NOT EVERY STEP IS PROGRESS: EVENT-GROUNDED CREDIT REASSIGNMENT FOR ROBOT POLICY OPTIMIZATION

Anonymous authors
Paper under double-blind review

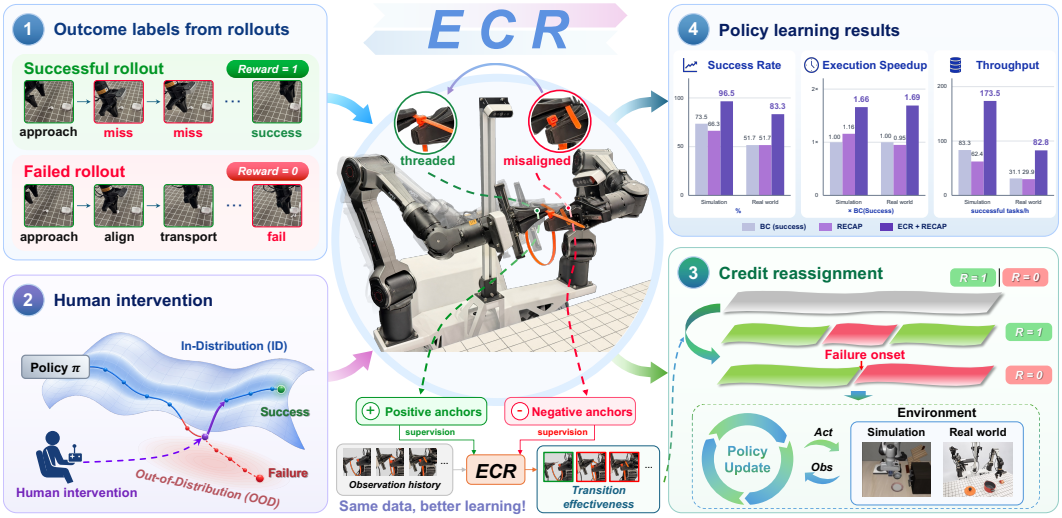


Figure 1: **Overview of ECR.** ECR learns transition effectiveness from episode outcomes, annotated failure onsets, and human interventions to reassign local credit for policy improvement.

ABSTRACT

Learning from deployment experience can improve robot policies beyond demonstrations. Sparse episode-level outcomes, however, obscure how credit should be assigned within a rollout. Successful rollouts may include locally ineffective behavior, whereas failed rollouts may still exhibit substantial productive execution. Even dense progress or value supervision can conflate effective task progress with elapsed interaction time. To address these issues, we introduce **Event-Grounded Credit Reassignment (ECR)**, which learns local transition effectiveness from episode outcomes, human interventions, and sparse annotations of failure onset. ECR models uncertainty about when ineffective execution begins and marginalizes over plausible onset locations to construct soft supervision for otherwise unlabeled transitions. The resulting estimates can then revise value targets for advantage-conditioned policy learning and reward signals for offline and online reinforcement learning. We instantiate these settings with RECAP, IQL, and RL Token, respectively. Across four simulation tasks and four real-world tasks, ECR improves the mean success rate over RECAP by 31.0 percentage points and achieves 2.78× the mean throughput of RECAP. In offline evaluation on the same task suite, ECR achieves a mean AUROC of 95.3% and a mean AP of 93.3% for distinguishing effective from ineffective transitions. Additional experiments with IQL and RL Token show consistent gains in offline and online reinforcement learning settings. Our results demonstrate that correcting local credit assignment improves policy learning from mixed-quality robot experience.

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.

1 INTRODUCTION

Pretrained vision-language-action (VLA) models provide strong priors for robot manipulation (Physical Intelligence et al., 2025b; 2026; Bjorck et al., 2025), but their supervised adaptation to downstream tasks remains constrained by the coverage of task demonstrations. Reinforcement learning provides a complementary path for policy improvement by allowing these models to learn directly from experience collected during their own execution (Xu et al., 2026a; Physical Intelligence et al., 2025a). However, episode outcomes provide only coarse supervision for these rollouts, leaving local credit assignment unresolved. A successful outcome does not imply that every local behavior contributed to task progress, just as eventual failure does not indicate that all preceding execution was ineffective. The final outcome alone therefore does not reveal which transitions were locally productive.

Process reward and value models alleviate outcome sparsity by providing denser estimates of task progress or expected return (Huang et al., 2026; Liang et al., 2026; Xu et al., 2026b; Ma et al., 2023; 2025). However, denser feedback alone does not ensure that credit is assigned to the behavior that actually advances the task. In particular, when supervision is derived primarily from temporal position along a rollout, the resulting target may remain favorable through locally ineffective execution. Conversely, an episode-level return can reflect eventual failure without indicating which preceding transitions were locally effective. Recent work has therefore incorporated richer temporal and interaction signals into process and value learning (Qin et al., 2026; Liu et al., 2026). These developments motivate explicitly learning local transition effectiveness to guide credit assignment within mixed-quality rollouts.

We introduce Event-Grounded Credit Reassignment (ECR), which learns local transition effectiveness from episode outcomes, human interventions, and sparse annotations of failure onset. Rather than treating an observed event as a precise onset boundary, ECR represents the onset with a distribution over plausible locations within each event window. We couple this onset distribution to transition effectiveness through its cumulative distribution, termed Event-CDF. Event-CDF converts onset uncertainty into soft effectiveness targets for otherwise unlabeled transitions. The resulting transition effectiveness estimates are used for value target revision in RECAP and reward correction in IQL and RL Token (Physical Intelligence et al., 2025a; Kostrikov et al., 2021; Xu et al., 2026a).

Across four simulation tasks and four real-world tasks, ECR improves the mean success rate over RECAP by 31.0 percentage points and achieves $2.78\times$ its mean throughput. Offline analysis further shows that ECR distinguishes effective from ineffective transitions within both successful and failed rollouts. Additional experiments with IQL and RL Token show gains under offline and online reinforcement learning objectives. Our contributions are as follows:

1. We introduce ECR, which treats local transition effectiveness as distinct from episode outcome and task progress and uses learned effectiveness estimates to reassign credit within robot rollouts of mixed quality.
2. We develop Event-CDF coupling, which links uncertainty over the onset of ineffective execution to transition effectiveness by marginalizing over plausible onset locations to construct soft supervision for otherwise unlabeled transitions.
3. We evaluate ECR across four simulation and four real-world tasks, demonstrating improved transition effectiveness estimation and gains in success and throughput with RECAP. We further apply the learned transition effectiveness signal to IQL and RL Token through reward correction, extending the evaluation to offline and online reinforcement learning.

2 RELATED WORK

2.1 PROCESS REWARD AND VALUE LEARNING

Visual reward and value models provide dense feedback through cross-embodiment alignment and value-based representations (Zakka et al., 2022; Ma et al., 2022; 2023). Pretrained vision-language models also support zero-shot reward specification (Rocamonde et al., 2024) and in-context progress estimation (Ma et al., 2025). RoboReward learns general-purpose reward models from robot videos (Lee et al., 2026). Recent methods further exploit temporal distance and procedure-grounded

progress (Huang et al., 2026; Feng et al., 2026), as well as subtask and stage annotations (Kim et al., 2025; Chen et al., 2026).

Reward learning also exploits trajectory quality and local execution structure. Ranked demonstration methods infer rewards from relative trajectory quality (Brown et al., 2019; 2020), while recent robotic reward and value models use trajectory comparisons, success and failure visitation, signed progress, and retry events to learn from mixed-quality experience (Liang et al., 2026; Tsao et al., 2026; Xu et al., 2026b; Qin et al., 2026). ReWiND uses video-rewinding augmentation to model decreasing task progress (Zhang et al., 2025). ECR instead models local transition effectiveness as the intermediate quantity for credit reassignment. This separates local execution quality from task progress and episode return.

2.2 Human Intervention and Credit Assignment

Interactive imitation learning collects expert feedback on states visited by the learner (Ross et al., 2011; Kelly et al., 2019). Intervention signals also inform execution-quality estimation (Spencer et al., 2020) and data reweighting (Liu et al., 2025). RLIF derives rewards from intervention feedback (Luo et al., 2024), while HIL-SERL combines human corrections with reinforcement learning for robot manipulation (Luo et al., 2025).

Recent work uses intervention structure more directly for local credit assignment. HABC separates viability and efficiency signals and restricts outcome supervision to autonomous segments, while GAINS models return variability associated with delayed and inconsistent intervention timing (Fang et al., 2026; Zhang et al., 2026). PACT is closely related to ECR in its focus on local credit correction, using a progress model to identify suboptimal segments and preferences induced by human interventions to recalibrate Bellman targets (Liu et al., 2026). ECR interprets intervention events as temporally ambiguous evidence about preceding autonomous execution. Event-CDF converts this ambiguity into soft transition supervision, and the resulting transition effectiveness estimates can revise value targets or rewards.

3 Method

ECR learns local transition effectiveness as an intermediate signal for credit reassignment within robot rollouts. As illustrated in Fig. 2, ECR predicts an effectiveness logit and an onset logit for each completed transition from visual observations and executed actions. Direct transition anchors, annotated failure onsets, and Event-CDF targets jointly supervise the estimator. The learned effectiveness scores and onset distribution support value target revision in RECAP, while the effectiveness scores provide reward correction in IQL and RL Token.

3.1 Problem Setup

We consider a robot policy $\pi_\theta(a_t \mid o_t, \ell)$ conditioned on observation o_t and task instruction ℓ. Executing the policy yields a rollout $\tau = (o_0, a_0, o_1, \ldots, a_{T-1}, o_T)$ with binary outcome $Y \in \{0, 1\}$, where $Y = 1$ denotes task success and $Y = 0$ task failure. For $t = 1, \ldots, T$, let $u_t = (o_{t-1}, a_{t-1}, o_t)$ be the completed transition ending at o_t. We use local transition effectiveness to characterize whether a completed transition constitutes productive execution toward task completion, separately from the final episode outcome or temporal progress.

For each transition u_t, let $H_t = (\mathcal{O}_t, \mathcal{A}_t)$ denote its observation and action context. The observation context contains the current and previous observations together with four sparsely sampled earlier observations, while the action context contains recently executed actions. ECR maps H_t to an effectiveness logit f_t and a scalar onset logit b_t. The effectiveness logit represents the predicted local effectiveness of transition u_t, while the onset logits associated with transitions in an event window are later normalized to form a distribution over candidate onset locations.

We assume access to expert demonstrations and policy rollouts, some of which contain human interventions. For a human intervention triggered by an observed execution error, the preceding autonomous segment defines an event window whose final ineffective segment has an unknown onset. For every failed rollout, an annotation z^* marks the first transition of the final ineffective segment

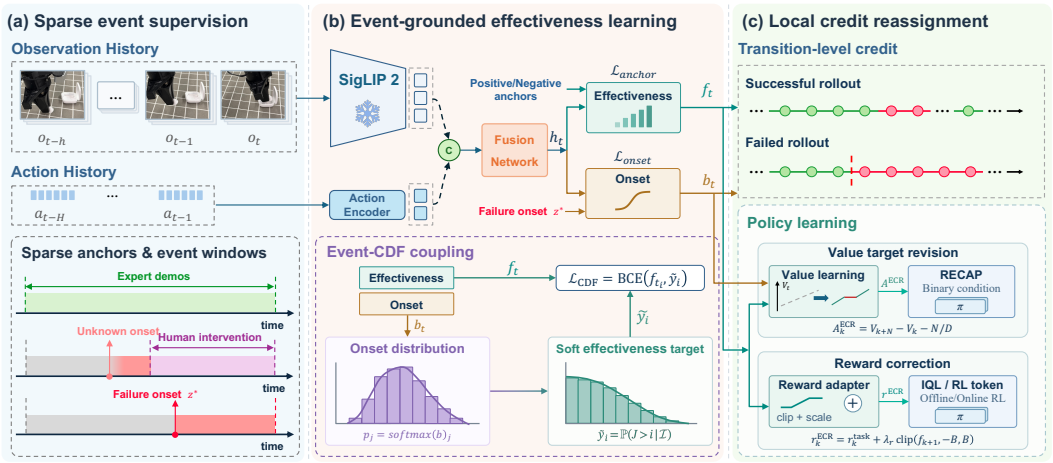


Figure 2: **ECR framework.** (a) Demonstrations, human interventions, and failure onset annotations provide sparse supervision. (b) Visual observations and executed actions are encoded to predict effectiveness and onset logits, while Event-CDF coupling provides soft transition targets. (c) The learned effectiveness and onset estimates support value target revision in RECAP, while effectiveness estimates provide reward correction in IQL and RL Token.

within its last autonomous segment. These signals provide direct transition supervision, event windows with latent onsets, and direct onset supervision.

3.2 Event-CDF Coupling

Consider an event window with transition indices $\mathcal{I} = (t_1, \ldots, t_m)$, where t_i denotes the rollout index of its i-th transition. Let $J \in \{1, \ldots, m\}$ denote the local position of the first transition in its final ineffective segment. ECR couples this latent onset to transition effectiveness by associating each candidate onset $J = j$ with a binary effectiveness target:

$$y_i^{(j)} = \mathbf{1}\{i < j\}, \qquad i = 1, \ldots, m, \tag{1}$$

so that transitions preceding j are treated as effective and those from j onward as ineffective.

The pointwise onset logits b_{t_i} define a distribution over candidate onset locations:

$$p_j \equiv \mathbb{P}(J = j \mid \mathcal{I}) = \frac{\exp(b_{t_j})}{\sum_{k=1}^{m} \exp(b_{t_k})}, \qquad j = 1, \ldots, m. \tag{2}$$

Event-CDF transfers this uncertainty over onset location to transition effectiveness by marginalizing the targets associated with all candidate onsets:

$$\widetilde{y}_i = \mathbb{E}_{J \sim p(\cdot \mid \mathcal{I})}\left[y_i^{(J)}\right] = \sum_{j=i+1}^{m} p_j, \qquad i = 1, \ldots, m. \tag{3}$$

Equivalently, $\widetilde{y}_i = \mathbb{P}(J > i \mid \mathcal{I})$, which gives the probability that transition u_{t_i} occurs before the final ineffective segment begins. The complement of the onset cumulative distribution therefore serves as a soft target for transition effectiveness.

3.3 Learning from Direct and Event Supervision

Direct supervision. Direct transition supervision is constructed from positive and negative anchors. Positive anchors comprise transitions from expert demonstrations and transitions executed under human control. Negative anchors comprise the final autonomous transition immediately preceding a human intervention and, for failed rollouts, the transitions from the annotated onset onward in the final autonomous segment. Let $y_t \in \{0, 1\}$ denote the resulting anchor label. We sample

anchor transitions according to $\rho_{\mathcal{A}}$, which balances positive and negative labels as well as the contributing sources within each label.

For each failed rollout, let $\mathcal{C} = (c_1, \ldots, c_n)$ denote the transition indices of its final autonomous segment, and let j^* denote the local position of the annotated onset, such that $z^* = c_{j^*}$. The direct supervision objectives are

$$\begin{aligned} \mathcal{L}_{\text{anchor}} &= \mathbb{E}_{(H_t, y_t) \sim \rho_{\mathcal{A}}} \left[\text{BCE}(f_t, y_t) \right], \\ \mathcal{L}_{\text{onset}} &= \mathbb{E}_{(\mathcal{C}, j^*)} \left[-\log \mathbb{P}(J = j^* \mid \mathcal{C}) \right]. \end{aligned} \tag{4}$$

Here, $\text{BCE}(f, y)$ denotes binary cross entropy applied to logit f and target y. The onset objective supervises the onset distribution using the annotated onset location.

Marginalized event supervision. For event windows without a direct onset annotation, the Event-CDF targets in Eq. 3 provide soft transition supervision:

$$\mathcal{L}_{\text{CDF}} = \mathbb{E}_{\mathcal{I}} \left[\frac{1}{m} \sum_{i=1}^{m} \text{BCE}\left(f_{t_i}, \widetilde{y}_i\right) \right]. \tag{5}$$

Gradients are retained through $\widetilde{y}_i$, allowing the onset distribution and transition effectiveness predictions to be optimized jointly. The complete ECR objective combines these supervision terms:

$$\mathcal{L}_{\text{ECR}} = \lambda_a \mathcal{L}_{\text{anchor}} + \lambda_z \mathcal{L}_{\text{onset}} + \lambda_c \mathcal{L}_{\text{CDF}}, \tag{6}$$

where λ_a, λ_z, and λ_c weight direct transition supervision, direct onset supervision, and Event-CDF supervision, respectively. The direct objectives supervise the effectiveness and onset outputs, while $\mathcal{L}_{\text{CDF}}$ couples them through the marginalized transition targets.

3.4 Credit Reassignment for Policy Learning

The learned effectiveness estimates can be incorporated into policy learning by revising the supervision from which credit is derived. We consider two forms of credit reassignment: value target revision and reward correction. We instantiate these interfaces with RECAP, IQL, and RL Token.

Value target revision. RECAP derives binary policy conditions from temporal differences in a learned value signal (Physical Intelligence et al., 2025a). ECR revises the value targets using predicted local transition effectiveness.

For a successful rollout, the revised value target is defined recursively as

$$\bar{v}_T = 0, \qquad \bar{v}_k = \bar{v}_{k+1} - \frac{g_{k+1}}{D}, \tag{7}$$

where $g_{k+1} = \mathbf{1}\{f_{k+1} \geq 0\}$ indicates whether transition u_{k+1} is predicted effective, and D is the task's maximum trajectory length. Thus, predicted ineffective transitions leave the target unchanged.

For a failed rollout with final autonomous segment $\mathcal{C} = (c_1, \ldots, c_n)$, we infer the onset from the learned distribution:

$$\widehat{j} = \arg\max_{1 \leq j \leq n} \mathbb{P}(J = j \mid \mathcal{C}), \qquad \widehat{z} = c_{\widehat{j}}, \tag{8}$$

where $\widehat{z}$ is the rollout index of the most probable local onset $\widehat{j}$. The failed rollout target is then constructed as

$$\begin{aligned} \bar{v}_0 &= -\frac{\bar{L}_{\text{succ}}}{D}, \qquad \bar{v}_{k+1} = \bar{v}_k + \frac{\delta_{k+1}}{D}, \\ \delta_{k+1} &= \begin{cases} g_{k+1}, & k+1 < \widehat{z}, \\ 2g_{k+1} - 1, & k+1 \geq \widehat{z}, \end{cases} \end{aligned} \tag{9}$$

where $\bar{L}_{\text{succ}}$ is the mean length of successful reference trajectories. Predicted ineffective transitions yield zero target increments before $\widehat{z}$ and negative increments thereafter.

Following RECAP, we clip the revised targets to $[-1, 0]$ and fit the same distributional value model. Let V_k be its expected value prediction at step k. We compute

$$A_k^{\text{ECR}} = V_{k+N} - V_k - \frac{N}{D}, \tag{10}$$

where N is the lookahead horizon. Within each task, the highest-scoring 30% of training transitions receive the positive policy condition.

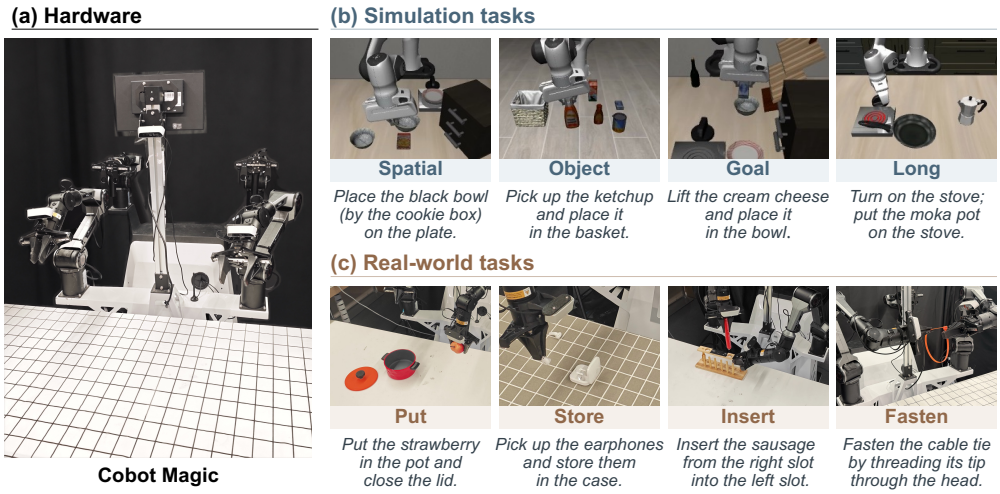


Figure 3: **Evaluation platform and tasks.** (a) The Cobot Magic dual-arm robot used for real-world experiments. (b) Four LIBERO-PRO tasks (Zhou et al., 2025) spanning Spatial, Object, Goal, and Long. (c) Four real-world tasks: Put, Store, Insert, and Fasten.

Reward correction. When policy learning consumes transition rewards directly, ECR augments the task reward with local transition effectiveness:

$$r_k^{\mathrm{ECR}} = r_k^{\mathrm{task}} + \lambda_r \,\mathrm{clip}\left(f_{k+1}, -B, B\right), \tag{11}$$

where f_{k+1} scores the completed transition $u_{k+1} = (o_k, a_k, o_{k+1})$, λ_r scales the correction, and B clips the effectiveness logit. For IQL, r_k^{ECR} replaces the task reward in the Q function backup (Kostrikov et al., 2021); for RL Token, it is used in critic updates (Xu et al., 2026a).

4 Experiments

We evaluate whether sparse event supervision enables ECR to learn transition effectiveness that improves downstream policy learning. Our experiments focus on four questions: **(Q1)** Does ECR improve policy learning from mixed-quality robot experience? **(Q2)** Does ECR distinguish effective from ineffective autonomous transitions in held-out trajectories and improve credit assignment? **(Q3)** Which components of ECR drive these improvements, and what annotation effort is required? **(Q4)** Can the learned effectiveness signal also benefit offline and online reinforcement learning?

4.1 Experimental Setup

Tasks and data. We evaluate ECR on four LIBERO-PRO tasks and four real-world tasks using a Cobot Magic dual-arm robot, as shown in Fig. 3. The simulation evaluation includes one task from each of Spatial, Object, Goal, and Long, while the real-world tasks are Put, Store, Insert, and Fasten. For LIBERO-PRO, we use the Position evaluation setting, varying target object positions while keeping the task instructions fixed. All policies are initialized from the pretrained $\pi_{0.5}$ checkpoint (Physical Intelligence et al., 2025b) and fine-tuned on 50 demonstrations per task. We then collect 30 successful and 10 failed policy rollouts for each task. During rollout collection, human corrections are provided through gamepad control in simulation and teleoperation on the real robot. Each failed rollout receives one annotation z^* marking the first transition of its final ineffective segment. The average onset annotation time was 6.0 seconds per failed rollout. The annotation protocol and interface are provided in Appendices B.2 and B.3.

Baselines. We compare behavior cloning trained on demonstrations and successful policy rollouts (BC-S) and its counterpart additionally using failed rollouts (BC-SF), standard RECAP, and RECAP variants using signals from RynnValue (RynnV.) (Huang et al., 2026), Robo-Dopamine 2.0 (RoboD.) (Xu et al., 2026b), Robometer (RoboM.) (Liang et al., 2026), and Success Visitation

Table 1: **Simulation policy performance.** Results are averaged over two evaluation seeds, with 50 autonomous episodes per seed. Bold and underlined entries mark the best and second-best results per row.

Task	BC-S	BC-SF	RECAP	RynnV.	RoboD.	RoboM.	SVM	ECR-Sign	**ECR**
A. Success rate (%) ↑									
Spatial	87.0	12.0	42.0	60.0	67.0	70.0	81.0	**94.0**	91.0
Object	47.0	30.0	57.0	78.0	90.0	79.0	33.0	81.0	**99.0**
Goal	75.0	48.0	72.0	46.0	46.0	66.0	86.0	95.0	**98.0**
Long	85.0	84.0	94.0	95.0	79.0	85.0	86.0	93.0	**98.0**
Avg.	73.5	43.5	66.3	69.8	70.5	75.0	71.5	90.8	**96.5**
B. Speedup on successful episodes (×) ↑									
Spatial	1.00	0.99	0.91	1.01	1.01	0.98	0.98	**1.05**	**1.05**
Object	1.00	1.08	1.66	1.76	**2.28**	1.57	1.30	1.56	1.73
Goal	1.00	1.12	1.11	1.36	1.03	1.16	1.19	2.39	**2.77**
Long	1.00	0.96	1.08	1.44	1.43	1.30	1.02	1.35	**1.51**
Geo. mean	1.00	1.04	1.16	1.37	1.36	1.23	1.11	1.52	**1.66**
C. Throughput (tasks/h) ↑									
Spatial	176.4	15.3	61.1	100.6	117.8	123.7	154.4	**211.5**	197.8
Object	35.4	22.7	55.4	90.6	**142.1**	85.4	26.4	88.3	134.9
Goal	71.2	42.7	71.9	43.1	39.3	65.5	98.3	214.5	**271.4**
Long	50.0	47.6	61.1	81.1	58.2	61.4	51.4	73.5	**89.9**
Avg.	83.3	32.1	62.4	78.8	89.3	84.0	82.6	147.0	**173.5**

Matching (SVM) (Tsao et al., 2026). We also evaluate ECR-Sign, which directly maps the sign of the predicted effectiveness logit to binary policy conditions. In these comparisons, ECR refers to the full method, which revises RECAP value targets using the learned effectiveness and onset estimates.

Metrics. Simulation results are averaged over two independent evaluation seeds, with 50 evaluation episodes per seed. For each real-world task, we evaluate every method over 15 autonomous trials. Timeouts count as failures. We report task success rate, execution speedup relative to BC-S, and successful task throughput. Speedup compares the mean duration of successful episodes with that of BC-S. Throughput measures successful tasks per hour, counting each failed trial as consuming the full task budget (Appendix C.1). Transition effectiveness metrics are evaluated on autonomous transitions only, excluding expert demonstrations and actions under human control.

Table 2: **Real-world policy performance.** 15 autonomous trials per method and task. Bold and underlined entries mark the best and second-best results per row.

Task	BC-S	BC-SF	RECAP	RynnV.	RoboD.	RoboM.	SVM	ECR-Sign	**ECR**
A. Success rate (%) ↑									
Put	73.3	46.7	73.3	86.7	93.3	80.0	66.7	93.3	**100.0**
Store	60.0	20.0	66.7	60.0	66.7	53.3	73.3	73.3	**86.7**
Insert	46.7	20.0	46.7	53.3	53.3	60.0	53.3	60.0	**80.0**
Fasten	26.7	6.7	20.0	26.7	46.7	40.0	33.3	26.7	**66.7**
Avg.	51.7	23.3	51.7	56.7	65.0	58.3	56.7	63.3	**83.3**
B. Speedup on successful episodes (×) ↑									
Put	1.00	0.77	0.74	1.46	1.43	1.01	0.83	1.43	**1.50**
Store	1.00	0.70	1.06	1.16	1.11	0.98	1.35	1.48	**1.78**
Insert	1.00	0.87	1.04	1.15	1.15	1.28	1.14	1.55	**1.92**
Fasten	1.00	0.84	0.98	0.99	1.45	1.59	1.44	1.28	**1.60**
Geo. mean	1.00	0.79	0.95	1.18	1.27	1.19	1.17	1.43	**1.69**
C. Throughput (tasks/h) ↑									
Put	52.4	26.2	43.1	89.5	103.5	59.9	41.4	103.2	**127.0**
Store	34.5	9.1	40.9	37.0	41.8	29.4	54.0	56.6	**88.4**
Insert	28.4	11.0	28.8	35.3	35.2	43.1	35.1	46.9	**84.4**
Fasten	8.9	2.1	6.6	8.9	18.7	15.8	12.4	9.4	**31.5**
Avg.	31.1	12.1	29.9	42.7	49.8	37.1	35.7	54.0	**82.8**

4.2 Policy Performance

In Tables 1 and 2, ECR achieves mean success rates of 96.5% in simulation and 83.3% in the real world, outperforming RECAP on all eight tasks. Its mean throughput is 2.78× and 2.77× that of RECAP in simulation and the real world, respectively. The corresponding geometric mean speedups on successful episodes are 1.66× and 1.69× relative to BC-S.

Compared with BC-S, BC-SF reduces mean success from 73.5% to 43.5% in simulation and from 51.7% to 23.3% on the real-world tasks, showing that treating all actions as equally desirable imitation targets is insufficient.

With RECAP fixed as the downstream learner, variants using RynnV., RoboD., RoboM., and SVM remain below ECR in mean success across both settings. Holding the ECR estimator fixed, the full ECR method improves mean success rates from 90.8% to 96.5% in simulation and from 63.3% to 83.3% on the real-world tasks compared with ECR-Sign.

4.3 Credit Assignment Analysis

Transition effectiveness. We evaluate effectiveness on held-out autonomous transitions using five folds split by trajectory. Experts annotate all clearly ineffective segments in each trajectory; the resulting dense transition labels are used only for evaluation. For AP and AUROC, ineffective transitions are treated as the positive class.

Table 3: **Transition effectiveness on held-out trajectories.** All values are percentages. Succ. and Fail. report AUROC within successful and failed trajectories, respectively. Bold and underlined entries indicate the best and second-best values per column.

	Simulation				Real World			
Method	AUROC ↑	AP ↑	Succ. ↑	Fail. ↑	AUROC ↑	AP ↑	Succ. ↑	Fail. ↑
RynnV.	<u>71.1</u>	68.0	70.2	<u>72.2</u>	52.6	50.7	53.4	45.8
RoboD.	68.5	72.4	63.5	71.9	<u>70.4</u>	<u>64.2</u>	<u>68.1</u>	<u>74.4</u>
RoboM.	58.1	60.8	56.4	59.2	58.3	54.8	57.6	60.1
SVM	69.6	<u>73.9</u>	<u>70.7</u>	57.7	65.2	59.3	64.0	56.8
ECR	**97.1**	**96.8**	**97.3**	**96.2**	**93.5**	**89.8**	**94.5**	**91.6**

As shown in Table 3, ECR achieves AP of 96.8% in simulation and 89.8% in the real world, exceeding the strongest baseline in each setting by 22.9 and 25.6 percentage points, respectively. Within successful and failed trajectories, ECR achieves AUROC of 97.3% and 96.2% in simulation and 94.5% and 91.6% in the real world, distinguishing local effectiveness even when the episode outcome is fixed.

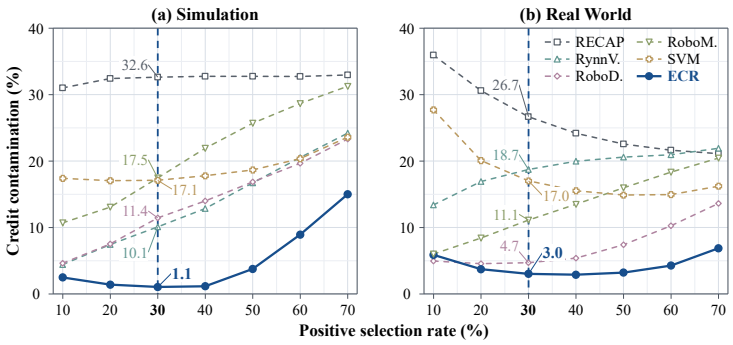


Figure 4: **Credit contamination across positive selection rates.**

Policy credit across selection rates. We compare credit contamination at matched positive selection rates from 10% to 70%. Within each task, all training transitions are ranked by their condition-

ing scores, and the top $p\%$ receive the positive condition. Credit contamination is the fraction of positively conditioned transitions annotated as ineffective.

ECR achieves the lowest credit contamination at every evaluated rate in simulation. In the real world, RoboD. is slightly lower at 10%, while ECR achieves the lowest contamination at all remaining rates. At the 30% operating point used for policy training, credit contamination is 1.1% in simulation and 3.0% in the real world, the lowest among all compared signals. Together, Table 3 and Fig. 4 show that ECR improves both transition discrimination on held-out trajectories and the quality of the policy conditions derived from its scores.

4.4 Ablation Analysis

We isolate three design choices in ECR under matched supervision and training conditions. *Direct* retains only $\mathcal{L}_{\text{anchor}}$ and $\mathcal{L}_{\text{onset}}$. *MAP targets* replace soft Event-CDF targets with hard prefix and suffix targets induced by the most probable onset. *Stop-gradient* retains the soft Event-CDF targets but blocks gradients through $\widetilde{y}_i$.

Held-out AP follows Sec. 4.3. Success rate (SR) and throughput (TPH, successful tasks per hour) are measured using policies trained with each estimator. All three metrics are averaged over the four simulation tasks.

Table 4 shows that ECR achieves higher AP (96.8 vs. 93.0), SR (96.5 vs. 92.0), and TPH (173.5 vs. 149.6) than MAP targets, the strongest ablated variant. Removing Event-CDF supervision or blocking gradients through its targets also lowers all three metrics.

Table 4: **ECR component analysis in simulation.**

Variant	AP ↑	SR ↑	TPH ↑
Direct	88.1	88.8	135.4
MAP targets	93.0	92.0	149.6
Stop-gradient	89.4	90.3	141.0
ECR	**96.8**	**96.5**	**173.5**

4.5 Extensions to Offline and Online RL

We apply ECR reward correction (Eq. 11) to IQL and RL Token on two real-world tasks, Store and Insert. For IQL, we match the offline data and optimization budget. For RL Token, we match the initial policy and online interaction budget. During online RL, the ECR estimator is updated after each round of data collection.

As shown in Table 5, ECR increases mean success from 53.3% to 73.3% for IQL and from 56.7% to 83.3% for RL Token. Beyond value target revision, ECR also improves policy learning through reward correction in both offline and online RL.

Table 5: **ECR reward correction in offline and online RL.** Success (%) over 15 autonomous trials per method and task.

Method	Insert	Store	Avg.
Offline RL			
IQL	46.7	60.0	53.3
IQL + ECR	**73.3**	**73.3**	**73.3**
Online RL			
RL Token	53.3	60.0	56.7
RL Token + ECR	**80.0**	**86.7**	**83.3**

5 Conclusion and Limitations

Dense progress or value feedback does not by itself determine which transitions deserve learning credit. ECR addresses this ambiguity by learning local transition effectiveness and marginalizing uncertainty over the onset of ineffective execution. Across four simulation and four real-world tasks, ECR improves effectiveness estimation on held-out trajectories and increases policy success and throughput through value target revision. Reward correction further improves IQL and RL Token on two real-world tasks.

Limitations. ECR requires failure onset annotations, and its single-onset targets do not represent repeated errors and recoveries within an event window. Its additive reward correction does not guarantee preservation of the original optimal policy. Future work includes recovery-aware event models and reward correction with policy-invariance guarantees.

AI USE STATEMENT

We used generative AI tools to polish the manuscript, assist with code optimization and debugging, and provide feedback on experimental design. These tools served an advisory and supporting role, with all final decisions made by the authors. The authors reviewed the AI-assisted edits, code changes, and suggestions and take full responsibility for the content of the paper, the implementation, and the reported results.

REPRODUCIBILITY STATEMENT

Implementation details, data collection and annotation protocols, evaluation metrics, and task-level results are provided in Appendices B–E. Algorithm 1 summarizes ECR value-target revision and RECAP policy training. We will release the implementation upon acceptance.

REFERENCES

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

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

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

Qianzhong Chen, Justin Yu, Mac Schwager, Pieter Abbeel, Fred Shentu, and Philipp Wu. Sarm: Stage-aware reward modeling for long horizon robot manipulation. In *International Conference on Learning Representations*, volume 2026, pp. 17091–17116, 2026.

Tongyan Fang, Siyuan Huang, Naiyu Fang, Ganlong Zhao, Zhongjin Luo, Jianbo Liu, Xiaogang Wang, Ying Dong, and Hongsheng Li. Hierarchical advantage weighting for online rl fine-tuning of vlas from sparse episode outcomes. *arXiv preprint arXiv:2606.17043*, 2026.

Youhe Feng, Hansen Shi, Haoyang Li, Xinlei Guo, Yang Wang, Chengyang Zhang, Jinkai Zhang, Xiaohan Zhang, Jie Tang, and Jing Zhang. Procvlm: Learning procedure-grounded progress rewards for robotic manipulation. *arXiv preprint arXiv:2605.08774*, 2026.

Dongchi Huang, Hongyin Zhang, Bohan Hou, Siteng Huang, Zhian Su, Hang Guo, Tong Lu, Zhaofeng Xu, Jiahao Tang, Jianfei Yang, et al. Rynnvalue: Scaling robotic value foundation models with temporal distance. *arXiv preprint arXiv:2608.09853*, 2026.

Michael Kelly, Chelsea Sidrane, Katherine Driggs-Campbell, and Mykel J Kochenderfer. Hg-dagger: Interactive imitation learning with human experts. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8077–8083. IEEE, 2019.

Changyeon Kim, Minho Heo, Doohyun Lee, Honglak Lee, Jinwoo Shin, Joseph Lim, and Kimin Lee. Subtask-aware visual reward learning from segmented demonstrations. In *International Conference on Learning Representations*, volume 2025, pp. 15142–15165, 2025.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

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

Anthony Liang, Yigit Korkmaz, Jiahui Zhang, Minyoung Hwang, Abrar Anwar, Sidhant Kaushik, Aditya Shah, Alex S Huang, Luke Zettlemoyer, Dieter Fox, et al. Robometer: Scaling general-purpose robotic reward models via trajectory comparisons. *arXiv preprint arXiv:2603.02115*, 2026.

Huihan Liu, Soroush Nasiriany, Lance Zhang, Zhiyao Bao, and Yuke Zhu. Robot learning on the job: Human-in-the-loop autonomy and learning during deployment. *The International Journal of Robotics Research*, 44(10-11):1727–1742, 2025.

Zeyi Liu, Guangyao Liu, Yinuo Qu, Yuquan Xue, Bofang Jia, Chunhua Yang, Weihua Gui, Keke Huang, and Ziwei Wang. Preference-calibrated human-in-the-loop reinforcement learning for robotic manipulation. *arXiv preprint arXiv:2606.03949*, 2026.

Jianlan Luo, Perry Dong, Yuexiang Zhai, Yi Ma, and Sergey Levine. Rlif: Interactive imitation learning as reinforcement learning. In *International Conference on Learning Representations*, volume 2024, pp. 36329–36351, 2024.

Jianlan Luo, Charles Xu, Jeffrey Wu, and Sergey Levine. Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning. *Science Robotics*, 10(105):eads5033, 2025.

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. *arXiv preprint arXiv:2210.00030*, 2022.

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

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

Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, et al. $\pi^*_{0.6}$: a VLA that learns from experience. *arXiv preprint arXiv:2511.14759*, 2025a.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025b.

Physical Intelligence, Bo Ai, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Greg Balke, Kevin Black, George Bokinsky, Shihao Cao, Thomas Charbonnier, et al. $\pi_{0.7}$: a steerable generalist robotic foundation model with emergent capabilities. *arXiv preprint arXiv:2604.15483*, 2026.

Xinyao Qin, Junjie Lu, Kaixin Wang, Chuheng Zhang, Sinjae Kang, Kimin Lee, Min Xu, Bin Liang, Jun Yang, and Li Zhao. Beyond monotonic progress: Retry-supervised value learning for robot imitation. *arXiv preprint arXiv:2606.24633*, 2026.

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

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635. JMLR Workshop and Conference Proceedings, 2011.

Jonathan Spencer, Sanjiban Choudhury, Matt Barnes, Matthew Schmittle, Mung Chiang, Peter Ramadge, and Siddhartha Srinivasa. Learning from Interventions: Human-robot interaction as both explicit and implicit feedback. In *Proceedings of Robotics: Science and Systems*, Corvalis, Oregon, USA, July 2020. doi: 10.15607/RSS.2020.XVI.055.

Raymond Tsao, Andrew Wagenmaker, and Sergey Levine. Learning process rewards via success visitation matching for efficient rl. *arXiv preprint arXiv:2606.23640*, 2026.

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

Charles Xu, Jost Tobias Springenberg, Michael Equi, Ali Amin, Adnan Esmail, Sergey Levine, and Liyiming Ke. Rl token: Bootstrapping online rl with vision-language-action models. *arXiv preprint arXiv:2604.23073*, 2026a.

Yijie Xu, Haopeng Jin, Run Zhou, Shengbang Liu, Sixiang Chen, Hongyang Cheng, Sicheng Hu, Peterson Co, Jinwen Luo, Huajie Tan, et al. Robo-dopamine 2.0: History-conditioned and ood-aware process reward modeling for robotic manipulation. *arXiv preprint arXiv:2608.15680*, 2026b.

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

Jiahui Zhang, Yusen Luo, Abrar Anwar, Sumedh Anand Sontakke, Joseph J Lim, Jesse Thomason, Erdem Biyik, and Jesse Zhang. Rewind: Language-guided rewards teach robot policies without new demonstrations. *arXiv preprint arXiv:2505.10911*, 2025.

Xinyi Zhang, Yinuo Zhao, Pei Ren, Lechun Jiang, Huiqian Jin, Lei Sun, Dapeng Wu, Zhengping Che, Chi Harold Liu, and Jian Tang. Gains: Leveraging inconsistent human intervention signals in reinforcement learning. *arXiv preprint arXiv:2608.15707*, 2026.

Xueyang Zhou, Yangming Xu, Guiyao Tie, Yongchao Chen, Guowen Zhang, Duanfeng Chu, Pan Zhou, and Lichao Sun. Libero-pro: Towards robust and fair evaluation of vision-language-action models beyond memorization. *arXiv preprint arXiv:2510.03827*, 2025.

APPENDIX CONTENTS

A Additional Method Details **14**

A.1 Joint Optimization Through the Soft Targets 14

A.2 Value Targets and Relative Policy Conditions 14

A.3 Algorithmic Procedures 14

B Experimental Protocol and Implementation **14**

B.1 Tasks and Data Collection 14

B.2 Event Construction and Annotation 15

B.3 Collection and Annotation Interfaces 16

B.4 Estimator Training 16

B.5 Comparison Models and Score Alignment 18

B.6 Value Learning and Policy Conditioning 18

C Evaluation Protocols **19**

C.1 Closed-Loop Policy Metrics 19

C.2 Transition Discrimination 20

C.3 Credit Contamination 21

D Additional Results and Analysis **21**

D.1 Effectiveness by Task 21

D.2 Qualitative Analysis of Transition Scores 22

D.3 Task-Level Credit Contamination 25

D.4 Component Ablations 25

E Offline and Online Reinforcement Learning **26**

E.1 Offline IQL 26

E.2 RL Token Initialization 26

E.3 Single-Round Online Adaptation and Evaluation 27

A Additional Method Details

A.1 Joint Optimization Through the Soft Targets

We follow the main-text indexing: transition $u_t = (o_{t-1}, a_{t-1}, o_t)$ scores action a_{t-1}. Event-CDF normalizes over all onset candidates $J \in \{1, \dots, m\}$.

For fixed effectiveness logits, let $L_j = m^{-1} \sum_i \mathrm{BCE}(f_{t_i}, \mathbf{1}\{i < j\})$. Since $\mathrm{BCE}(f, y) = \mathrm{softplus}(f) - yf$ is affine in y, the window loss satisfies

$$L_{\mathcal{I}} = \frac{1}{m} \sum_{i=1}^{m} \mathrm{BCE}(f_{t_i}, \widetilde{y}_i) = \sum_{j=1}^{m} p_j L_j. \tag{12}$$

With σ denoting the sigmoid, the partial derivatives with respect to the output logits are

$$\frac{\partial L_{\mathcal{I}}}{\partial f_{t_i}} = \frac{\sigma(f_{t_i}) - \widetilde{y}_i}{m}, \qquad \frac{\partial L_{\mathcal{I}}}{\partial b_{t_j}} = p_j \big(L_j - L_{\mathcal{I}}\big). \tag{13}$$

The onset gradient favors candidate boundaries with lower transition loss, while the effectiveness logits fit their soft targets. Shared parameters receive both contributions and the direct-supervision gradients. Stop-gradient removes the path through $\widetilde{y}_i$ to the onset logits, retaining direct onset supervision and shared-parameter updates. A reverse cumulative sum constructs all targets in $O(m)$ time.

A.2 Value Targets and Relative Policy Conditions

For normalization, trajectory length is counted in observations, giving $D = \max_{\tau \in \mathcal{D}_{\mathrm{train}}} (T_\tau + 1)$. We compute $\overline{L}_{\mathrm{succ}}$ as the mean transition count over the task's 50 demonstrations and 30 successful policy rollouts. Algorithm 1 applies the main-text recursions and clips the completed target sequence to $[-1, 0]$. Full ECR uses soft targets during estimator training and MAP onset decoding during failed-trajectory reassignment; the MAP-target ablation instead changes estimator training.

Near a trajectory end, use $j = \min(k + N, T)$ and the actual horizon $h = j - k$ in the temporal score (Eq. 15). Here $N = 20$ in simulation and 50 on the real robot; $k = T$ has no outgoing action. Substituting the successful-rollout target (Eq. 7) into the temporal score gives the following identity when the value model reproduces that target exactly and clipping is inactive:

$$A_k^{\mathrm{ECR}} = \frac{\sum_{t=k+1}^{j} g_t - h}{D} \le 0. \tag{14}$$

Positive policy conditions are assigned by within-task score rank. ECR-Sign thresholds f_{k+1} at zero.

A.3 Algorithmic Procedures

Algorithm 1 converts the trained estimator into RECAP supervision. Score f_{k+1} describes action a_k, and condition c_k applies to the entire chunk starting at k. Full ECR uses soft Event-CDF targets for estimator training and MAP onset decoding for failed-trajectory value targets.

B Experimental Protocol and Implementation

B.1 Tasks and Data Collection

The simulation evaluation selects one task from each of the Spatial, Object, Goal, and Long suites of LIBERO-PRO. We use its Position setting, varying target-object positions while retaining the task instruction. The real-world evaluation uses a Cobot Magic dual-arm robot. The tasks are illustrated in Figure 3.

For each task, we fine-tune $\pi_{0.5}$ (Physical Intelligence et al., 2025b) on 50 teleoperation demonstrations and collect 30 successful and 10 failed policy rollouts. Corrections use gamepad control

Algorithm 1 ECR value targets and RECAP policy supervision

Require: One task's training trajectories and trained estimator F_ϕ; lookahead $N = 20$ (simulation) or 50 (real robot); actor chunk length H.
Ensure: Conditional policy trained on ECR-derived supervision.
Set $D \leftarrow \max_\tau(T_\tau + 1)$ and $\overline{L}_{\text{succ}} \leftarrow \text{mean}_{\tau:Y_\tau=1}\, T_\tau$.
for all training trajectories τ **do**
    Evaluate (f_t, b_t); set $g_t \leftarrow \mathbf{1}\{f_t \geq 0\}$ for $t = 1, \ldots, T$.
    If $Y = 0$, decode $\widehat{z} = c_{\widehat{j}}$ with $\widehat{j} = \arg\max_{1 \leq j \leq n} b_{c_j}$ over all positions of the final autonomous segment $\mathcal{C}$.
    Construct $\overline{v}_{0:T}$ using Eq. 7 for $Y = 1$ or Eq. 9 for $Y = 0$.
    Clip the completed target sequence $\overline{v}_{0:T}$ to $[-1, 0]$.
end for
Fit the distributional value model to these fixed targets (Appendix B.6).
for all training trajectories τ and action starts $0 \leq k < T_\tau$ **do**
    Set $j \leftarrow \min(k + N, T_\tau)$ and $h \leftarrow j - k$.
    For finite value endpoints, set $A_k^{\text{ECR}} \leftarrow V(o_j) - V(o_k) - h/D$.
end for
Assign chunk-start conditions c_k from valid scores using the task-wise 30% selection setting.
Train on both conditions with Eq. 16; deploy with the positive condition.

in simulation and teleoperation on the real robot. In some collection episodes, we allow several repeated erroneous attempts before intervening, retaining examples of sustained ineffective execution. IQL and IQL+ECR use the same task-specific offline pool. RL Token's separate warmup and single collection round are described in Appendix E.

Estimator training uses demonstration/human anchors, reactive events, and annotated failure onsets. Dense ineffective-interval labels are reserved for offline evaluation, with each test trajectory excluded from scorer fitting and normalization. Credit analysis instead ranks eligible training action positions, including demonstration and human-controlled segments.

Closed-loop evaluation uses 100 autonomous episodes per simulation task and method and 15 autonomous trials per real-world task and method. Timeouts are failures. Table 10 gives the evaluation limits.

B.2 Event Construction and Annotation

Training events. Reactive corrections follow an observed error; anticipatory corrections precede one. Only reactive corrections generate the pre-intervention negative anchor and Event-CDF window in Section 3.3. Demonstration and human-controlled transitions provide positive anchors. For failed episodes, z^* marks the first transition of the final ineffective suffix within the last autonomous segment: erroneous attempts persist from this point without subsequent recovery or task completion. Its suffix provides negative anchors and its position supervises the onset head.

Offline evaluation labels. Annotators mark ineffective intervals throughout successful and failed trajectories, using visible errors such as a missed grasp, object slippage, grasping the wrong target, failed insertion, or placing earphones outside the case. These dense interval labels are used only for offline evaluation. Reviewed autonomous transitions outside the annotated ineffective intervals and failure suffix form the effective class; demonstrations and human-controlled actions are excluded. An effective evaluation label does not imply positive policy advantage.

Onset annotation cost. Three annotators each label the same 80 trajectories spanning simulation and real-world tasks, then cross-check their annotations. Individual onset labeling averages 6.0 seconds per trajectory before cross-review, including video playback, revisiting frames, and final confirmation. Onsets recorded during collection are withheld from annotators during this timing study.

B.3 Collection and Annotation Interfaces

The collector automatically logs human/autonomous control ownership and supports recording failure onsets via a keyboard shortcut (Figure 5). The offline interface supports synchronized video review, onset selection, and multiple ineffective intervals (Figure 6); a completion flag distinguishes reviewed empty labels from unfinished reviews.

Simulation collection operates at 10 Hz with 10-action predictions and replanning every five steps. Records pair pre-action observation o_k with executed action a_k, scored at arrival observation o_{k+1}. Evaluation durations follow Appendix C.1.

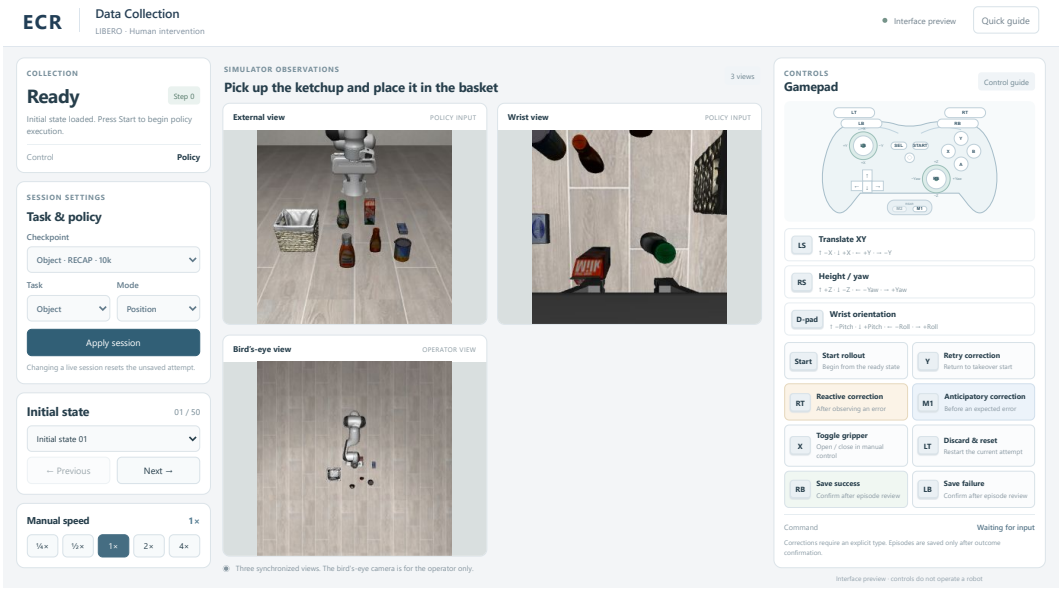


Figure 5: Simulation collection interface preview with synchronized observations and the control diagram. External and wrist views are policy inputs; the bird's-eye view is for the operator.

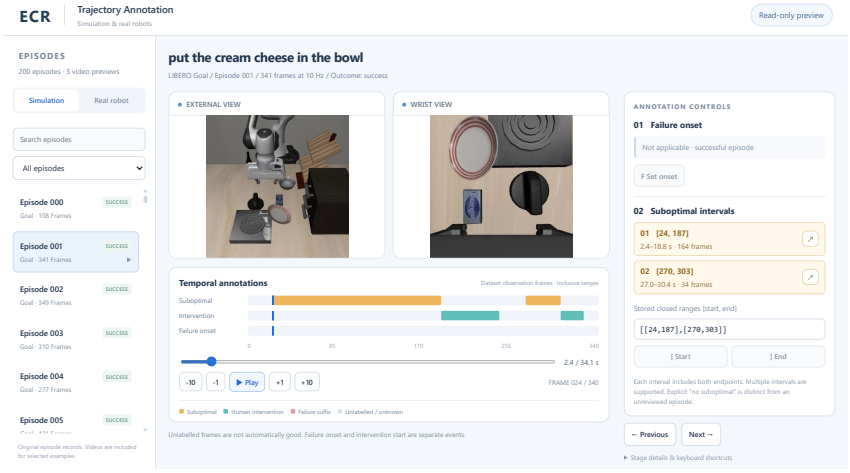
B.4 Estimator Training

The visual context comprises the current observation o_t, the previous observation o_{t-1}, and four earlier observations sampled at 2 Hz before o_{t-1}. We encode these observations with a frozen SigLIP 2 So400m/14 backbone at 224×224 resolution (Tschannen et al., 2025). Visual-context and action-history features are concatenated and processed by a shared fusion network with two scalar output heads for effectiveness and onset. Table 6 summarizes the inputs. Action normalization is fitted only on training trajectories.

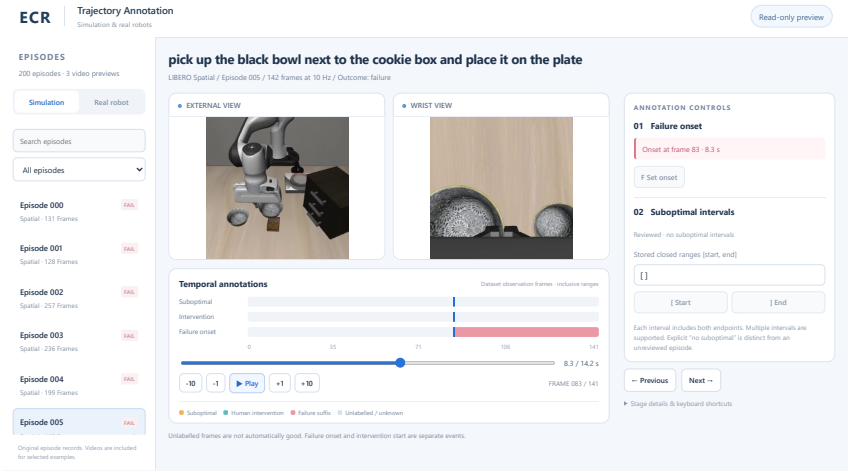
Table 6: Input configuration of the ECR estimator.

Setting	Simulation	Real world
Sampling rate	10 Hz	25 Hz
Camera views	External + wrist	External + two wrists
Visual history	Four observations before o_{t-1}, sampled at 2 Hz	
Action context	Recent executed-action history	
Visual encoder	Frozen SigLIP 2 So400m/14, 224×224	
Fusion inputs	Concatenated visual context and action history	

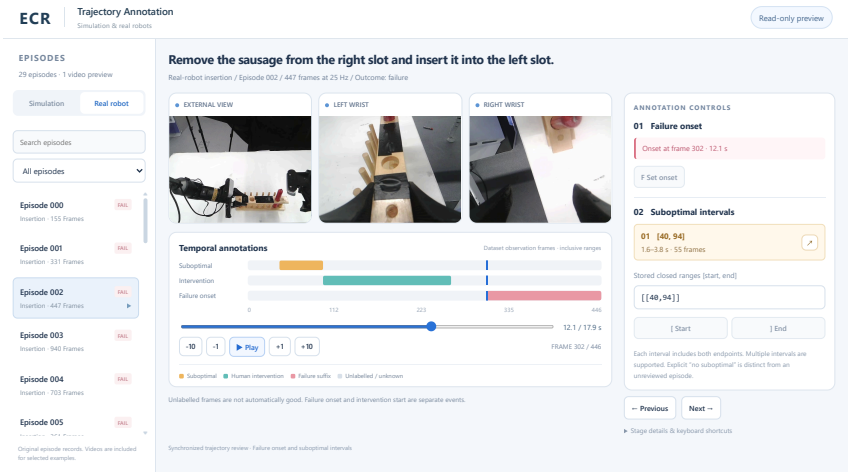
We train the estimator with Eq. 6, using $\lambda_a = \lambda_z = \lambda_c = 1$. We use AdamW with a constant learning rate of 10^{-4} and weight decay of 10^{-4}. For five-fold evaluation, hyperparameters and update budgets are fixed before test-fold evaluation, and each fold uses its final checkpoint without test-fold model selection.



(a) Simulation: ineffective intervals in a successful episode.



(b) Simulation: failure onset in a failed episode.



(c) Real robot: three synchronized views and existing annotations.

Figure 6: Annotation interfaces for recorded trajectories. Yellow marks ineffective intervals, green human intervention, and pink the segment from failure onset to the end of the trajectory. Blue indicates the current playback position. Indices refer to the reviewed sequences.

B.5 Comparison Models and Score Alignment

Detection scores increase with ineffectiveness; policy scores favor positive conditioning (Table 7). Let $q_k = f_{k+1}$ be ECR's transition logit and p_k a baseline's progress at o_k. For transition detection, progress differences use a lag of $\nu = 10$ steps in simulation and $\nu = 25$ steps on real data.

Table 7: Action-aligned detection readouts and policy-conditioning interfaces. A progress decrease ends at the observation resulting from action a_k.

Method	Scalar readout	Detection score	Policy score
RynnV.	Negative remaining time	$p_{k+1-\nu} - p_{k+1}$	$p_j - p_k$
RoboD.	Fused progress	$p_{k+1-\nu} - p_{k+1}$	$p_j - p_k$
RoboM.	Expected progress	$p_{k+1-\nu} - p_{k+1}$	$p_j - p_k$
SVM	$q(o_k, a_k)$	$-q(o_k, a_k)$	$V_j - V_k - h/D$
ECR	$q_k = f_{k+1}$	$-q_k$	$V_j - V_k - h/D$

Each value-based row uses its corresponding fitted value model. Temporal endpoints are defined in Eq. 15.

RynnValue and Robometer. We use RynnValue-8B and Robometer-4B (Huang et al., 2026; Liang et al., 2026). Inference uses FP32, the instruction, and four primary-camera frames from the causal prefix: $\lfloor rk/3 \rfloor$, $r = 0, 1, 2, 3$, allowing early repeats. RynnValue limits image sides to 320 pixels and reports negative completion time in seconds, without per-episode min–max normalization. Robometer uses the last of four temporal slots: $p_k = \sum_{r=0}^{9} (r/9) w_{k,r}$, with ten-bin probabilities supplied directly or obtained by softmax.

Robo-Dopamine. We use Robo-Dopamine 2.0-8B (Xu et al., 2026b) with fixed model parameters across evaluation folds. We convert its progress estimates to detection and conditioning scores using Table 7.

SVM and ECR. Success Visitation Matching (Tsao et al., 2026) trains a state–action discriminator from episode outcomes; ECR scores completed transitions. Detection uses negated raw logits. SVM uses 1,000 updates per task and fold in both simulation and real-world experiments. ECR follows Table 6. Training folds retain demonstrations and fit normalization; copies of each original trajectory share a fold.

B.6 Value Learning and Policy Conditioning

ECR reassigns distributional value targets; RynnV., RoboD., and RoboM. condition policies directly on progress estimates fixed during policy training without additional value fitting.

RECAP baseline targets. RECAP uses a remaining-length cost with an additional penalty of D for failed episodes. Targets are the negative cost divided by $2D$ and clipped to $[-1, 0]$. ECR instead uses the local-credit targets in Eqs. 7 and 9, with the same clipping range.

Distributional prediction. The value model takes current multi-view RGB images, task text, and validity masks, without actions or proprioception. Projected SigLIP 2 tokens and text form a Gemma-3-270M prefix; a Gemma value expert with a learned classification token predicts probabilities π_i over 201 atoms $z_i = -1 + i/200$. Both backbones are trainable during value fitting. Targets linearly interpolate between adjacent atoms, with unit mass on exact matches. Training minimizes batch-mean cross entropy against these fixed weights; inference returns $V(o) = \sum_i z_i \pi_i$.

Temporal scores and positive conditions. Set $j = \min(k + N, T)$ and $h = j - k$, with $N = 20$ in simulation and 50 on real data. The temporal scores are

$$\begin{aligned} A_k^{\mathrm{ECR/SVM}} &= V(o_j) - V(o_k) - h/D, \\ A_k^{\mathrm{RECAP}} &= V(o_j) - V(o_k), \\ A_k^{\mathrm{progress}} &= p_j - p_k, \end{aligned} \tag{15}$$

Table 8: Distributional value model and optimization settings.

Setting	Specification
Visual input	Two 224×224 RGB views in simulation; three real-robot views.
Value expert	4 layers; width 128; feed-forward width 448; 1 query and 1 key/value head; attention-head dimension 256.
Output	201 atoms spanning $[-1, 0]$; two-hot targets.
Updates and batch	1,000 updates; 4 devices $\times$ 16 samples $\times$ 4 accumulation steps, effective batch 256.
Learning rate	500-step warmup; constant 5×10^{-5} for backbones and 10^{-4} for expert/head.
Regularization	Weight decay 10^{-10}; gradient norm clip 1.
Precision and sampling	BF16; frames sampled in proportion to their frequency in selected trajectories.

Each value-based method uses its fitted model. All methods use the same truncated endpoint, without rescaling short intervals by N/h. Valid scores require $h > 0$ and finite endpoints.

The task-wise 30% selection setting assigns one condition to the entire chunk starting at k. ECR-Sign directly uses $\mathbf{1}\{f_{k+1} \geq 0\}$. Appendix C.3 defines the separate ranking diagnostic.

Conditional flow matching. The RECAP actor trains with positive and negative task-prompt conditions and deploys with the positive condition. For a normalized, padded chunk $a \in \mathbb{R}^{H \times 32}$, sample $\epsilon \sim \mathcal{N}(0, I)$ and $\tau = 0.999\beta + 0.001$, $\beta \sim \mathrm{Beta}(1.5, 1)$:

$$x_\tau = (1 - \tau)a + \tau\epsilon, \qquad \ell_{\mathrm{FM}} = \frac{\|v_\theta(x_\tau, o, c, \tau) - (\epsilon - a)\|_F^2}{32H}. \tag{16}$$

The reduction includes action padding. RECAP averages ℓ_{FM} across examples; IQL uses continuous weights without binary conditions (Appendix E). Euler integration generates chunks, followed by unnormalization and selection of physical action dimensions.

Table 9: Actor optimization settings. The real-world IQL configuration is shared by Store and Insert.

Setting	Real-world IQL	Real-world conditional	LIBERO conditional
Physical action	14D absolute joint	14D absolute joint	7D delta
Chunk horizon H	25	50	10
State representation	Discretized	Discretized	Continuous
Actor initialization	$\pi_{0.5}$ base	$\pi_{0.5}$ base	$\pi_{0.5}$ base
Updates	40,000	40,000	40,000
Global batch/devices	32/4	32/4	32/4
Warmup updates	1,000	1,000	1,000
Peak learning rate	2.5×10^{-5}	2.5×10^{-5}	2×10^{-5}
Final learning rate	2.5×10^{-6}	2.5×10^{-6}	2×10^{-5}
Schedule	Cosine	Cosine	Constant

All actor configurations use AdamW with $\beta = (0.9, 0.95)$, $\epsilon = 10^{-8}$, weight decay 10^{-10}, and gradient norm clip 1. Both real-world configurations use parameter EMA with decay 0.99. LIBERO replans after five executed actions.

C Evaluation Protocols

C.1 Closed-Loop Policy Metrics

For method m and task q, let n_{mq} be the trial count, Y_i the success indicator, and d_i the measured duration of a successful episode. Write $s_{mq} = \sum_i Y_i$ for the number of successes and $t_{mq}^{\mathrm{succ}} =$

$\sum_{i:Y_i=1} d_i$ for their total duration. With failure budget b_q in seconds, the metrics are

$$\mathrm{SR}_{mq} = 100\,\frac{s_{mq}}{n_{mq}}, \qquad \overline{d}^{\,\mathrm{succ}}_{mq} = \frac{t^{\mathrm{succ}}_{mq}}{s_{mq}},$$
$$\mathrm{Speedup}_{mq} = \frac{\overline{d}^{\,\mathrm{succ}}_{\text{BC-S},q}}{\overline{d}^{\,\mathrm{succ}}_{mq}}, \qquad \mathrm{TPH}_{mq} = \frac{3600\,s_{mq}}{t^{\mathrm{succ}}_{mq} + (n_{mq} - s_{mq})b_q}. \tag{17}$$

Speedup compares mean successful-episode durations relative to BC-S and is defined for task–method pairs with at least one success. Budget-accounted throughput charges every failed attempt the full task budget and excludes reset, placement, and recovery outside the recorded episode.

Timing and budgets. In simulation, timing starts at the first policy action after initial settling and ends at the first action satisfying success. Duration is the executed policy-step count divided by 10 Hz; failures use the full step budget. Real-world speedup uses successful-episode recording durations, measured from the first action–observation timestamp to the terminal-observation timestamp and including waits within that interval. Real-robot hard limits are 1,600, 1,800, 1,500, and 2,500 action frames for Put, Store, Insert, and Fasten. Recording rates vary across real-world episodes. We therefore convert each task's frame limit using approximately 0.045 seconds per frame to define its failure budget. These charges are nominal budgets, not measured durations of individual failed episodes; successful-episode durations use actual timestamps.

Table 10: Task budgets used for failed-attempt accounting. Simulation budgets correspond to policy steps at 10 Hz; real-world charges approximate the frame limits at 0.045 seconds per frame.

Simulation task	Budget (s)	Real-world task	Failure charge (s)
Spatial	30.0	Put	72.0
Object	50.0	Store	81.0
Goal	50.0	Insert	67.5
Long	80.0	Fasten	112.5

Aggregation. Simulation evaluation uses 100 episodes per task and method, with 50 episodes per evaluation seed. Each simulation task–method pair uses one checkpoint from a single training run. We average success rates and throughput arithmetically across tasks, and speedups geometrically. Rounding follows aggregation.

C.2 Transition Discrimination

For AP and AUROC, the positive class comprises ineffective autonomous transitions, and ECR uses $-f_t$ as its detection score. AP sums precision weighted by recall increments at distinct score thresholds; AUROC assigns half credit to tied positive/negative pairs. All methods use identical action identities and the readouts in Table 7. Successful- and failed-episode AUROC evaluate discrimination while holding episode outcome fixed.

We use five folds partitioned by trajectory, retraining SVM and ECR in each fold on the remaining trajectories. Within each fold, successful- and failed-episode AUROC pool eligible actions from the corresponding trajectory subsets. We average fold metrics within each task, then average the four task means within each domain. Task-level AP error bars show the sample SD across the five folds. AP requires ineffective examples, and AUROC requires both classes; undefined required fold metrics yield an N/A macro-average.

RynnV., RoboD., and RoboM. use fixed model parameters across evaluation folds.

Action alignment. Labels are aligned to the scored action: transition u_t corresponds to action a_{t-1}. Failed-rollout actions from a_{z^*-1} onward are ineffective. Human-controlled actions are excluded from transition discrimination.

C.3 Credit Contamination

Within each task's eligible pool $\mathcal{E}_q$, select the $K = \lfloor p|\mathcal{E}_q|/100 \rfloor$ highest conditioning scores, breaking ties with a fixed random permutation after trajectory/action ordering. Average task statistics equally within each domain. This fixed-budget ranking diagnostic is separate from training-label construction.

For ineffective-action indicators h_i and selected unique action positions $\mathcal{S}_p$, define

$$C(p) = 100 \frac{\sum_{i \in \mathcal{S}_p} h_i}{|\mathcal{S}_p|}. \tag{18}$$

Each selected action position is counted once.

Credit analysis includes demonstration, human-controlled, and autonomous actions. Transition discrimination uses autonomous actions with the required temporal context. Within each analysis, all methods are evaluated on the same action positions.

D Additional Results and Analysis

D.1 Effectiveness by Task

Figure 7 and Tables 11–12 report task-level discrimination, averaging five trajectory folds and rounding to one decimal place.

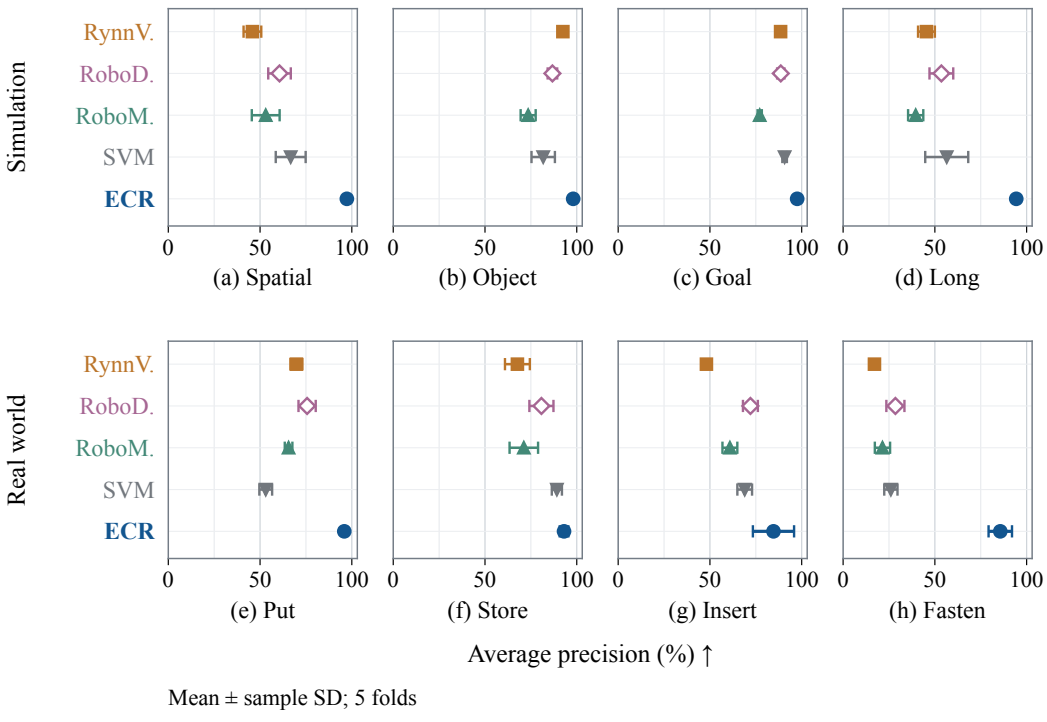


Figure 7: Task-level AP for ineffective-action detection: mean and sample SD over five trajectory folds. SVM and ECR exclude each test fold during fitting.

Table 11: Transition-effectiveness metrics by task in simulation.

	Spatial				Object			
Method	AUROC	AP	Succ.	Fail.	AUROC	AP	Succ.	Fail.
RynnV.	64.6	45.8 ± 4.8	60.4	66.5	82.5	92.4 ± 1.2	82.1	85.2
RoboD.	58.3	60.6 ± 6.1	38.9	67.9	74.3	86.7 ± 2.6	74.0	80.8
RoboM.	59.8	53.1 ± 7.6	52.8	64.0	56.8	73.5 ± 4.1	56.6	57.1
SVM	65.3	66.7 ± 8.1	60.7	65.7	67.6	81.7 ± 6.4	70.0	55.0
ECR	98.2	$\mathbf{97.3} \pm 0.6$	96.5	99.7	97.0	$\mathbf{98.1} \pm 0.7$	97.3	95.7
	Goal				Long			
Method	AUROC	AP	Succ.	Fail.	AUROC	AP	Succ.	Fail.
RynnV.	70.7	88.5 ± 2.8	70.3	73.0	66.6	45.4 ± 4.6	68.0	64.1
RoboD.	69.1	88.6 ± 2.0	69.4	63.7	72.5	53.5 ± 6.5	71.8	75.1
RoboM.	55.2	77.1 ± 1.2	55.2	56.0	60.4	39.5 ± 4.1	61.0	59.6
SVM	79.1	90.7 ± 1.2	83.9	50.7	66.3	56.5 ± 11.7	68.2	59.3
ECR	96.2	$\mathbf{97.5} \pm 1.0$	97.0	93.1	97.0	$\mathbf{94.3} \pm 1.0$	98.5	96.2

Means are percentages; AP is mean ± sample SD (percentage points), $n = 5$ trajectory folds. Succ./Fail.: AUROC within successful/failed trajectories.

Table 12: Transition-effectiveness metrics by task in the real world.

	Put				Store			
Method	AUROC	AP	Succ.	Fail.	AUROC	AP	Succ.	Fail.
RynnV.	72.4	69.8 ± 3.2	72.0	72.8	50.1	67.7 ± 6.8	52.0	26.7
RoboD.	76.1	75.6 ± 4.7	72.6	80.8	71.3	80.7 ± 6.6	70.8	75.9
RoboM.	62.7	65.5 ± 2.1	60.9	65.0	55.9	71.2 ± 7.8	55.4	61.6
SVM	57.3	53.1 ± 3.5	55.4	55.6	81.7	89.2 ± 2.8	80.1	55.9
ECR	98.1	$\mathbf{95.9} \pm 1.1$	98.3	97.6	93.2	$\mathbf{93.1} \pm 2.4$	93.4	91.8
	Insert				Fasten			
Method	AUROC	AP	Succ.	Fail.	AUROC	AP	Succ.	Fail.
RynnV.	42.5	48.1 ± 2.5	43.8	40.7	45.3	17.1 ± 2.9	45.9	42.9
RoboD.	68.4	72.1 ± 4.1	63.9	73.6	65.7	28.5 ± 4.9	65.1	67.2
RoboM.	59.1	60.9 ± 4.1	57.9	61.4	55.3	21.4 ± 4.2	56.4	52.5
SVM	60.5	69.0 ± 4.0	59.3	60.2	61.5	26.0 ± 3.6	61.1	55.6
ECR	86.7	$\mathbf{84.6} \pm 11.2$	88.8	83.6	96.1	$\mathbf{85.6} \pm 6.4$	97.4	93.6

Means are percentages; AP is mean ± sample SD (percentage points), $n = 5$ trajectory folds. Succ./Fail.: AUROC within successful/failed trajectories.

ECR has the highest task-mean AP on all eight tasks. In the real world, its AP ranges from 84.6% on Insert to 95.9% on Put; the corresponding fold standard deviations are 11.2 and 1.1 percentage points. The fold variation is larger on Insert, despite its high mean discrimination performance.

D.2 Qualitative Analysis of Transition Scores

Figures 8–11 show four trajectories excluded from scorer training. They plot native value/progress/logit outputs on method-specific scales; quantitative detection uses the aligned readouts in Appendix B.5. Salmon shading marks annotated ineffective intervals, blue shading marks human control, and red dashed lines mark annotated failure onsets. Both successful examples include human corrections. The displayed baseline names correspond to RynnV., RoboD., and RoboM. in the quantitative tables.

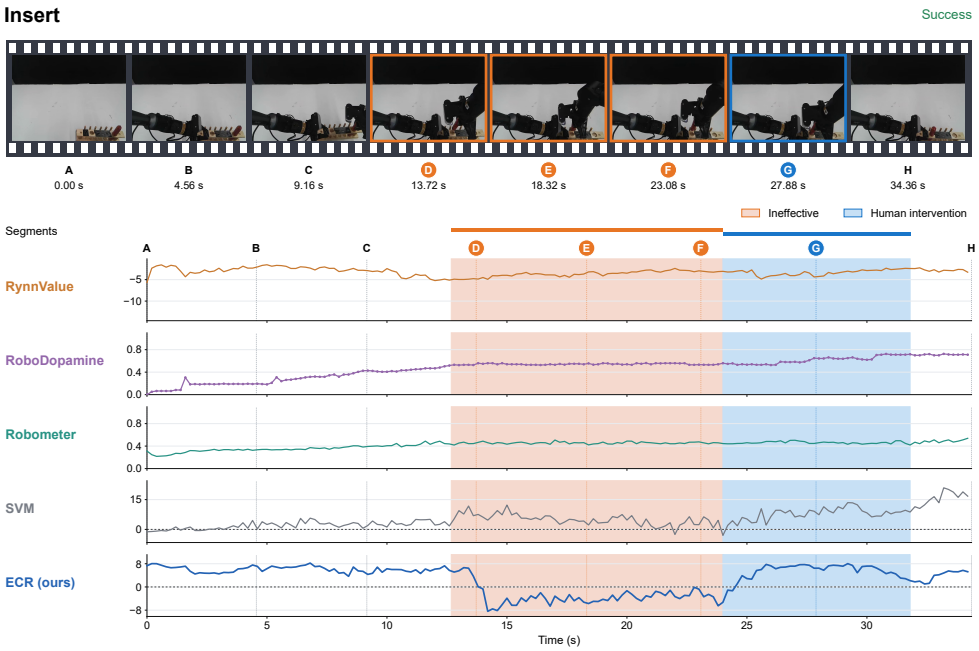


Figure 8: **Insert: successful rollout.** At A–C (0–9.16 s), the grippers approach the sausage and slot fixture. During D–F (13.72–23.08 s), manipulation near the slots coincides with the annotated ineffective interval: ECR becomes negative, while RoboD. and RoboM. plateau and SVM remains mostly positive. RynnV. drifts upward through this interval. During human control at G (27.88 s), ECR returns positive; the final frame H (34.36 s) shows the sausage upright in the fixture after the correction.

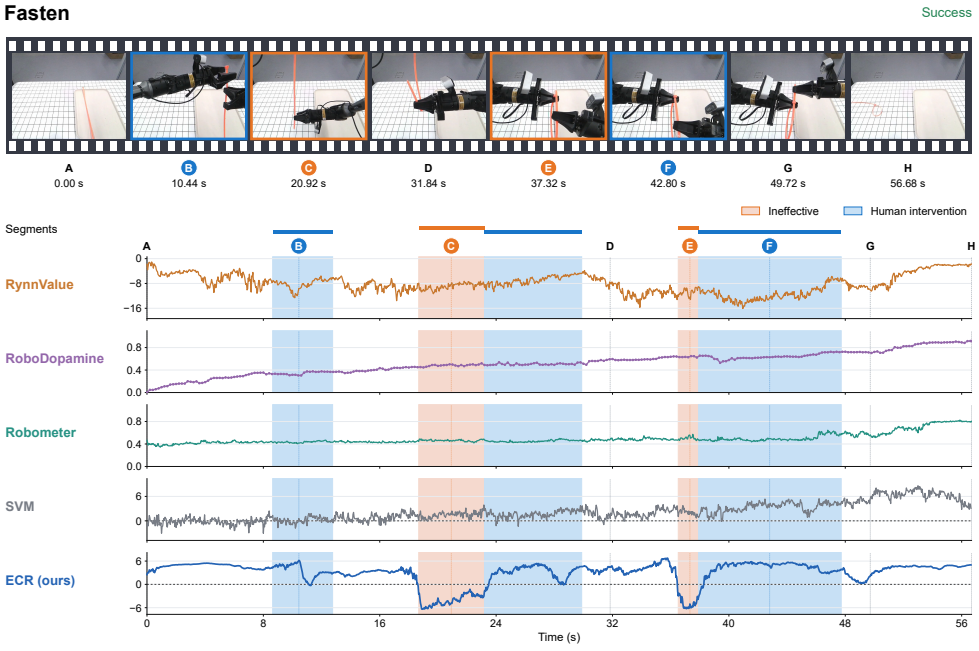


Figure 9: **Fasten: successful rollout.** At C (20.92 s) and E (37.32 s), the grippers manipulate the hanging tie and bring its ends together within two annotated ineffective intervals. ECR becomes negative in both intervals and recovers during subsequent human control, including F (42.80 s). RoboD. continues its overall rise, RoboM. stays nearly flat, and SVM remains mostly positive. At H (56.68 s), the looped tie lies on the table, separating eventual task completion from the earlier ineffective attempts.

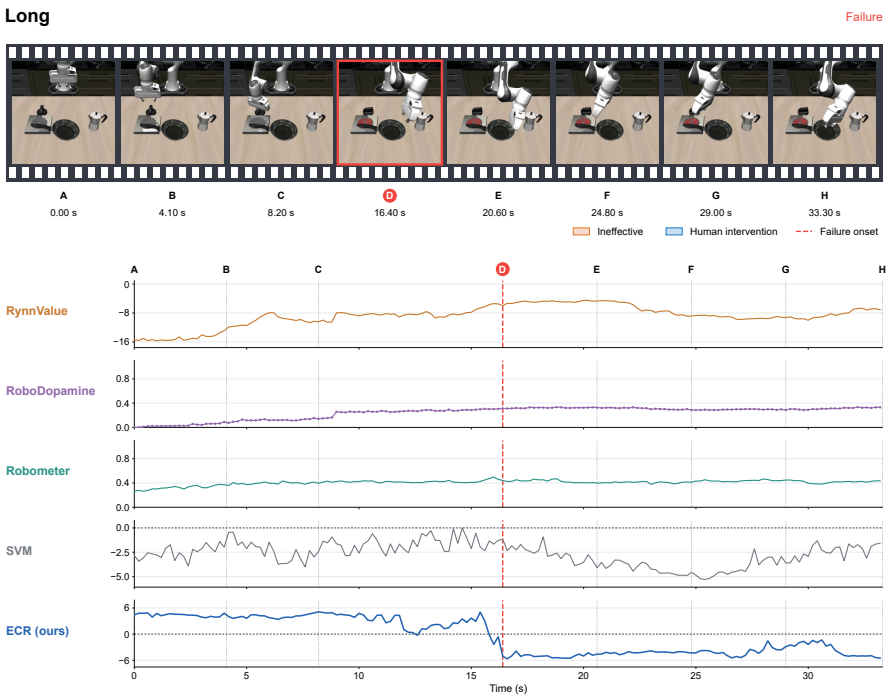


Figure 10: **Long: failed rollout.** During A–C (0–8.20 s), the gripper approaches the stove control; the burner is visibly on by D (16.40 s). ECR is mostly positive during this initial progress, then remains negative after the annotated onset at D. During E–H (20.60–33.30 s), the gripper moves around the stove and pan while the moka pot remains beside them. RynnV. initially rises after D; RoboD. and RoboM. largely plateau, while SVM is negative in both phases.

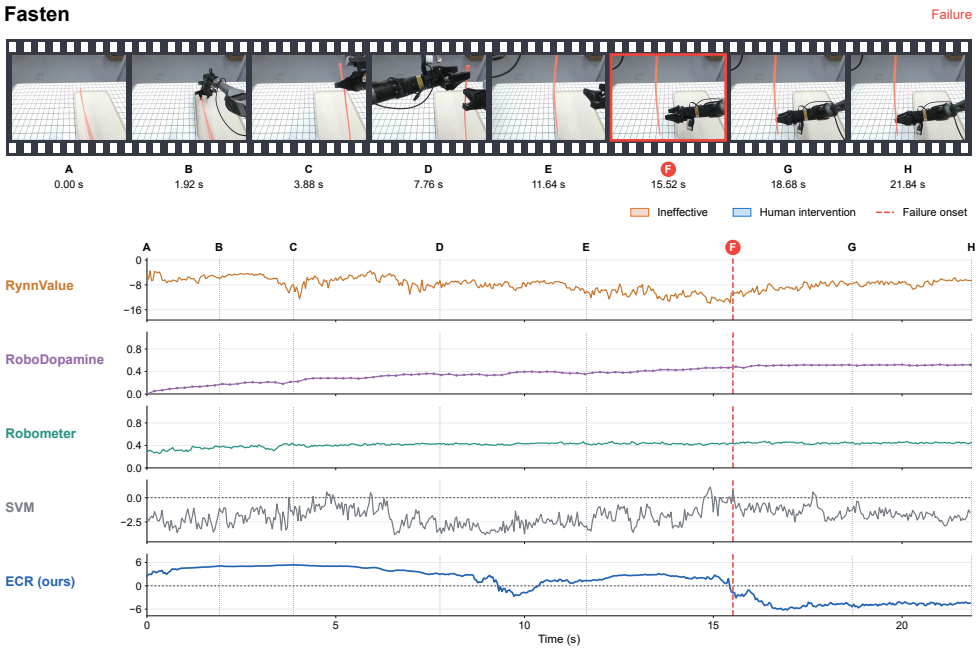


Figure 11: **Fasten: failed rollout.** During A–D (0–7.76 s), the grippers lift and reposition the cable tie, with ECR initially positive. After the annotated failure onset at F (15.52 s), the tie remains extended through H (21.84 s), and ECR stays negative. RoboD. continues to increase, RoboM. remains nearly flat, and RynnV. rises again after F; SVM fluctuates mostly below zero across both early and late phases.

D.3 Task-Level Credit Contamination

Figure 4 in the main text reports domain-level contamination. Here, Figure 12 shows the underlying task-level variation.

At 30%, ECR has the lowest contamination on seven of the eight tasks. On Put, RoboD. is lower (0.9% versus 2.2% for ECR). For ECR, contamination grows more sharply at high selection rates on Object and Goal than on Spatial and Long.

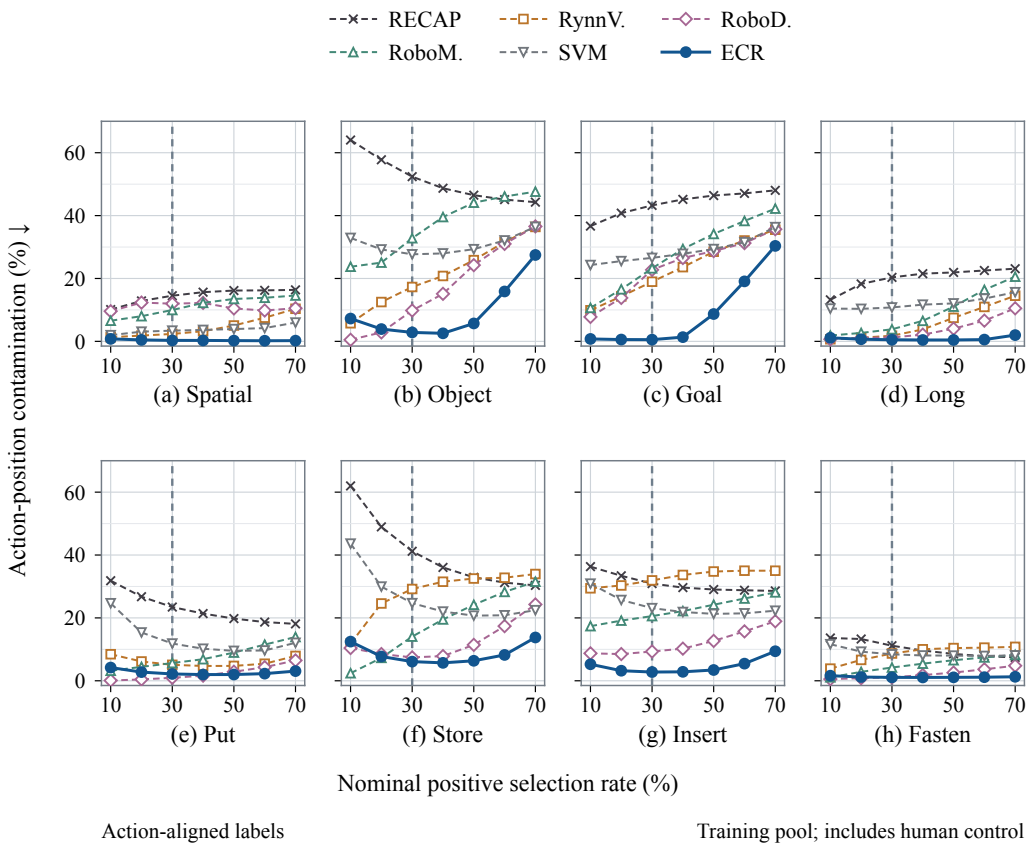


Figure 12: Task-level credit contamination at matched selection rates. The training pool includes demonstrations and human-controlled segments, with each selected action position counted once. Each curve varies the selection threshold over fixed conditioning scores. The vertical dashed line marks 30%; ECR is solid and comparison signals are dashed.

D.4 Component Ablations

The component study changes the supervision of the ECR estimator while retaining its downstream policy interface. Within each task and fold, scorer variants share the trajectory partition, input features, anchor and onset labels, architecture, optimizer, update budget, and seed. Direct removes the additional event loss, MAP targets replace the onset distribution with a point assignment when constructing the targets, and Stop-gradient retains soft targets while detaching their gradient path. ECR retains both the soft assignment and the gradient through it. All retained loss terms have unit weight.

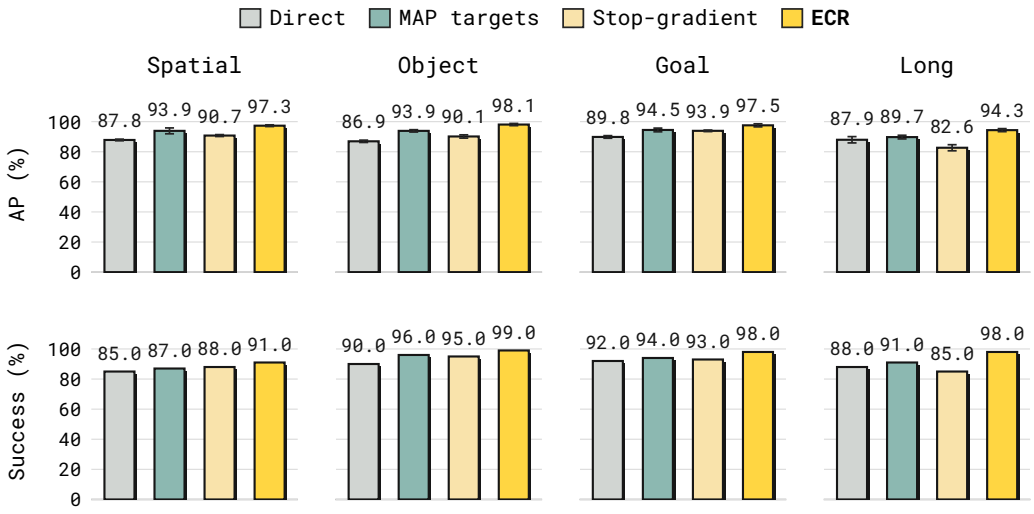


Figure 13: Task-level component ablations in simulation. Top: AP for ineffective-action detection, with sample SD over five trajectory folds. Bottom: closed-loop policy success rate. Values are shown to one decimal place. Error bars apply only to AP.

Figure 13 shows that ECR attains the highest AP and policy success rate on all four tasks. Relative to MAP targets, mean AP increases from 93.0% to 96.8%, and mean success rate from 92.0% to 96.5%.

E Offline and Online Reinforcement Learning

We evaluate ECR reward correction on Store and Insert. IQL comparisons share the offline data and optimization budget; RL Token comparisons share the initial policy and online interaction budget. Per task, IQL uses 50 demonstrations, 30 successful rollouts, and 10 failed rollouts. RL Token uses 50 demonstrations for representation learning, 10 successful trajectories for warmup, and one collection round of 30 trajectories (20 successful and 10 failed).

Reward correction. Equation 11 specifies reward correction for an individual transition. In IQL and RL Token, we average its ECR correction term over the executed, non-padding actions in each chunk and add the mean once to the task reward aggregated by the corresponding algorithm. Both algorithms use $\lambda_r = 0.02$ and $B = 5$.

E.1 Offline IQL

We follow the IQL (Kostrikov et al., 2021) policy-learning setup of RynnValue (Huang et al., 2026): expectile value regression, twin-Q temporal-difference learning, and advantage-weighted flow matching for $\pi_{0.5}$. Store and Insert use the same settings. Each aggregate transition spans up to 25 executed actions, with task reward zero upon successful termination and -1 otherwise. We use expectile 0.8, discount 0.99 per chunk, and target-update coefficient 0.005. ECR supplies the shared reward correction described above. Deployment uses the learned policy alone.

E.2 RL Token Initialization

RL Token (Xu et al., 2026a) initialization follows the RLinf implementation. Stage 1 combines reconstruction of detached VLA features with the flow-matching loss. Stage 2 freezes the VLA and token representation and trains an actor with twin critics. Conditioned on the RL token, proprioception, and VLA reference chunk, the actor directly predicts normalized actions through a tanh output. Its objective combines Q maximization with cloning toward the VLA reference, or the ex-

ecuted action during human control. Table 13 lists the actor–critic settings used in our RL Token experiments.

Table 13: RL Token actor–critic settings.

Setting	Value
Actor and twin critics	Three hidden layers of 256 units
Learning rate	3×10^{-4}
Per-action discount; target update	0.96; 0.005
Critic-to-actor update ratio	$4 : 1$
Q / cloning / entropy weights	0.1 / 5 / 0
Reference-input dropout; action std.	0.5; 0.002

E.3 Single-Round Online Adaptation and Evaluation

After the single collection round, we update the ECR estimator and use its completed-transition scores to correct the critic reward. RL Token retains the success-terminal task reward of 1 (zero otherwise) and RLinf's discounted chunk aggregation, followed by the shared mean ECR correction described above. Each final policy is evaluated over 15 autonomous trials per task, with timeouts counted as failures. Table 5 reports the results and equal-task means.