3}, value coefficient .5, gradient-norm limit 1, γ=.999, λ=.95; actor learning rate adaptive to Kullback–Leibler (KL) divergence from 10^{-6} with target .01 per denoising step. Oracle learnability requires at least two of three histories to improve success or completion with a confidence-interval lower bound above zero. The critic and product study uses a fixed bank of 512 seeds shared across objectives and iterations $0, 10, 20, 30, 40$. Five training histories per objective share one evaluated initialization, giving 41 distinct model checkpoints and 20,992 model–seed executions. Model identities, the seed bank and the descriptive analysis are fixed before evaluating the intermediate checkpoints, and all are scored before analysis. The primary comparison is iteration 40 versus initialization; intermediate evaluations do not select checkpoints or hyperparameters. A separate 128-seed panel supplies the collected-trajectory semantic audit in Section 5.3. Training and evaluation seeds are functions of iteration, worker and slot, so a restarted unit reproduces its episodes exactly.

Each arm runs resident workers that load the model once and wait for the learner to publish the next weights; the learner consumes an iteration only when every worker has written its completion marker, writes an atomic checkpoint, then publishes the next weights with a checksum index. A worker that crashes is restarted alone and redoes its own unit under the same weights and seeds; only a learner failure or an exhausted worker restart budget fails the arm. The primary critic and product evaluation uses the same OSMesa renderer, evaluator and scorer for every model; the initial policy is scored once and shared by all ten comparisons. Oracle controls use their own seed sets and the rendering backend of their training. Two further equivalence checks are recorded with the results: disabling per-step camera rendering leaves trajectories, actions, latents and saved frames bit-identical, and the resident scorer reproduces the offline scorer used for all earlier experiments to 0.0 on shared episodes. Numerically, recomputing $\log \pi_\theta$ under the sampling weights matches the recorded values to 1.07×10^{-4} on the first minibatch with TF32 disabled; each arm reports this check.

B.1.1 CREDIT ASSIGNMENT AND NUMERICAL DETAILS

The mean μ_ϕ in Equation 2 is the $\eta = 0$ DDIM update with its clean-sample estimate clipped to $[-1, 1]$. The policy executes the first four actions of each unnormalized output; 300 control steps give 75 observation-conditioned decisions. Sampler calibration is reported in Supplementary §S1.3.

For the *dense oracle* histories, the completion increment at decision h is $r_h^O = \Phi(d_{h+1}) - \Phi(d_h)$, where $\Phi(d) = \mathrm{clip}(d/d^\star, 0, 1)$ and $d^\star = 54\,\mathrm{mm}$. The undiscounted sum telescopes to terminal minus initial completion; the PPO objective uses discount $\gamma = .999$. The terminal oracle uses a single terminal completion reward.

The value network takes frozen encoder features and the decision index, including the policy's proprioceptive condition but no privileged object state. Generalized advantage estimation (GAE) (Schulman et al., 2016) uses $\gamma = .999$ and $\lambda = .95$; the decision advantage is standardized within an iteration and copied to all K denoising transitions of that decision, which are not discounted relative to one another. Each iteration collects 9,450 decisions and runs ten epochs with six optimizer steps of 1,512 decisions. The PPO clip is .2, value coefficient .5, and gradient-norm limit 1. The implementation sums coordinate log-densities and uses a KL-adaptive actor rate bounded by $[10^{-8}, 10^{-4}]$. TF32 is disabled for both sampling and training.

Implemented rewards. For a learned-reward arm the reward of episode e at decision k is $r_{e,k} = \mathbb{1}[k = 75]\, s_e$, where s_e is the frozen scorer's value for that episode: Robometer's last-frame success probability for critic-only, its product with the verifier probability for the product arm, and the verifier probability alone for verifier-only. The terminal oracle uses $r_{e,k} = \mathbb{1}[k = 75]\,\mathrm{completion}_e$ and the dense oracle the per-decision completion increment. No score is rescaled before GAE. An executable test of the deployed learner (byte-identical to the code that ran) confirms that the terminal reward of every learned key equals the scorer file and is unchanged when the simulator completion changes.

On the real training data of six learned trainings (critic and product trainings 3–5; 30,240 episodes), every iteration's scorer file covers exactly the executed episodes, the product equals Robometer times verifier with zero error, and the reward means logged by the learner equal the scorer file's means exactly.

B.2 ALL PRIMARY ENDPOINTS AND UNCERTAINTY

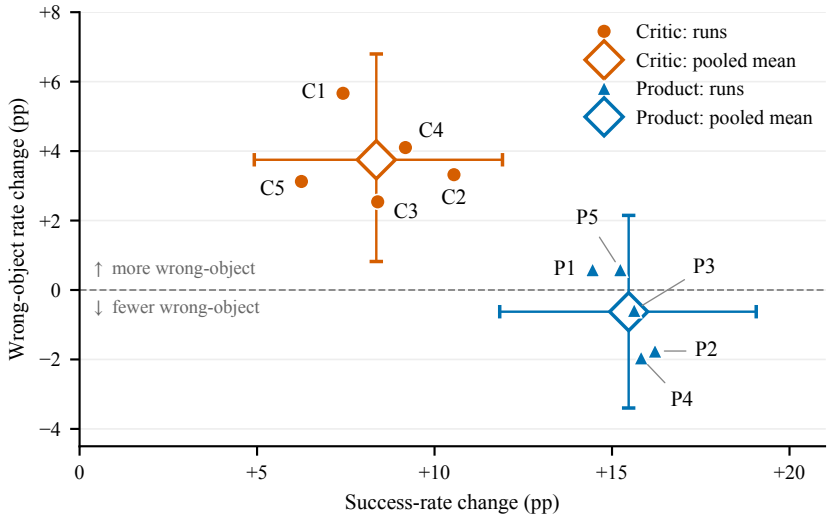

Figure 5: **Task improvement can coexist with wrong-object amplification.** Each small marker is one full-denoiser fine-tuning run on the occluded drawer task, placed by its change in task success and in wrong-object failure from the shared initialization to iteration 40, in percentage points on the same 512 evaluation seeds. A wrong-object failure opens the upper distractor drawer without completing the requested middle drawer. Critic runs (circles) train on the Robometer critic alone; Product runs (triangles) train on the critic multiplied by a frozen success verifier. Open diamonds mark the mean of the five runs of each reward. Their horizontal and vertical bars are marginal 95% intervals for the success and wrong-object changes respectively, resampling runs and evaluation seeds; together they do not form a joint confidence region.

Figure 5 places each critic and product training by its iteration-40 change in success and wrong-object failure; Figure 6 adds per-training intervals and Robometer changes. All five critic and all five product trainings enter the primary comparison. The initial policy is evaluated once on the 512 shared seeds. Per-training intervals resample paired evaluation seeds. Family estimates average the five per-training paired differences; their intervals resample training histories and one common evaluation-seed draw per replicate. The product–critic contrast draws training histories independently between objectives and retains the common evaluation-seed resample. All intervals use 20,000 bootstrap draws. Alternative resampling designs are reported separately in Supplementary §S1.2.

Three critic trainings individually establish both reward gain and wrong-object amplification; the other two have positive wrong-object estimates whose intervals include zero. Every product training establishes a success gain. No individual product training establishes non-increasing wrong-object failure relative to initialization. These per-training comparisons and the pooled product–critic contrast measure different effects.

Outcome composition. Over the 5×512 paired executions of the critic-only trainings, 302 other failures become successes and 189 become wrong-object failures, while 45 successes become wrong-object failures and 50 wrong-object failures become successes. For the product the corresponding counts are 422, 133, 19 and 83. Measured regardless of outcome, the share of executions that open the distractor by more than 20 mm rises by $6.6\,[3.3, 10.2]$ points under critic-only training. The product change, $+2.1\,[-1.2, +5.4]$ points, remains unresolved. Mean maximum distractor opening changes by $+3.96\,[2.06, 5.91]$ mm for critic and $+1.27\,[-0.58, +3.14]$ mm for product. Changes in unconditional distractor opening under the four oracle controls also have intervals containing zero.

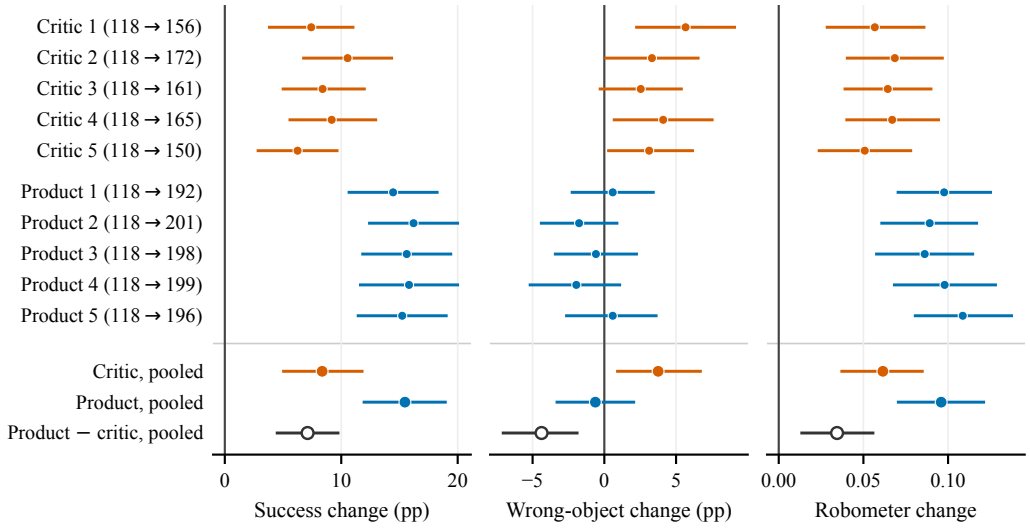

Figure 6: **All ten fine-tuned policies, each with its own intervals.** Each row compares one full-denoiser fine-tuning run at iteration 40 with the shared initialization on the same 512 occluded-drawer seeds; parentheses give its successes out of 512 at initialization and iteration 40. Panels show the changes in task success and wrong-object failure, in percentage points, and in Robometer score. A wrong-object failure opens the upper distractor drawer without completing the requested middle drawer. Critic runs train on the Robometer critic alone; Product runs train on the critic multiplied by a frozen success verifier. The last three rows pool the five runs of each reward and contrast the two. Bars are 95% intervals, over evaluation seeds for single runs and over runs and seeds for pooled rows.

B.3 Training dynamics on the fixed evaluation bank

Figure 7 reports all five histories per objective at the five fixed checkpoints. Main-paper Figure 2 averages those histories and shows 95% pointwise percentile intervals from 20,000 crossed-bootstrap draws. Each draw resamples five histories independently within each objective and one common set of 512 seed columns. The same history and seed draws apply across checkpoints, preserving their dependence. The common initialization has only evaluation-seed uncertainty. These intervals describe each checkpoint level, rather than simultaneous coverage or the endpoint contrasts in Table 1.

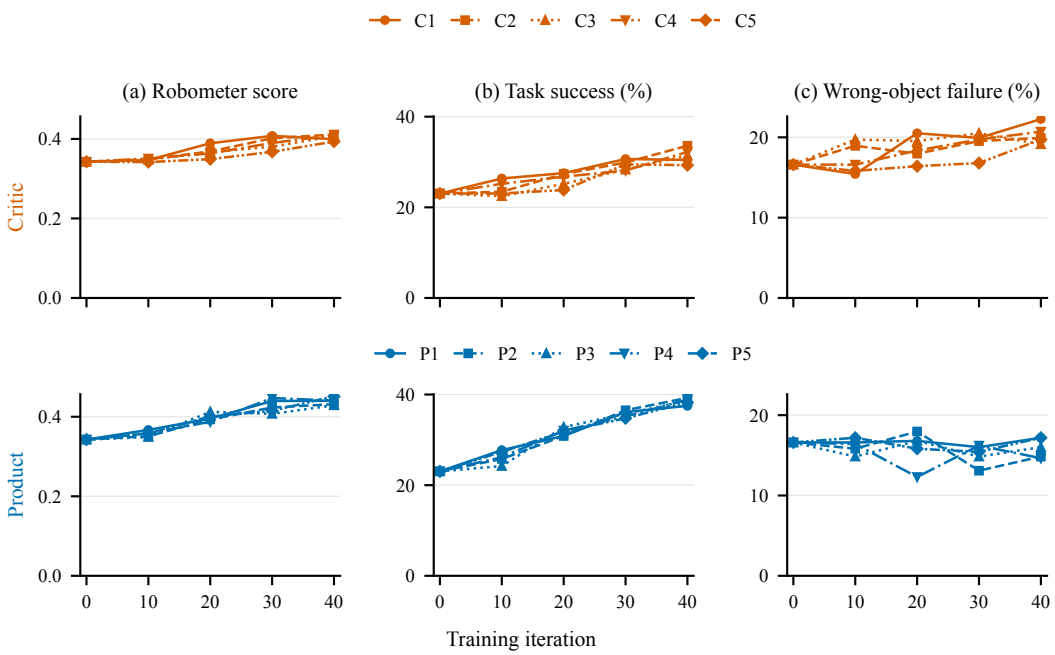

Figure 7: **Individual full-denoiser training histories.** Robometer score, task success and wrong-object failure on the same fixed 512-seed evaluation bank. C1–C5 are the five critic-only histories; P1–P5 use critic × frozen success verifier (Product). Each line averages one history's 512 executions at each checkpoint; all histories share the initial policy. Lines connect evaluated checkpoints without smoothing. The primary endpoint is iteration 40.

B.4 Policy, scorer and evaluation provenance

Initial policy. The shared initial weights are not a purely supervised policy. They descend from a diffusion policy trained by supervised learning on middle-drawer demonstrations, followed by three earlier updates in this project: (i) a five-epoch warm start on the supervised data mixed with copies of one optimized episode from an earlier critic study (5.3% of samples; that episode pushes the distractor drawer closed without opening the target, which is not the wrong-object pattern measured here); (ii) online RL against RoboReward-8B; and (iii) 600 steps of offline reward-weighted denoising regression toward Robometer on the evaluation scene, which in its own evaluation raised wrong-object failures from 1 to 7 of 32. The whole-network study therefore starts from a policy already exposed to the critic it optimizes. Every reported change is measured against this common initialization, and the constrained-selector experiments use the same decoder.

Scorer. Robometer-4B at a fixed revision scores eight evenly spaced frames of a third-person camera with the task text "open the middle drawer"; the reward is the success probability at the last frame, and the base vision-language model is loaded offline at a fixed revision. The verifier is the frozen model of Section 6. Rescoring 32 stored episodes on a second installation reproduced the stored Robometer scores, all eight per-frame success probabilities and the verifier outputs exactly (maximum absolute difference 0).

All primary critic and product evaluations use the same OSMesa renderer, evaluator and scorer. Weights are fixed at iteration 40; no intermediate checkpoint is selected by its evaluation outcome. The training sampler is stochastic DDIM with five steps and standard deviation .05. The artifact records model identities, evaluation seeds, configurations and score files.

B.5 COLLECTED-TRAJECTORY SEMANTIC AUDIT

The reward audit summarized in Section 5.3 rescores episodes that were already produced and scored, so it adds no rollouts. The separate checkpoint-diagnostic panel contains every distinct episode of the learned arms: the shared initial-policy episodes once (every arm reproduces them value for value) and iterations 10–40 of each of the eleven learned-reward trainings, 5,760 episodes with 2,017 successes, 1,232 wrong-object failures and 2,511 other failures. Robometer's mean score is .702 on wrong-object failures against .678 on successes; 1,117 of the 3,743 failures score at or above .5, 1,059 of them wrong-object, and 12 of its 16 highest-scored episodes are wrong-object failures. Under the verifier 211 failures clear that threshold and under the product 75. Restricting the panel to the eleven final policies (1,408 episodes) gives AUROC against wrong-object failures of .389 for Robometer, .887 for the verifier and .859 for the product, and the 16 highest-scored episodes are all successes under both the verifier and the product (Table 3).

	Robometer			Frozen verifier			Product		
Ranked set	Success	Wrong-object	Other	Success	Wrong-object	Other	Success	Wrong-object	Other
All distinct trajectories ($n = 5,760$)									
Top 16	4	12	0	16	0	0	16	0	0
Top 32	9	23	0	32	0	0	32	0	0
Top 64	17	47	0	64	0	0	64	0	0
Final policies only ($n = 1,408$)									
Top 16	7	9	0	16	0	0	16	0	0
Top 32	11	21	0	32	0	0	32	0	0
Top 64	25	39	0	64	0	0	62	2	0

Table 3: **Physical outcomes among the highest-scored trajectories.** Each row counts successes, wrong-object failures and other failures among the top k trajectories ranked separately by each scoring rule; each group of three columns sums to k. The upper panel uses all 5,760 distinct trajectories in the collected-trajectory audit of Figure 3; the lower panel uses the 1,408 trajectories from the eleven final policies. Product multiplies Robometer by the frozen verifier output.

C Learnability and Supervised-initialization Controls

C.1 Oracle learnability and verifier-only feedback

The dense oracle optimizes measured target-completion increments. Three independent histories improve success from 31 to 45, 20 to 55, and 33 to 54 of 128 executions, respectively. Each paired success interval excludes zero. The terminal-completion control improves from 18 to 51 successes under once-per-episode feedback. Together these controls show that the policy family can learn the task under the implemented update, including with a terminal reward.

	Count, initialization → 40		Change from initialization		
Policy	Success	Wrong-object	Success (pp)	Wrong-object (pp)	Robometer
Verifier only	32 → 64	21 → 21	+25.0 [+15.6, +34.4]	+0.0 [−7.0, +7.0]	+.126 [+.062, +.189]
Oracle A	31 → 45	22 → 21	+10.9 [+3.1, +19.5]	−0.8 [−7.8, +6.3]	not scored
Oracle B	20 → 55	21 → 14	+27.3 [+19.5, +35.2]	−5.6 [−11.9, +0.8]	not scored
Oracle C	33 → 54	18 → 12	+16.4 [+8.6, +24.2]	−4.7 [−10.9, +1.6]	not scored
Terminal oracle	18 → 51	23 → 13	+25.8 [+18.0, +34.4]	−7.8 [−14.8, −1.6]	not scored

Table 4: **Learnability and verifier-only controls.** Counts and changes compare iteration 40 with initialization using paired evaluation seeds for each training. Brackets give 95% confidence intervals from resampling those seeds. Oracle rewards use simulator-measured completion; oracle policies were not scored by Robometer.

The controls have separate evaluation seeds and rendering assignments: oracle A and the terminal oracle use EGL; oracles B–C and verifier-only use OSMesa. Each is compared with the initial policy under its own setting. All oracle wrong-object point estimates are nonpositive. Changes in unconditional distractor opening remain unresolved. Sampler calibration and worker-reset details are in Supplementary §S1.3.

The single verifier-only training improves success from 32 to 64/128, with the same wrong-object count (21). Its paired success change is $+25.0\,[15.6, 34.4]$ percentage points. The frozen verifier therefore supplies a useful completion signal.

C.2 AMPLIFICATION FROM A CAPABLE SUPERVISED INITIALIZATION

Policy and task. We use a supervised diffusion policy further trained for ten epochs on 167 physically validated drawer demonstrations, with its own visual encoder and normalizer. No learned reward or reinforcement learning enters this policy's training lineage. Before RL, a fixed capability screen evaluates this checkpoint on twelve drawer layouts with eight seeds each. Layouts with 2–6 successes are eligible; we select the most successful, breaking ties by manifest order. The selected layout has 3/8 successes. It differs from the warm-start study's layout and from the ceiling-limited control in Supplementary §S3.4.

Training and evaluation. Three independent trainings per objective optimize either Robometer or terminal target completion, with the same supervised initialization, full-denoiser PPO update, 40 iterations and 126 feedback episodes per iteration. Completion is the final target opening divided by 54 mm and clipped to $[0, 1]$; wrong-object failure requires an unsuccessful task and more than 20 mm additional distractor opening. The endpoint is fixed at iteration 40. We evaluate all six final policies and the common initialization on the same 512 held-out seeds (Table 5).

Reward	Δ success (pp)	Δ wrong-object (pp)
Robometer	$+9.6$ [6.2, 13.0]	$+11.8$ [8.1, 15.6]
Completion	$+14.1$ [9.2, 19.3]	$+1.2$ [-2.6, 5.1]

Table 5: **Capable supervised initialization: 512-seed evaluation.** Three trainings per objective, each evaluated at fixed iteration 40 against the common initialization. Changes use 95% crossed training-by-evaluation-seed intervals.

Pooled 95% intervals use 20,000 crossed bootstrap draws over trainings and one shared set of evaluation seeds. Cross-objective contrasts resample the training histories independently across objectives and share the evaluation seed resample. On the 512-seed bank, Robometer minus completion rewards gives a wrong-object change of $+10.7\,[6.2, 15.0]$ percentage points and a success change of $-4.5\,[-9.6, .3]$ points. The latter difference is unresolved.

Training	Success	Wrong-object
Robometer		
Training 1	88 → 136	72 → 141
Training 2	88 → 138	72 → 124
Training 3	88 → 137	72 → 133
Completion		
Training 1	88 → 178	72 → 92
Training 2	88 → 148	72 → 72
Training 3	88 → 154	72 → 70

Table 6: **All six supervised-initialization trainings.** Counts at initialization and iteration 40, each out of 512 seeds. The initialization is evaluated once and shared across objectives and trainings. Wrong-object failure uses the same task-failure and 20 mm distractor-opening rule as the primary warm-start study.

An independent six-training repetition from the same supervised initialization, on its own 128-seed bank, reproduces Robometer wrong-object amplification ($+16.4\,[9.4, 23.7]$ points), but completion rewards also increase wrong-object failures ($+10.7\,[2.6, 18.8]$). The Robometer-minus-completion wrong-object contrast, $+5.7\,[-2.1, 13.5]$ points, is unresolved, and the success contrast, $-8.6\,[-16.9, -.8]$ points, favors completion. The complete repetition is reported in Supplementary §S3.5.

D CONSTRAINED-POLICY RECURRENCE AND MATCHED VERIFICATION

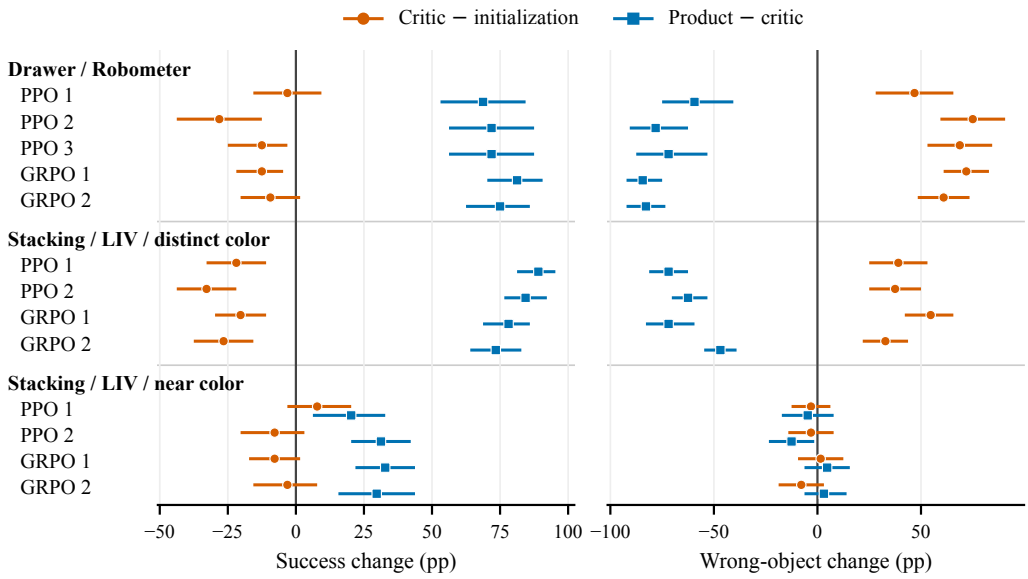

Figure 8: **Semantic amplification and the verification intervention.** Critic minus initialization (orange circles) and product minus critic (blue squares), with 95% paired execution intervals in percentage points. Rows are grouped by task, critic and color condition. These 13 strata contain all nine repeated-error trainings and both trained stacking colors; the near-color strata are retained even when induction is unresolved. All 29 strata, including Robometer stacking and drawer LIV boundaries, appear in Supplementary Figure S3. Training numbers identify histories within each task–critic–optimizer group; the two stacking colors share histories. Original IDs and complete counts are in Supplementary Tables S15 and S1.

Figure 8 contains the repeated drawer Robometer and LIV stacking comparisons, including both trained stacking colors. The tables below report all nine repeated-error histories across these settings. The complete 29-stratum grid, all supervision budgets and boundary settings appear in Supplementary §S2. Training numbers are local to each task–critic–optimizer group. Original run IDs are retained in Supplementary Table S15.

D.1 Oracle, Exploitation and Repair

Tables 7 and 8 report original critic scores, physical task successes and paired changes for all nine histories. All nine oracle policies improve true success with intervals above zero. For LIV stacking, their native-score changes are negative in all four trainings, also with intervals excluding zero. The oracle learns the physical task despite this critic disagreement. Conversely, original-LIV optimization raises native score while losing success in all four.

Training	n	Initialization	Oracle	Critic	Product
(a) Original critic score					
Drawer / Robometer					
PPO 1	32	0.4050	0.5517	0.7141	0.6594
PPO 2	32	0.3903	0.4817	0.8102	0.6249
PPO 3	32	0.3393	0.4273	0.7709	0.6456
GRPO 1	64	0.3008	0.6152	0.7937	0.6314
GRPO 2	64	0.3609	0.5946	0.7803	0.6120
Stacking / LIV / distinct color					
PPO 1	64	−0.1700	−0.1784	−0.1631	−0.1784
PPO 2	64	−0.1710	−0.1762	−0.1625	−0.1764
GRPO 1	64	−0.1693	−0.1757	−0.1627	−0.1750
GRPO 2	64	−0.1707	−0.1765	−0.1637	−0.1754
(b) Task successes (count)					
Drawer / Robometer					
PPO 1	32	4	24	3	25
PPO 2	32	11	19	2	25
PPO 3	32	6	18	2	25
GRPO 1	64	12	54	4	56
GRPO 2	64	10	58	4	52
Stacking / LIV / distinct color					
PPO 1	64	20	63	6	63
PPO 2	64	24	57	3	57
GRPO 1	64	15	55	2	52
GRPO 2	64	22	51	5	52
(c) Wrong-object failures (count)					
Drawer / Robometer					
PPO 1	32	9	2	24	5
PPO 2	32	4	3	28	3
PPO 3	32	6	4	28	5
GRPO 1	64	8	1	54	0
GRPO 2	64	14	0	53	0
Stacking / LIV / distinct color					
PPO 1	64	21	0	46	0
PPO 2	64	18	2	42	2
GRPO 1	64	18	6	53	7
GRPO 2	64	15	9	36	6

Table 7: **Oracle, exploitation and repair across nine training histories.** Panel (a) reports mean original critic scores: Robometer success probability for drawer and native LIV cosine for stacking. Panels (b) and (c) report physical task successes and wrong-object failures out of n evaluation executions per policy. Initialization is the uniform noise-selection policy; Oracle, Critic and Product optimize physical completion, the visual critic, and critic $\times$ frozen verifier, respectively. Oracle policies are evaluated with the same visual critic as the other policies. Stacking uses the 2,536-label verifier. Original run IDs are mapped in Supplementary Table S15.

Drawer PPO 2 and 3 and GRPO 1 also meet the strict reward-up, success-down, repaired-success-up criterion. Drawer PPO 1 and GRPO 2 have success-loss intervals crossing zero, but reliably increase wrong-object failures while increasing reward. Drawer PPO 1 additionally reduces mean clipped target completion from .2576 to .1082, a paired change of $-.1494\,[-.2917, -.0137]$. The oracle and product reach .9146 and .8300.

Training	n	Reward change	Task-success change (count of n)		
		Critic − Init.	Oracle − Init.	Critic − Init.	Product − Critic
Drawer / Robometer					
PPO 1	32	+.3091 [+.1915, +.4252]	+20 [+14, +26]	−1 [−5, +3]	+22 [+17, +27]
PPO 2	32	+.4199 [+.3139, +.5218]	+8 [+3, +14]	−9 [−14, −4]	+23 [+18, +28]
PPO 3	32	+.4316 [+.3184, +.5410]	+12 [+6, +18]	−4 [−8, −1]	+23 [+18, +28]
GRPO 1	64	+.4929 [+.4252, +.5601]	+42 [+34, +50]	−8 [−14, −3]	+52 [+45, +58]
GRPO 2	64	+.4195 [+.3398, +.4970]	+48 [+41, +54]	−6 [−13, +1]	+48 [+40, +55]
Stacking / LIV / distinct color					
PPO 1	64	+.0070 [+.0047, +.0093]	+43 [+36, +50]	−14 [−21, −7]	+57 [+52, +61]
PPO 2	64	+.0085 [+.0065, +.0105]	+33 [+25, +41]	−21 [−28, −14]	+54 [+49, +59]
GRPO 1	64	+.0067 [+.0047, +.0086]	+40 [+32, +48]	−13 [−19, −7]	+50 [+44, +55]
GRPO 2	64	+.0070 [+.0051, +.0091]	+29 [+22, +36]	−17 [−24, −10]	+47 [+41, +53]

Table 8: **Oracle learnability, reward exploitation and repair: paired changes.** Each row is one training history; entries are paired differences with 95% intervals in brackets. Init. is the uniform noise-selection policy. The reward change is the change in mean original critic score under critic-only training: Robometer success probability for drawer and native LIV cosine for stacking. Success changes are counts out of n evaluation executions. Seven of nine trainings have a success-loss interval strictly below zero; drawer PPO 1 and GRPO 2 establish amplified wrong-object behavior but not a reliable binary-success loss. Native reward and repair success increase in all nine.

Continuation from the three exploited drawer policies, with product, original-critic, verifier-only and oracle rewards and a drawer opening held out from continuation, is reported in Supplementary §S4.5.

D.2 VERIFIER-ONLY ABLATION

The ablation removes the original critic and optimizes $R_V = p_\theta$ with the identical verifier, labels, policy capacity and update protocol. Supplemental same-head controls retain the original seeds and repeat uniform executions to check reference reuse. These paired ablations reuse the original training histories. Supplementary Table S2 retains every available matched control and marks missing controls explicitly.

Source of the hacking repairs. For drawer Robometer GRPO, verifier-only reaches 55 and 53/64 successes with zero wrong-object errors, compared with product outcomes 56/0 and 52/0. Product-minus-only success differences are $+1[0, 3]/64$ and $-1[-3, 0]/64$. For distinct-color LIV stacking, expanded-only, the verifier-only control with the 2,536-label head, reaches 63/57 successes under PPO and 50/55 under GRPO; the product reaches 63/57 and 52/52. Neither optimizer establishes an additional physical benefit from retaining LIV. In the second PPO training, product and only have identical outcomes on every paired seed. These controls attribute much of the repair to outcome verification.

Fixed support and replication. The repeated trainings vary training and evaluation seeds on fixed 24-component development supports; they do not resample support banks. The drawer support was selected during development without an outcome-blind bank-selection rule. Both stacking colors participate in training. Intervals condition on the fitted policies and selected support. Appendix A.1 specifies the label costs and shared heads; Supplementary §S2.1 gives the exact selector representation, objectives and optimizer settings.

Temporal verification in ball placement. The controlled ball-placement extension compares full-history and final-quarter verifiers in a matched PPO ablation at equal label and frame budgets. In two independent histories, the product with the full-history verifier reaches 63 and 64/64 successes on new seeds, against 12/64 for the uniform selector and 2 and 5/64 for the original reward; the same product with a final-quarter verifier reaches 17 and 2/64. Its protocol and new-seed endpoints are in Supplementary §S4.4 and Supplementary Table S13.

E SIM-TO-REAL CASE STUDY: PROTOCOL AND ADDITIONAL RESULTS

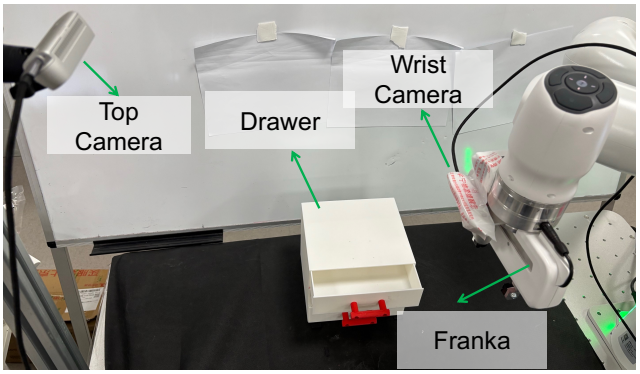

Figure 9: Real-robot deployment

The main hardware outcomes are reported in Section 7. Here, we provide the physical setup, deployed policy identities, and outcome definitions for the real-robot drawer evaluation. Fig. 9 shows the environment setup for deployment on a Franka robot.

Physical setup and reset procedure. Experiments use a Franka Research 3 robot in a physical drawer setup matched to the simulation at 1:1 scale, including cabinet placement, drawer geometry, initial drawer openings, and camera placement. The requested task is to open the middle drawer. Before each trial, the target middle drawer is reset to its closed position, and the upper distractor drawer is returned to the same marked partial opening. All policies are evaluated under the same physical setup and deployment configuration.

Policy identities and checkpoint mapping. The hardware comparison includes three conditions: *Initialization*, *Critic-trained* and *Completion-trained*. Initialization is the shared warm-start initialization of the full-denoiser study (Appendix B.4), which has prior reward exposure. Critic-trained is Critic 5 at iteration 40, chosen after the simulation evaluations without a prespecified quantitative selection rule; the checkpoint was fixed before any physical trials. No hardware outcomes were used in the choice. Among the five critic trainings, Critic 5 has the second-largest wrong-object increase on the 128-seed checkpoint-diagnostic panel of Section 5.3 (per-training values in Supplementary §S1) and the smallest success gain on the 512-seed primary panel (Figure 6). Completion-trained is the terminal-completion oracle of Appendix C.1 at iteration 40, trained from the same initialization. All policies are deployed with the five-step DDIM sampler used during fine-tuning. Because only one training history per fine-tuned condition was deployed, the hardware study is a sim-to-real case study rather than an estimate of transfer across training histories.

Deployment and evaluation. Each policy executes 40 trials, giving 120 trials across the three conditions. Policy parameters remain fixed throughout hardware evaluation, with no training or fine-tuning on the physical robot. The evaluation therefore examines the physical execution of policies trained in simulation. Task outcomes are assessed from physical drawer motion independently of the learned visual reward.

Outcome definitions. Each trial is assigned to one of three mutually exclusive categories. A *success* requires opening the target middle drawer by at least 6 cm. A *wrong-object failure* occurs when the upper distractor drawer is pulled more than 2 cm beyond its initial marked opening while the requested task remains unsuccessful. All remaining unsuccessful trials are classified as *other failures*. Grasping a drawer handle alone does not satisfy the success criterion.

Table 9 gives the outcome rates and their changes from Initialization for the hardware comparison in Section 7.

	Outcome rate (%)			Change from Initialization (pp)	
Policy	Success	Wrong-object	Other	Success	Wrong-object
Initialization	30.0 [18.1, 45.4]	22.5 [12.3, 37.5]	47.5 [32.9, 62.5]	–	–
Critic-trained	25.0 [14.2, 40.2]	37.5 [24.2, 53.0]	37.5 [24.2, 53.0]	−5.0 [−23.8, +14.3]	+15.0 [−5.0, +33.5]
Completion-trained	40.0 [26.3, 55.4]	17.5 [8.7, 31.9]	42.5 [28.5, 57.8]	+10.0 [−10.6, +29.5]	−5.0 [−22.4, +12.7]

Table 9: **Real-robot outcomes.** Each policy executes 40 physical drawer trials. Rates are percentages of trials with Wilson 95% intervals; changes relative to Initialization are in percentage points with Newcombe 95% intervals. No change is statistically resolved (two-sided Fisher exact $p = .80$ and .22 for Critic-trained success and wrong-object, .48 and .78 for Completion-trained). Success opens the requested middle drawer by at least 6 cm; a wrong-object failure pulls the upper distractor more than 2 cm beyond its initial opening without success. Initialization is the shared warm-start initialization of the full-denoiser study; Critic-trained is Critic 5 at iteration 40; Completion-trained is the terminal-completion oracle of Appendix C.1 at iteration 40.

F Scope and Retrospective Analyses

F.1 Prevalence across the constrained grid

A constrained-grid stratum shows wrong-object amplification if and only if the paired 95% interval for the critic-minus-uniform change in wrong-object failure has a lower bound above zero. Under this definition, 11 of the 29 evaluated strata (37.9%) show amplification (Table 10). The 29 strata come from 20 training histories, because the two stacking colors of one training share its history; they are not 29 independent trainings. In all 11 amplified strata, the critic-minus-uniform change in original reward also has an interval lower bound above zero. One drawer LIV stratum has a wrong-object interval whose lower bound is exactly zero and is not counted. In the two amplified near-color Robometer stacking strata, amplification occurs together with a positive success change. The nine repeated-error histories of Section 5.2 come from the two settings with amplification in every stratum and do not estimate prevalence. Table 11 lists every stratum with its intervals; Supplementary §S2 adds per-policy outcome counts and all supervision budgets, and separate RoboReward experiments outside the grid are in Supplementary §S3.

Task	Critic, decoy color	Amplified strata
Drawer	Robometer	5/5
Stacking	LIV, distinct color	4/4
Stacking	LIV, near color	0/4
Stacking	Robometer, near color	2/5
Stacking	Robometer, distinct color	0/5
Drawer	LIV	0/6
All		11/29

Table 10: **Wrong-object amplification across the constrained grid.** A stratum counts as amplified when the paired 95% interval for the critic-minus-uniform change in wrong-object failure lies above zero. The 29 strata come from 20 training histories; the two stacking colors of a training share its history.

F.2 Retrospective association with initial-policy misranking

For each of the 29 strata, two AUROCs are computed from initial-policy trajectories alone, in the notation of Equation 3: the aggregate success–failure AUROC $A_{S,F}$ and the semantic success–wrong-object AUROC $A_{S,W}$. Each is compared with the later critic-minus-uniform change in wrong-object failure, ΔW. The Spearman correlation of $A_{S,W}$ with ΔW is $-.82\,[-.91, -.68]$, against $-.34\,[-.63, +.05]$ for $A_{S,F}$; their difference is $-.48\,[-.80, -.20]$. Intervals resample the 20 training histories. Computing each AUROC and ΔW on disjoint halves of the initial-policy panel gives $-.74$ for $A_{S,W}$ and $-.31$ for $A_{S,F}$. The association is carried by differences between task–critic families and between decoy colors: it is absent within both drawer families, and four near-color LIV strata have $A_{S,W} < .5$ without amplification. It is therefore a retrospective description of these settings, not a validated prospective predictor. Metric definitions, the disjoint-halves procedure and the complete correlations are in Supplementary §S2.5 and Supplementary Table S3.

Training	Decoy	n	Change in count of n		Critic score	Amplified
			Wrong-object	Success		
Drawer / Robometer						
PPO 1	–	32	$+15$ [+9, +21]	-1 [−5, +3]	$+.3091$ [+.1915, +.4252]	✓
PPO 2	–	32	$+24$ [+19, +29]	-9 [−14, −4]	$+.4199$ [+.3139, +.5218]	✓
PPO 3	–	32	$+22$ [+17, +27]	-4 [−8, −1]	$+.4316$ [+.3184, +.5410]	✓
GRPO 1	–	64	$+46$ [+39, +53]	-8 [−14, −3]	$+.4929$ [+.4252, +.5601]	✓
GRPO 2	–	64	$+39$ [+31, +47]	-6 [−13, +1]	$+.4195$ [+.3398, +.4970]	✓
Stacking / LIV						
PPO 1	near	64	-2 [−8, +4]	$+5$ [−2, +13]	$+.0007$ [−.0006, +.0021]	
PPO 1	distinct	64	$+25$ [+16, +34]	-14 [−21, −7]	$+.0070$ [+.0047, +.0093]	✓
PPO 2	near	64	-2 [−9, +5]	-5 [−13, +2]	$+.0047$ [+.0031, +.0064]	
PPO 2	distinct	64	$+24$ [+16, +32]	-21 [−28, −14]	$+.0085$ [+.0065, +.0105]	✓
GRPO 1	near	64	$+1$ [−6, +8]	-5 [−11, +1]	$+.0012$ [−.0002, +.0026]	
GRPO 1	distinct	64	$+35$ [+27, +42]	-13 [−19, −7]	$+.0067$ [+.0047, +.0086]	✓
GRPO 2	near	64	-5 [−12, +2]	-2 [−10, +5]	$+.0025$ [+.0009, +.0042]	
GRPO 2	distinct	64	$+21$ [+14, +28]	-17 [−24, −10]	$+.0070$ [+.0051, +.0091]	✓
Stacking / Robometer						
PPO 1	near	64	$+13$ [+5, +21]	$+6$ [−2, +14]	$+.1770$ [+.1047, +.2491]	✓
PPO 1	distinct	64	-16 [−22, −10]	$+40$ [+33, +47]	$+.4278$ [+.3518, +.5030]	
PPO 2	near	64	$+2$ [−6, +10]	$+10$ [+3, +17]	$+.1361$ [+.0725, +.2002]	
PPO 2	distinct	64	-19 [−26, −12]	$+34$ [+27, +41]	$+.3450$ [+.2654, +.4233]	
PPO 3	near	64	$+13$ [+5, +21]	$+6$ [−1, +13]	$+.1448$ [+.0828, +.2109]	✓
PPO 3	distinct	64	-12 [−19, −6]	$+35$ [+28, +43]	$+.4290$ [+.3485, +.5098]	
GRPO 1	near	64	$+7$ [−1, +15]	$+12$ [+5, +19]	$+.1674$ [+.1040, +.2322]	
GRPO 1	distinct	64	-19 [−26, −12]	$+41$ [+33, +48]	$+.4345$ [+.3538, +.5102]	
GRPO 2	near	64	$+2$ [−5, +9]	$+19$ [+13, +25]	$+.2655$ [+.2036, +.3295]	
GRPO 2	distinct	64	-4 [−11, +3]	$+31$ [+25, +37]	$+.3914$ [+.3178, +.4640]	
Drawer / LIV						
PPO 1	–	64	-1 [−5, +3]	$+3$ [−1, +7]	$+.0023$ [+.0002, +.0049]	
PPO 2	–	64	$+2$ [+0, +5]	-1 [−3, +0]	$-.0001$ [−.0008, +.0006]	
PPO 3	–	64	-10 [−16, −4]	-7 [−13, −1]	$+.0222$ [+.0194, +.0250]	
PPO 4	–	64	-5 [−10, −1]	$+1$ [−9, +11]	$+.0220$ [+.0193, +.0245]	
GRPO 1	–	64	-11 [−16, −7]	-4 [−12, +3]	$+.0219$ [+.0192, +.0245]	
GRPO 2	–	64	-7 [−12, −3]	$+2$ [−6, +11]	$+.0219$ [+.0194, +.0244]	

Table 11: **All 29 strata of the constrained grid.** Each row is one training history evaluated on one task and critic and, for stacking, one decoy color; the two stacking colors of a training share its history. Entries are critic-minus-uniform changes after optimization with paired 95% intervals: wrong-object failures and successes as counts out of n evaluation executions, and the original critic score in its native units (Robometer success probability; LIV cosine). A check marks the 11 strata whose wrong-object interval lies above zero.