RAP-WAM: LEARNING WORLD-ACTION MODELS THROUGH PREDICTIVE REHEARSAL AND ACTION PERTURBATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent world action models draw on video generation priors to jointly predict future observations and robot actions. Adapting these priors to control requires determining how future predictions should guide policy learning. RAP-WAM combines predictive Rehearsal and Action Perturbations (RAP) with future-supervised queries and Coupled Action Projection for policy learning. Predictive rehearsal trains on model-generated inputs while retaining recorded targets, and accumulates recurring errors into a spatial prior. We perturb actions and measure changes in video predictions. Action-response-based spatiotemporal credit assignment combines these responses, prediction error, and the rehearsal prior to allocate visual supervision; spatially aggregating the responses determines supervision intensity across future action steps. No separate scoring network or deployment-time probe is required. The base learns from a video-pretrained backbone without intermediate pretraining on external embodied data. We evaluate the fixed base and its agent-refined, task-conditioned system. On 42 RoboDojo tasks, their average success rates are 12.85% and 15.62%, with Scores of 17.86 and 20.59. Across 15 real-robot tasks, the complete system achieves 89.00% task-macro-average success, exceeding the strongest of five baselines by 37.67 percentage points. Across 315 fixed-output simulation windows, 45.21% of response-top20 positions fall outside error-top20, revealing information beyond prediction error.

1 INTRODUCTION

Generative robot policies combine visual representations with continuous action generation, as exemplified by Diffusion Policy and π_0 (Chi et al., 2025; Black et al., 2024). The π series has since advanced toward open-world generalization in $\pi_{0.5}$ (Physical Intelligence et al., 2025) and multimodal steerability and compositional task generalization in $\pi_{0.7}$ (Intelligence et al., 2026). World action models additionally predict future observations: UWM couples video and action diffusion (Zhu et al., 2025); DreamZero builds on pretrained video diffusion (Ye et al., 2026b); X-WAM incorporates multi-view RGB-D prediction and asynchronous denoising (Guo et al., 2026).

OpenWAM studies backbone inheritance, world action interaction, and embodied-pretraining scale (Wang et al., 2026). We focus on adapting video priors directly to target-task demonstrations: which visual locations and future action steps should receive additional training emphasis? Earlier work offers complementary perspectives. Prioritized replay changes how frequently experience transitions are reused (Schaul et al., 2016), while value-aware model learning incorporates decision structure into prediction losses (Farahmand et al., 2017). These motivate distinguishing prediction accuracy from policy relevance; allocating supervision across visual locations and future action steps in joint video–action models remains an open problem.

We introduce RAP-WAM, coupling predictive rehearsal with action perturbations to guide representations and supervision. Its central signal is the video prediction's response to perturbations of the model's action input. Prediction error measures target mismatch, whereas this response reveals which visual locations and future times depend more strongly on the action representation. Combining them distinguishes prediction difficulty from action dependence, providing a shared basis for prioritizing visual and action prediction.

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.

The framework couples three designs. Recorded future observations supervise predictive queries, giving action generation access to future-oriented features. Predictive rehearsal conditions subsequent demonstration segments on model-predicted observations while retaining recorded targets (Huang et al., 2025; Bengio et al., 2015), and accumulates recurring errors into a spatial prior. The credit mechanism integrates this prior with current error and action response: spatial maps guide visual supervision, while spatially pooled response weights future action steps. Together, these designs turn existing demonstrations into representation targets, rehearsed inputs, and adaptive supervision without extra region annotations or a separate scoring network.

Our contributions are as follows:

- We develop RAP-WAM to adapt video priors through future-supervised action representations and predictive rehearsal: recorded futures shape action-generating features, while rehearsal exposes the policy to observations affected by its own prediction errors.
- We design an action-response-driven spatiotemporal credit mechanism: action response, current prediction error, and a historical error prior guide visual supervision, while spatially pooling the same response allocates supervision across future action steps.
- We integrate execution-guided post-training with task-specific policy retention, using evaluation feedback to refine demonstration sampling and preserve complementary specializations.

2 Related Work

Generative control and world action models. Beyond the action and world action formulations introduced above, UniPi derives actions from video plans through inverse dynamics (Du et al., 2023). Motus and MotuBrain use multimodal expert architectures (Bi et al., 2026; MotuBrain Team et al., 2026). GigaWorld-Policy emphasizes action-centered joint learning (Ye et al., 2026a); AHA-WAM uses asynchronous video planning (Cai et al., 2026), while Fast-WAM retains video co-training without test-time future imagination (Yuan et al., 2026). Our focus is how existing target demonstrations supervise a joint policy. Further context on action interfaces and adaptive supervision appears in Appendix I.

Post-training for embodied policies. RECAP improves $\pi_{0.6}$ into $\pi^*_{0.6}$ through reinforcement learning from demonstrations, autonomous experience, and expert corrections (Intelligence et al., 2025). SIME selects interaction segments (Jin et al., 2025); RISE optimizes policies through imagined outcomes (Yang et al., 2026a). ENPIRE supports agent-directed experiments (Xiao et al., 2026), while ASPIRE retains validated program repairs (Lu et al., 2026). Our agent instead guides world action policy training on existing demonstrations; response-based credit allocates supervision, and evaluation retains complementary task specialists. The reported gains concern this combined procedure.

3 RAP-WAM: Predictive Rehearsal and Action Perturbations

Figure 1 summarizes RAP-WAM: a future-supervised action interface, predictive rehearsal, and response-weighted supervision on target-task demonstrations. We separately evaluate post-training and task-specific selection, with full protocols in the appendix.

3.1 A future-supervised action interface

A shared video–action predictor does not automatically give its action features an explicit representation of the future. We introduce a future-supervised interface within that predictor; standard conditioning and flow matching (Lipman et al., 2023) are described in Appendix C.1. Recent interfaces supervise observed geometry (Li et al., 2026) or latent foresight (Ma et al., 2026); ours aligns queries to recorded future video latents. We append K learned query tokens Q to video V, noisy-action A, and observed-state S tokens, forming $[V, A, S, Q]$. Video tokens contain the observed prefix and noisy future latents. Let z be recorded-future latents and h_k the final transformer representation of query k. The alignment target is

$$r_k = \mathrm{sg}[P_t \,\mathrm{Pool}_G(z)_k]\,, \qquad \mathcal{L}_{\mathrm{align}} = \frac{1}{K}\sum_{k=1}^{K}[1 - \cos(P_q h_k, r_k)]. \tag{1}$$

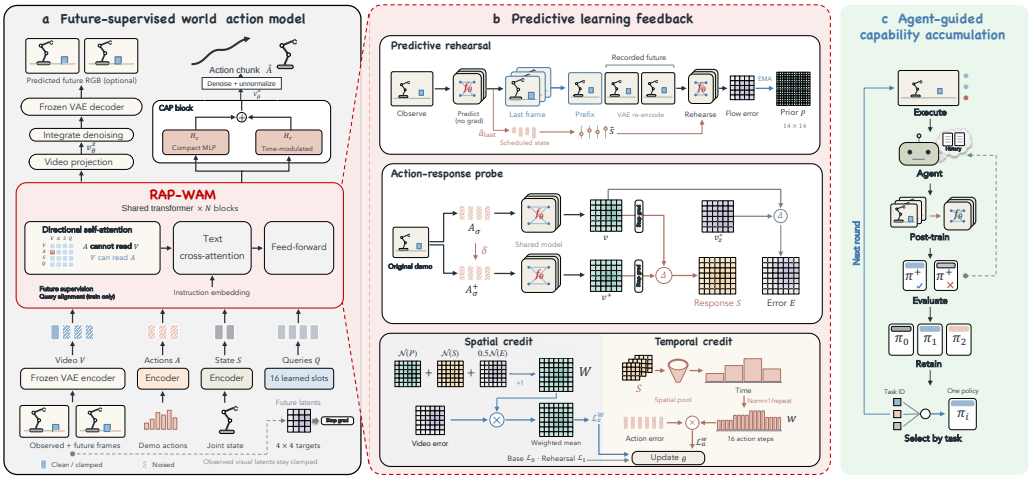


Figure 1: **Predictive learning and agent-guided policy refinement.** **a**, future latents supervise query alignment during training. CAP predicts action flow-matching velocity v_θ^a; denoising and unnormalization yield action chunk $\hat{A}$. **b**, rehearsal updates spatial prior P. The matched probe produces video-denoising velocity predictions v and v^+ under original and perturbed action inputs, sharing model, condition, video noise, and diffusion level. Δ denotes channel-mean squared discrepancy: S compares v^+ with v, and E compares v with flow-matching target v_z^*. Spatial weights combine separately normalized sources, $W = 1 + \mathcal{N}(P) + \mathcal{N}(S) + 0.5\mathcal{N}(E)$. Temporal weights use $1 + \mathcal{N}_t(\mathrm{mean}_{hw}\, S)$, repeated over action steps. Weight construction is detached. **c**, the agent proposes post-training interventions. Evaluation retains improvements by task; other tasks keep their policies. Accepted and rejected outcomes inform subsequent rounds. Each task selects one complete policy. Scenes and fields are schematic.

Here P_t, P_q project targets and queries to a common 128-dimensional space; sg stops gradients. We pool z onto a time–height–width grid $\mathcal{G} = (1, 4, 4)$, yielding $K = 16$ targets. Alignment updates P_q and query representations, while P_t remains fixed. Learned query embeddings initialize Q at each forward with action-side noise conditioning; recorded futures are needed only for training supervision. The base objective is

$$\mathcal{L}_0 = \mathcal{L}_z + \mathcal{L}_a + \lambda_q \mathcal{L}_{\text{align}}, \tag{2}$$

where $\mathcal{L}_z, \mathcal{L}_a$ are mean-squared errors against video and action flow-matching targets; we set the alignment coefficient $\lambda_q = 0.1$. To encourage actions to use the future-supervised query features, we make attention connectivity directional. Within each block, we remove direct reads from action-token query rows to video-token key columns. For their index sets $\mathcal{A}, \mathcal{V}$,

$$M_{ij} = \begin{cases} -\infty, & i \in \mathcal{A},\ j \in \mathcal{V}, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

The mask is added to attention logits. Action rows can read action, state, and predictive-query keys, but not video keys. Video rows can still read action keys, preserving the action-to-video path required by the response probe. State-mediated visual information remains accessible, so queries are not the sole visual pathway. GigaWorld-Policy also restricts future-video-to-action reads (Ye et al., 2026a). Coupled Action Projection (CAP) maps final action-token features h_a to the action flow-matching velocity v_θ^a. A compact MLP H_c and a time-modulated head H_r predict the same velocity target through their sum:

$$\begin{aligned} v_\theta^a &= H_c(h_a) + H_r(h_a, e_\sigma), \\ H_c(h) &= W_2\, \mathrm{GELU}(W_1\, \mathrm{LN}(h) + b_1) + b_2. \end{aligned} \tag{4}$$

Here H_r is the sequence head conditioned on the noise-level embedding e_σ; H_c uses a hidden width one quarter of the trunk width. Zero-initializing W_2, b_2 adds this path without initially changing the sequence-head output. Both branches learn through their sum against the same action-flow target, not separate coarse/fine labels. The MLP has no separate noise-level input, although h_a already depends on that noise level. The added capacity therefore changes the action projection without introducing a second policy or task router.

3.2 REHEARSAL UNDER PREDICTED INPUTS

Training uses recorded images and joint states; deployment instead supplies observations affected by the robot's preceding actions (Ross et al., 2011). Scheduled sampling (Bengio et al., 2015) and Self Forcing (Huang et al., 2025) motivate exposing sequence models to their own predictions. For two adjacent demonstration chunks, we generate the first chunk's future video and actions without gradients and replace the second chunk's initial observation with the last predicted frame. Scheduled state substitution optionally replaces its initial state with the last predicted absolute-joint action, first unnormalized as an action and then normalized as a state. The predicted image and recorded second-chunk future frames are jointly VAE-encoded; recorded actions remain the targets. Their video/action flow losses form $\mathcal{L}_1$, without an extra query-alignment loss. This *rehearsal* trains on predicted inputs while retaining recorded supervision; its schedules are in Appendix C.1. An initially zero spatial prior $\mathcal{P}$ accumulates the resulting prediction errors across updates. We average detached second-chunk video errors over channels, batch and future time, pool them onto a $g_P \times g_P$ grid to obtain e^{roll}, and update

$$\mathcal{P} \leftarrow \rho\mathcal{P} + (1-\rho)e^{\mathrm{roll}}. \tag{5}$$

We use grid width $g_P = 14$ and decay $\rho = 0.9$. The first valid error map initializes $\mathcal{P}$ directly; later updates use the moving average above. Before weighting the video loss, we resize $\mathcal{P}$ to the future-latent spatial grid and repeat it across future latent times. The predicted joint action approximates the next state, without simulator interaction or new corrective labels. Unlike DAgger, this does not relabel visited states with an expert (Ross et al., 2011); it also differs from imagined-return optimization in RISE (Yang et al., 2026a). The prior records where rehearsal predictions have repeatedly been inaccurate. The next step measures which predictions also depend on the action input.

3.3 RESPONSE-DERIVED SPATIOTEMPORAL CREDIT

Prioritized replay and focal loss allocate learning emphasis using error signals (Schaul et al., 2016; Lin et al., 2017). Our credit mechanism adds action-conditioned response and shares it across visual and action supervision. response-based credit compares two forwards of the same recorded demonstration. We draw fresh, independent noise for its future latents and actions at a shared noise level, retaining the recorded conditioning inputs, and compute a reference prediction with gradients. A second forward uses the same model, condition, video noise, and noise level but adds $\delta \sim \mathcal{N}(0, \tau^2 I)$ to the normalized noisy actions; this forward has no gradients. At future spatiotemporal position i, let v_i, v_i^+ denote the reference and perturbed video velocities with C channels. The target is $v^*_{z,i} = \epsilon_{z,i} - z_i$, where z_i is the clean recorded-future latent and $\epsilon_{z,i} \sim \mathcal{N}(0, I)$ is its sampled noise, consistent with $z_\sigma = (1-\sigma)z + \sigma\epsilon_z$. Here i indexes a future spatiotemporal location and $*$ denotes the supervision target. Holding these predictions constant for coefficient construction gives

$$\mathcal{S}_i = \frac{1}{C}\|v_i^+ - \mathrm{sg}(v_i)\|^2, \qquad \mathcal{E}_i = \frac{1}{C}\|\,\mathrm{sg}(v_i) - v^*_{z,i}\|^2. \tag{6}$$

We set the perturbation scale $\tau = 0.1$. $\mathcal{E}$ measures squared prediction error; $\mathcal{S}$ measures squared change after the action perturbation. These are changes in model outputs, not measured physical effects of executed actions (Appendices C.2–C.3). For each future visual position, we normalize the prior $\mathcal{P}$, response $\mathcal{S}$, and error $\mathcal{E}$ separately, then combine them into a video-loss weight. Here $\mathcal{N}$ is per-sample min–max normalization over future time and spatial positions, with denominator floor ϵ_{n}:

$$\mathcal{W}_i = 1 + \alpha_P \mathcal{N}(\mathcal{P})_i + \alpha_S \mathcal{N}(\mathcal{S})_i + \alpha_E \mathcal{N}(\mathcal{E})_i. \tag{7}$$

The coefficients are nonnegative; we set $\alpha_P = \alpha_S = 1$, $\alpha_E = 0.5$, and $\epsilon_{\mathrm{n}} = 10^{-6}$. The unit floor gives every location a nonzero loss coefficient, capped at 3.5 with these settings. To weight action steps, we first average the raw response over spatial positions, then normalize these means across future time:

$$c_t = 1 + \alpha_S \mathcal{N}_t(\mathrm{mean}_{hw}\,\mathcal{S})_t, \quad w_j = c_{\lfloor j/R \rfloor}, \quad R = \lceil H_a/T \rceil. \tag{8}$$

Here H_a is the action-chunk length, T the number of future latent steps, and $\mathcal{N}_t$ min–max normalization over those T steps within each sample. Using zero-based indices, each coefficient is repeated R times and truncated to H_a actions; the reported $T = 4, H_a = 16$ setting assigns one coefficient to four consecutive actions. This branch uses the raw response before spatial min–max normalization.

The prior is constant along time, so its normalized temporal contribution is zero; $\mathcal{E}$ does not enter this branch. Hence $1 \leq w_j \leq 1 + \alpha_S$. Visual and action prediction share one response source, with separate spatial and temporal weighting. Action coefficients depend on normalized differences between future-step spatial means, not on the locations of individual response peaks.

We apply these weights to video and action prediction errors. Let i enumerate all future video locations across the batch and j all action steps across the batch. At current parameters $\bar{\theta}$, normalize the detached loss coefficients as $q_i^v = \mathcal{W}_i / \sum_\ell \mathcal{W}_\ell$ and $q_j^a = w_j / \sum_\ell w_\ell$. For channel-mean video errors e_i^z and action-dimension-mean errors e_j^a, the auxiliary objective is

$$\widetilde{\mathcal{L}}_P(\theta; \bar{\theta}) = r_P(k) \left[\lambda_z \sum_i q_i^v(\bar{\theta}) e_i^z(\theta) + \lambda_a \sum_j q_j^a(\bar{\theta}) e_j^a(\theta) \right], \tag{9}$$

where k counts training updates, W_P is the credit warmup length, and $r_P(k) = \min(1, k/W_P)$; we set $\lambda_z = 0.5$ and $\lambda_a = 0.3$. The full update uses $\mathcal{L} = \mathcal{L}_0 + b_c \lambda_c r_c(k) \mathcal{L}_1 + b_P \widetilde{\mathcal{L}}_P$, with $b_c \sim \text{Bernoulli}(p_c(k))$ and $b_P \sim \text{Bernoulli}(p_P)$. Here $r_c(k), p_c(k)$ are the rehearsal warmup and invocation schedules (Appendix C.1); $\lambda_c = 1$ and $p_P = 0.5$. The video error e_i^z equals $\mathcal{E}_i$ numerically but retains gradients. Both video and action errors reuse the reference forward. Weight construction is detached, so backpropagation updates the predictor through these errors only; no additional forward is needed for this auxiliary loss. Unlike importance sampling with corrective weights (Katharopoulos & Fleuret, 2018), this deliberately changes the auxiliary loss emphasis. Positive, bounded coefficients retain every target while regulating relative emphasis, not gradient norms or capability preservation (Appendix C.4). No separate region-scoring network or reward head is needed.

The base objective supervises predictions and query features, rehearsal substitutes predicted images and states, and credit scales individual loss terms. At deployment, the queries, directional mask, and CAP remain active during joint video-latent/action denoising. We return unnormalized actions without decoding future RGB; neither rehearsal nor credit estimation is invoked. Architecture and sampling details are given in Appendix C.1.

3.4 Post-training and task-specific policy selection

We evaluate the fixed RAP-WAM base and a refined system separately. The latter extends the learner with an experiment state $X_r = (\mathcal{B}_r, g_r, \mathcal{H}_r)$: available policies, task assignments, and intervention history. The initial repertoire contains the base. Following execution-feedback approaches (Xiao et al., 2026; Lu et al., 2026), task evaluation informs subsequent training; accepted specialists remain available rather than replacing the base on every task.

From evidence to a training intervention. At round r, the agent receives task outcomes, execution evidence, and available diagnostics, denoted $\mathcal{F}_r$. Together with history $\mathcal{H}_r$, these specify an intervention u_r: the parent checkpoint, target behaviors, demonstration allocation, and trainable parameter set. Here $\mathcal{A}$ is the agent, π_r the selected parent policy, $\mathcal{D}_r$ the resampled demonstration windows, and $\mathcal{U}_{\text{RAP}}$ the training operator producing candidate π_r^+:

$$u_r = \mathcal{A}(\mathcal{F}_r, \mathcal{H}_r), \qquad \pi_r^+ = \mathcal{U}_{\text{RAP}}(\pi_r, \mathcal{D}_r; u_r). \tag{10}$$

Agent-guided refinement yields multiple task specializations with only a few thousand fine-tuning steps. Parent policies and rejected outcomes remain available.

Replay under recorded targets. We implement the replay intervention with 50% natural replay, 20% task-balanced replay, 15% failure-boundary windows, and 15% late-progress windows. Natural replay preserves coverage; task-balanced replay equalizes task sampling. Boundary replay favors demonstration phases near progress inferred from failed executions' partial-completion scores; late replay favors the final portion of demonstrations. The score-to-progress mapping is a sampling heuristic, not a measured failure time. Training retains recorded images and action targets, uses the RAP-WAM objectives with separate backbone and interface learning rates, and evaluates intermediate checkpoints. Appendix H specifies eligibility, fallback rules, optimizer settings, and budgets.

Retention without averaging policies. Let $m_t(\pi) = (\text{SR}_t(\pi), \text{Score}_t(\pi))$ denote success rate and mean partial-completion score from the same execution record. A task enters $\mathcal{C}_r$ if the candidate

Table 1: **RoboDojo comparison: Score / SR (%).** Pretr.: external embodied pretraining reported for the method. Video/VLM initialization and target-task demonstrations are excluded. $*$: AHA-WAM uses robot-data pretraining for real-world experiments, but not in its default simulation setup (Appendix B). Score and SR are ranked separately per column; highest and second-highest values are **bold** and underlined, including ties. ‡: VLA supervised by a frozen video generation model during training. §: matched 42-task coverage for our RAP-WAM base and post-training system; each task is averaged over its valid evaluation episodes.

Model	Pretr.	Gen-Std	Gen-Rand	Precision	Long-Hor.	Memory	Open	Average
VLA and other action policies								
G0.5 (Liu et al., 2026)	Yes	27.72/21.44	9.20/4.22	28.25/20.42	44.12/32.25	8.61/7.33	1.73/1.58	20.23/14.88
Xiaomi-Robotics-1 (Xiaomi Robotics Team et al., 2026)	Yes	**35.65/28.00**	**11.44/6.00**	26.69/18.83	38.39/23.67	7.81/6.56	3.94/3.58	20.07/13.93
Hy-Embodied-0.5-VLA (Zhang et al., 2026)	Yes	21.98/17.00	1.57/0.00	13.81/8.00	25.74/14.92	13.37/12.11	0.65/0.58	13.07/8.80
KinRT (Yang et al., 2026c)	Yes	20.94/14.78	7.09/2.44	15.65/9.92	26.40/18.08	4.82/3.56	4.23/3.83	13.02/8.80
SimpleMemVLA (Yin et al., 2026)	No	10.38/7.11	2.33/0.78	7.42/2.92	14.58/5.50	**33.71/33.22**	0.85/0.75	12.58/9.27
Pi-05 (Physical Intelligence et al., 2025)	Yes	20.93/15.00	5.82/1.00	12.40/5.50	23.54/14.67	5.78/4.56	1.98/1.67	11.41/6.91
InternVLA-A1.5‡ (Ma et al., 2026)	Yes	16.81/12.00	3.90/2.00	15.23/10.17	23.80/13.75	4.93/3.56	1.43/1.42	11.15/7.14
VLAct (Yang et al., 2026b)	Yes	16.33/12.00	2.74/1.00	20.62/15.25	20.12/13.67	0.66/0.56	2.37/2.25	10.66/7.60
MolmoAct2 (Fang et al., 2026)	Yes	12.53/7.33	4.53/2.00	10.92/4.50	15.20/7.00	5.53/4.56	**4.75/4.42**	8.99/5.03
World-model-based policies								
OpenWAM-α (Wang et al., 2026)	Yes	33.16/25.56	8.26/4.11	18.45/9.25	34.93/25.33	10.41/9.11	1.41/1.08	17.18/11.92
X-WAM (Guo et al., 2026)	Yes	11.24/5.00	3.54/1.00	6.72/1.83	17.47/9.08	6.32/4.67	0.57/0.25	7.69/3.83
GigaWorld-Policy-0 (Ye et al., 2026a)	Yes	10.28/6.00	0.41/0.00	6.15/1.83	15.51/8.92	3.46/2.22	0.54/0.50	6.20/3.27
AHA-WAM (Cai et al., 2026)	Yes*	10.32/6.00	1.26/0.00	5.86/2.42	8.61/2.67	2.97/2.78	0.88/0.83	4.82/2.39
Fast-WAM (Yuan et al., 2026)	No	4.33/2.00	0.34/0.00	1.96/0.00	9.14/5.17	3.55/3.44	0.42/0.42	3.48/2.03
World-model-based: ours								
RAP-WAM (base)§	No	21.55/15.00	0.82/0.00	23.97/16.51	45.12/32.50	7.97/7.00	1.06/0.75	17.86/12.85
RAP-WAM + refinement§ (Ours)	No	27.82/21.00	0.93/0.00	**28.66/21.03**	**48.43/36.50**	10.32/9.33	1.16/0.75	**20.59/15.62**

improves SR over its currently assigned policy, or ties SR and improves Score; a tie in both retains the incumbent. The assignment updates as

$$g_{r+1}(t) = \begin{cases} \mathrm{id}(\pi_r^+), & t \in \mathcal{C}_r, \\ g_r(t), & t \notin \mathcal{C}_r. \end{cases} \tag{11}$$

Here $\mathrm{id}(\pi)$ is a checkpoint identifier and $g_r(t)$ denotes the selected policy identity; the corresponding checkpoint remains in $\mathcal{B}_r$. An empty $\mathcal{C}_r$ leaves the active assignment unchanged. At inference, the task identity selects one whole checkpoint, with its architecture, normalizer, and inference settings. Parameters and within-episode action fragments are not averaged. This assignment operates across policies, whereas credit coefficients weight losses inside a policy's training.

The final assignment uses the base on 14 tasks and 16 other checkpoints on 28 tasks. History retains proposals, outcomes, and acceptance decisions for subsequent interventions. This is the recursive aspect of the experimental procedure; it does not establish autonomous improvement or a causal benefit from history reuse. Selection uses observed evaluation results, so independent policy-selection tests remain necessary. The complete state update and execution rule are retained in Appendix H.1.

4 EXPERIMENTS

We evaluate the fixed RAP-WAM base and the post-trained task-conditioned system on RoboDojo, analyze their supervision signals, and report physical execution results.

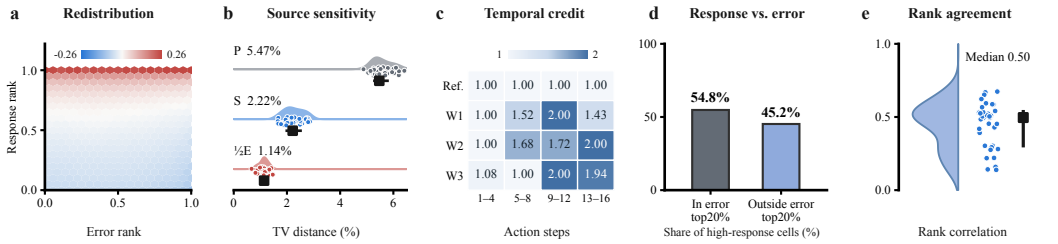


Figure 2: **Complementary response signals and spatiotemporal supervision.** **a**, signed spatial credit redistribution. **b**, sensitivity to removing prior, response, or half-error. **c**, temporal action credit in three phone-packing windows (W1–W3), with a unit reference. **d**, response-top20 membership within or outside the error-top20 set. **e**, rank correlation across 35 condition medians. Panels a, b, d, and e use the same 315 RoboDojo windows. Squares and segments indicate medians and IQRs; these fixed-output diagnostics involve no retraining.

4.1 Base and post-trained system performance

We report SR and partial-completion Score, averaged within each capability and then equally across the five capabilities (Chen et al., 2026b). Public results are taken from the RoboDojo leaderboard (Chen et al., 2026a); coverage and system accounting are specified in Appendices A and H. Across the 42 tasks, the full method achieves **15.62% average SR and 20.59 Score**, compared with 12.85% and 17.86 for the RAP-WAM base. The gains are 2.77 percentage points in SR and 2.73 Score. Precision increases from 16.51% to 21.03%, Memory from 7.00% to 9.33%, and Generalization from 7.50% to 10.50%. These gains arise in the task-conditioned system, where retained specializations contribute across different capability groups.

Long-Horizon success rises from 32.50% to **36.50%**, with Score increasing from 45.12 to **48.43** (Table 1). At the task level, tower building increases from 60% to 72%, pen-holder filling from 6% to 18%, and tic-tac-toe from 80% to 86%. The task-policy breakdown is given in Appendix D. Open success remains 0.75%, and random-Generalization success remains zero, indicating that the observed gains do not resolve broad scene-shift robustness. The candidate comparison in Appendix Figure 12 helps explain this pattern: the base has the highest aggregate performance among individual checkpoints, yet candidates improve different task subsets. Task-specific retention preserves these local gains despite regressions elsewhere. Appendix Figure 11 shows how the retained specialists divide the task set; their value is not captured by ranking whole-policy averages alone.

4.2 Action response changes the locations emphasized during training

Figure 2a–b uses 315 RoboDojo windows across 35 conditions. Holding model predictions fixed, we separately remove the historical error prior, action response, or current-error contribution and renormalize the remaining weights to compare their effects on supervision allocation (Appendix E). Response redistributes spatial emphasis; the median shares of normalized coefficients redistributed are 5.47%, 2.22%, and 1.14% for $\mathcal{P}, \mathcal{S}, 0.5\mathcal{E}$, respectively. The rehearsal prior produces the largest redistribution. These distances combine magnitude and spatial shape, not additive contributions to task success.

Figure 2c shows temporal supervision in three phone-packing demonstration windows. Spatially pooled action responses distinguish future action steps without an extra scoring network. Removing the response leaves all coefficients at one: $\mathcal{P}$ is time-constant and this branch excludes $\mathcal{E}$. Appendix E details reconstruction and four-action grouping.

Figure 2d–e shows complementary rankings: on average, *45.21% of response-top20 positions lie outside the error-top20 set*. The condition-level rank distribution further characterizes this difference over the full field. These measurements explain the local diagnostic signal; the supplementary comparison of normalized auxiliary-credit shares is retained in Appendix E.

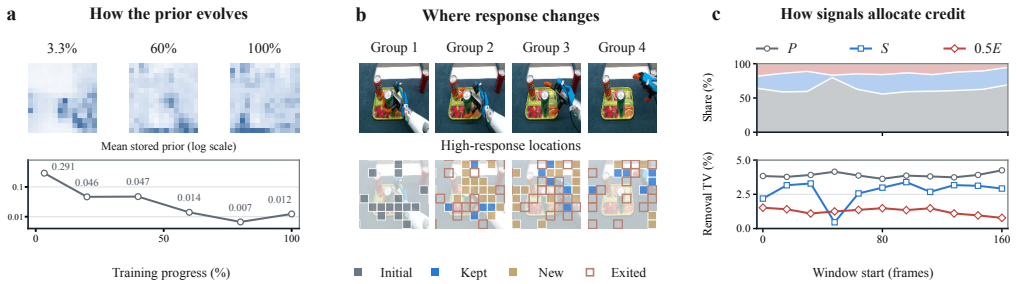


Figure 3: **Credit across three timescales. a**, prior spatial patterns at 3.3%, 60%, and 100% of the same can-placement run (top, independently normalized), and the raw spatial mean at six checkpoints (bottom, logarithmic axis). **b**, response-top40 locations across four future groups for the newly trained three-can tray-placement policy. **c**, stacked component shares (top) and fixed-output removal sensitivity (bottom) for $\mathcal{P}$, $\mathcal{S}$, and $0.5\mathcal{E}$ across all eleven exported windows of the same demonstration. Shares exclude the unit floor of $\mathcal{W}$; sensitivity is total variation (TV) between normalized full and component-removed weights, retaining the unit floor. Panel c uses all spatial cells and future groups. All panels use the same training run; b–c use its final checkpoint. Definitions and display protocols appear in Appendix F.

Local sensitivity and relative credit. Under small isotropic action perturbations, $\mathbb{E}_\delta[\mathcal{S}_i] \approx (\tau^2/C)\|J_i\|_F^2$, where J_i is the video-output Jacobian with respect to action inputs and C is the channel count (Appendix C.2). Response describes coupling at the current condition and diffusion level; a single direction gives a noisy estimate. Per-sample normalization expresses relative priority, not absolute sensitivity across tasks. Current error and rehearsal history complement this local coupling, while the unit floor preserves supervision at low-response locations.

4.3 Credit evolution within a task

Figure 3 follows one three-can tray-placement run. Panel a tracks spatial priors across training; b shows response locations across future groups at the final checkpoint; c summarizes component shares and removal sensitivity across eleven demonstration windows. The prior has the largest share and removal sensitivity, while response and error vary across windows. This distinguishes accumulated spatial difficulty from context-dependent reweighting: a final aggregate alone would hide variation over training and predicted time. The analysis characterizes fixed outputs, rather than retraining effects or object localization.

4.4 Real-robot evaluation and qualitative sequences

Figure 4 compares fifteen tasks, five baselines, and Ours, alongside four recorded task sequences. Event-aligned frames illustrate intermediate object states; twelve further sequences appear in Appendix G. The prediction-horizon analysis (Appendix Figure 13) separately evaluates all 50 illustrated training windows against a copy-condition baseline.

Ours achieves 89.00% task-macro-average SR, compared with 51.33% for the strongest baseline, G0.5, a margin of 37.67 percentage points. Supplementary phone-box assembly achieves 85% versus 5% for $\pi_{0.5}$; it is reported separately and excluded from the fifteen-task average (Appendix G).

Task variation and aggregation. Ours has the highest reported SR on each of the 15 physical tasks, but performance is not uniform. SR ranges from 70% for fan insertion to 100% for three-can placement, food placement, and coaster placement. Height-based bottle ordering reaches 90%, compared with 30% for the best baseline on that task. Each task contributes equally to the reported macro-average; it is not a pooled episode success rate. Trial counts and uncertainty estimates are not supplied with these percentages, so the comparisons describe the recorded outcomes rather than establish statistical significance. The separately reported phone-box assembly result does not change the common 15-task comparison set.

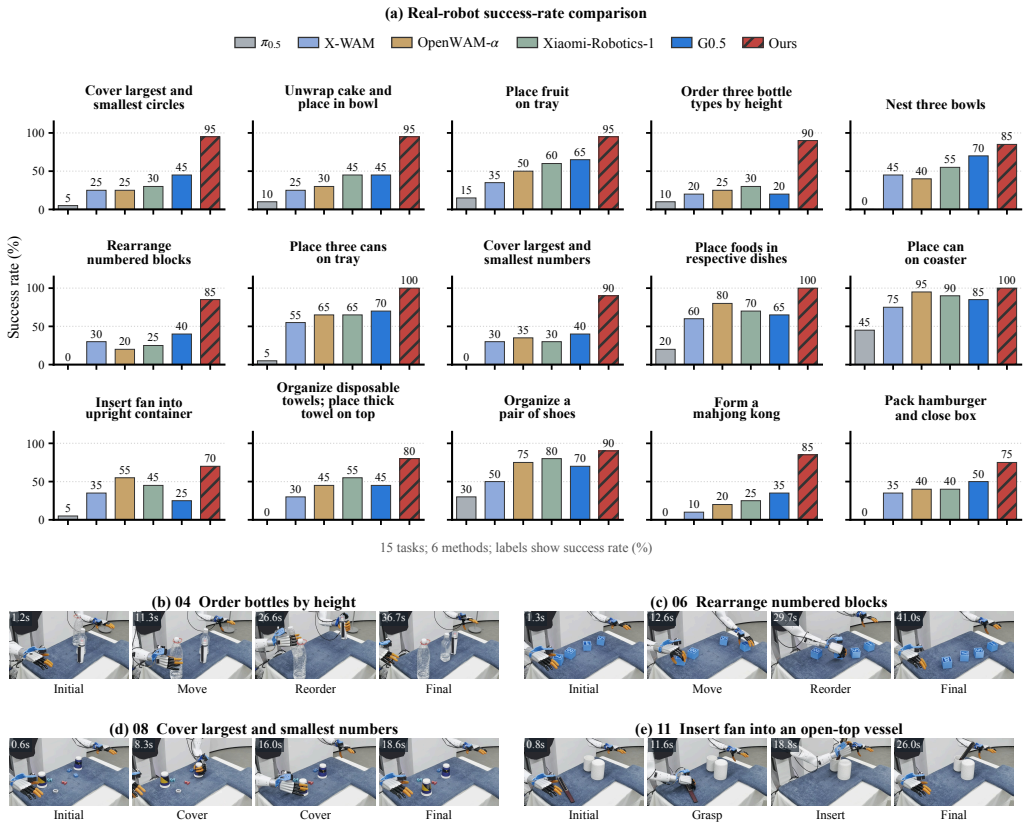


Figure 4: **Real-robot evaluation and four recorded task sequences.** (a) Success rates across fifteen tasks and six methods; bar labels show percentages. (b–e) Tasks 04, 06, 08, and 11 show height-based bottle ordering, numbered-block rearrangement, covering numerical extrema, and fan insertion. Each task occupies four consecutive frames in a two-row, eight-column grid. Timestamps refer to the source video. Frames are taken directly from the original videos with torsos and incidental text masked; task numerals are retained. The phone-box assembly task is provided in Appendix G.

4.5 Discussion

Two complementary patterns emerge: response and error prioritize different prediction locations, whereas candidate policies specialize on different tasks. The former concerns within-policy supervision; the latter concerns task-level selection. Separating their contributions requires budget-matched training ablations and policy selection independent of final testing. Weak Open and random-Generalization performance motivates new-layout evaluation. Specialist retention additionally requires checkpoint storage and reliable task identities.

5 Conclusion

We presented RAP-WAM, a world action framework coupling future-supervised action representations with predictive rehearsal. Its action-response signal organizes both spatial visual and temporal action supervision, complemented by current and persistent prediction errors. The framework gives future prediction two connected roles: shaping representations for action generation and supplying feedback that guides subsequent learning. Simulation and physical evaluations demonstrate the capabilities of the resulting policies, while task-specific post-training preserves complementary specializations within the complete system. Together, these findings motivate a unified treatment of predictive representation, supervision allocation, and task-level capability retention for world action models.

AI USE STATEMENT

Generative AI assisted with research discussions, post-training intervention design, implementation and experiment scripts, literature retrieval, manuscript drafting and translation, and figure preparation. The improvement agents used in the experimental workflow and their role in proposing training interventions are described in Section 3.4 and Appendix H; their outputs are part of the method under study. During manuscript preparation, assistance included checking citation records against primary sources, comparing reported values with saved experiment records, and compiling and inspecting the manuscript. All results reported in this paper have been manually reviewed by the authors. World-model predictions in Appendix J are identified as model outputs and paired with recorded targets. Reported quantitative results come from saved experiments or cited benchmark sources; pending measurements are not replaced with generated values. AI systems are not authors, and responsibility for the scientific claims, source attribution, code, and final submitted content remains with the authors.

ETHICS STATEMENT

This research does not involve human-subject experiments or user studies. We evaluate robot manipulation policies using robot demonstrations and simulation and real-robot execution records. Physical execution can cause unintended contact or motion; responsible deployment requires embodiment-specific checks of joint conventions and limits, an emergency stop, and supervised commissioning. Improved benchmark performance does not establish safety outside the evaluated conditions. Recorded demonstration frames and training-sample predictions are presented as qualitative evidence, not as additional autonomous-success trials. Display copies mask incidental identifying text, visible robot torsos in external-view sequences, and background faces in prediction comparisons; task-relevant numerals and mahjong symbols are retained. Original scene content is otherwise preserved, and the processing is disclosed in Appendices G and J. Masking alone does not establish consent or authorize redistribution: public release of demonstrations or identifiable recordings remains subject to the applicable dataset terms, participant permissions, and institutional requirements. No approval or consent status is inferred from the availability of these recordings. The limitations of task-conditioned selection, single-seed evaluation, and incomplete physical measurements are stated to avoid overstating generalization or autonomous capability.

REPRODUCIBILITY STATEMENT

We provide the source code in the supplementary materials. Section 3 defines the model interfaces, predictive rehearsal, response-based credit objective, and task-specific policy selection. Appendix A documents training and evaluation conventions, while Appendix C, including Algorithm 1, provides implementation details, assumptions, and derivations. Appendix H specifies post-training sampling, optimizer settings, checkpoint retention, inference, and aggregation; Appendix D reports task-level outcomes. Appendices E and F describe the diagnostic sampling and reconstruction procedures. The scope and processing of physical recordings and the selection of paired training predictions are described in Appendices G and J. The project records preserve source paths, hashes, selection manifests, and figure-generation scripts. These records support auditing the presented artifacts. Remaining reproducibility limits include the absence of independent policy-selection testing, multi-seed estimates, and completed physical-trial statistics.

REFERENCES

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28, 2015.

Hongzhe Bi, Hengkai Tan, Shenghao Xie, Zeyuan Wang, Shuhe Huang, Haitian Liu, Ruowen Zhao, Yao Feng, Chendong Xiang, Yinze Rong, et al. Motus: A unified latent action world model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 35101–35113, 2026.

Chris M Bishop. Training with noise is equivalent to tikhonov regularization. *Neural computation*, 7 (1):108–116, 1995.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Jisong Cai, Long Ling, Shiwei Chu, Zhongshan Liu, Jiayue Kang, Zhixuan Liang, Wenjie Xu, Yinan Mao, Weinan Zhang, Xiaokang Yang, et al. Aha-wam: Asynchronous horizon-adaptive world-action modeling with observation-guided context routing. *arXiv preprint arXiv:2606.09811*, 2026.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025.

Tianxing Chen, Yue Chen, Zixuan Li, Junyuan Tang, Kailun Su, Haoran Lu, Weijie Wan, Baijun Chen, Songling Liu, Haowen Yan, Honghao Su, Zhiyang Dou, Kaixuan Wang, Dandan Zhang, Yunze Liu, Yan Qin, Qiwei Liang, Qiwei Wu, Zijian Lin, Wenwei Lin, Yuran Wang, Minghua He, Tianshu Wu, Ruihai Wu, Jingquan Zhou, Kai-Chong Lei, Haibao Yu, Yuanfeng Ji, Weiyang Jin, Guanyu Lin, Xiaofan Li, Qi Xiong, Renjing Xu, Zhongyu Li, Wenhao Chai, Enze Xie, Ziwei Wang, Yao Mu, Hao Dong, Wojciech Matusik, Mingyu Ding, Wenbo Ding, Ping Luo, and Masayoshi Tomizuka. Robodojo: A unified sim-and-real benchmark for comprehensive evaluation of generalist robot manipulation policies, 2026a. URL https://arxiv.org/abs/2607.04434.

Tianxing Chen, Yue Chen, Zixuan Li, Junyuan Tang, Kailun Su, Haoran Lu, Weijie Wan, Baijun Chen, Songling Liu, Haowen Yan, et al. Robodojo: A unified sim-and-real benchmark for comprehensive evaluation of generalist robot manipulation policies. *arXiv preprint arXiv:2607.04434*, 2026b.

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

Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. *Advances in neural information processing systems*, 36:9156–9172, 2023.

Haoquan Fang, Jiafei Duan, Donovan Clay, Sam Wang, Shuo Liu, Weikai Huang, Xiang Fan, Wei-Chuan Tsai, Shirui Chen, Yi Ru Wang, et al. Molmoact2: Action reasoning models for real-world deployment. *arXiv preprint arXiv:2605.02881*, 2026.

Amir-massoud Farahmand, Andre Barreto, and Daniel Nikovski. Value-aware loss function for model-based reinforcement learning. In *Artificial Intelligence and Statistics*, pp. 1486–1494. PMLR, 2017.

Jun Guo, Qiwei Li, Peiyan Li, Zilong Chen, Nan Sun, Yifei Su, Heyun Wang, Yuan Zhang, Xinghang Li, and Huaping Liu. Unified 4d world action modeling from video priors with asynchronous denoising. *arXiv preprint arXiv:2604.26694*, 2026.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *Advances in Neural Information Processing Systems*, 38:167283–167308, 2025.

Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, Danny Driess, Michael Equi, Adnan Esmail, Yunhao Fang, Chelsea Finn, Catherine Glossop, Thomas Godden, Ivan Goryachev, Lachy Groom, Hunter Hancock, Karol Hausman, Gashon Hussein, Brian Ichter, Szymon Jakubczak, Rowan Jen, Tim Jones, Ben Katz, Liyiming Ke, Chandra Kuchi, Marinda Lamb, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Yao Lu, Vishnu Mano, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Charvi Sharma, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, Will Stoeckle, Alex Swerdlow, James Tanner, Marcel Torne, Quan Vuong, Anna Walling, Haohuan Wang, Blake Williams, Sukwon Yoo, Lili Yu, Ury Zhilinsky, and Zhiyuan Zhou. $\pi^*_{0.6}$: a vla that learns from experience, 2025. URL https://arxiv.org/abs/2511.14759.

Physical Intelligence, Bo Ai, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Greg Balke, Kevin Black, George Bokinsky, Shihao Cao, Thomas Charbonnier, Vedant Choudhary, Foster Collins, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Maitrayee Dhaka, Jared DiCarlo, Danny Driess, Michael Equi, Adnan Esmail, Yunhao Fang, Chelsea Finn, Catherine Glossop, Thomas Godden, Ivan Goryachev, Lachlan Groom, Haroun Habeeb, Hunter Hancock, Karol Hausman, Gashon Hussein, Victor Hwang, Brian Ichter, Connor Jacobsen, Szymon Jakubczak, Rowan Jen, Tim Jones, Gregg Kammerer, Ben Katz, Liyiming Ke, Mairbek Khadikov, Chandra Kuchi, Marinda Lamb, Devin LeBlanc, Brendon LeCount, Sergey Levine, Xinyu Li, Adrian Li-Bell, Vladislav Lialin, Zhonglin Liang, Wallace Lim, Yao Lu, Enyu Luo, Vishnu Mano, Nandan Marwaha, Aikys Mongush, Liam Murphy, Suraj Nair, Tyler Patterson, Karl Pertsch, Allen Z. Ren, Gavin Schelske, Charvi Sharma, Baifeng Shi, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, Will Stoeckle, Jiaming Tang, Jimmy Tanner, Shalom Tekeste, Marcel Torne, Kyle Vedder, Quan Vuong, Anna Walling, Haohuan Wang, Jason Wang, XuDong Wang, Chris Whalen, Samuel Whitmore, Blake Williams, Charles Xu, Sukwon Yoo, Lili Yu, Wuming Zhang, Zhuoyang Zhang, and Ury Zhilinsky. $\pi_{0.7}$: a steerable generalist robotic foundation model with emergent capabilities, 2026. URL https://arxiv.org/abs/2604.15483.

Joel Jang, Seonghyeon Ye, Zongyu Lin, Jiannan Xiang, Johan Bjorck, Yu Fang, Fengyuan Hu, Spencer Huang, Kaushil Kundalia, Yen-Chen Lin, et al. Dreamgen: Unlocking generalization in robot learning through video world models. *arXiv preprint arXiv:2505.12705*, 2025.

Yang Jin, Jun Lv, Wenye Yu, Hongjie Fang, Yong-Lu Li, and Cewu Lu. Sime: Enhancing policy self-improvement with modal-level exploration. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 9792–9799. IEEE, 2025.

Angelos Katharopoulos and François Fleuret. Not all samples are created equal: Deep learning with importance sampling. In *International conference on machine learning*, pp. 2525–2534. PMLR, 2018.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Fuhao Li, Wenxuan Song, Han Zhao, Jingbo Wang, Pengxiang Ding, Donglin Wang, Long Zeng, and Haoang Li. Spatial forcing: Implicit spatial representation alignment for vision-language-action model. In *International Conference on Learning Representations*, 2026.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.02747.

Yicheng Liu, Zibin Dong, Baijun Ye, Tianyuan Yuan, Tao Jiang, Anqi Yang, Shicheng Cao, Haonan Liu, Yue Sun, Zihan Guo, et al. G0.5: One autoregressive stream for robot reasoning and action. *arXiv preprint arXiv:2608.11739*, 2026.

Runyu Lu, Yubo Wu, Ethan Kou, Letian Fu, Wenli Xiao, Ajay Mandlekar, Yinzhen Xu, Guanya Shi, Ken Goldberg, Ang Chen, et al. ASPIRE: Agentic /skills discovery for robotics. *arXiv preprint arXiv:2607.00272*, 2026.

Haoxiang Ma, Junhao Cai, Xiaoxu Xu, Hao Li, Yuyin Yang, Yang Tian, Jiafei Cao, Hongrui Zhu, Zherui Qiu, Yuqiang Yang, et al. InternVLA-A1.5: Unifying understanding, latent foresight, and action for compositional generalization. *arXiv preprint arXiv:2607.04988*, 2026.

MotuBrain Team, Chendong Xiang, Fan Bao, Haitian Liu, Hengkai Tan, Hongzhe Bi, James Li, Jiabao Liu, Jingrui Pang, Kiro Jing, et al. Motubrain: An advanced world action model for robot control. *arXiv preprint arXiv:2604.27792*, 2026.

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*, 2025.

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.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *International Conference on Learning Representations*, 2016. URL https://arxiv.org/abs/1511.05952.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Yuran Wang, Siqiao Huang, Mingleyang Li, Chenhao Zhang, Jiaqi Liang, Weiyang Jin, Yue Chen, Xuemin Chi, Donghao Zhou, Qize Yu, et al. Openwam: An open, modular exploration towards systematic world-action model pretraining. *arXiv preprint arXiv:2609.07398*, 2026.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Wenli Xiao, Jia Xie, Tonghe Zhang, Haotian Lin, Letian Fu, Haoru Xue, Jalen Lu, Yi Yang, Cunxi Dai, Zi Wang, et al. Enpire: Agentic robot policy self-improvement in the real world. *arXiv preprint arXiv:2606.19980*, 2026.

Xiaomi Robotics Team, Jun Guo, Piaopiao Jin, Jason Li, Peiyan Li, Yingyan Li, Futeng Liu, Wanli Peng, Optimus Qin, Yifei Su, et al. Xiaomi-robotics-1: Scaling vision-language-action models with over 100k hours of real-world trajectories. *arXiv preprint arXiv:2607.15330*, 2026.

Jiazhi Yang, Kunyang Lin, Jinwei Li, Wencong Zhang, Tianwei Lin, Longyan Wu, Zhizhong Su, Hao Zhao, Ya-Qin Zhang, Li Chen, et al. Rise: Self-improving robot policy with compositional world model. *arXiv preprint arXiv:2602.11075*, 2026a.

Senqiao Yang, Chengyao Wang, Yuxin Chen, Zixuan Wang, Longxiang Tang, Haokun Gui, Jinhui Ye, Changsheng Lu, Xiaoyang Wu, Mingkang Zhu, et al. Beyond data scaling: Representation-centric continued pre-training for vision-language-action models. *arXiv preprint arXiv:2608.27550*, 2026b.

Tianhang Yang, Yanze Zheng, Junjie Wang, Wei-Bin Kou, Ruotong Li, and Yujiu Yang. Route by kinematics, act by observation: Kinematics-supervised expert routing in moe-augmented vla. *arXiv preprint arXiv:2607.26807*, 2026c.

Angen Ye, Boyuan Wang, Chaojun Ni, Guan Huang, Guosheng Zhao, Hao Li, Hengtao Li, Jie Li, Jindi Lv, Jingyu Liu, et al. Gigaworld-policy: An efficient action-centered world–action model. *arXiv preprint arXiv:2603.17240*, 2026a.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, et al. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026b.

Cheng Yin, Wang Xu, Junpeng Yang, Sikyuen Tam, Hanyu Liu, Yuan Yao, Xiangrui Zeng, Junbo Cui, Yequan Wang, Zhouping Yin, and Yankai Lin. Simplememvla: A simple but effective native-video memory for vision-language-action models, 2026. URL https://arxiv.org/abs/2609.05533.

Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-wam: Do world action models need test-time future imagination? *arXiv preprint arXiv:2603.16666*, 2026.

He Zhang, Lingzhu Xiang, Haitao Lin, Zeyu Huang, Minghui Wang, Dingyan Zhong, Yubo Dong, Yihao Wu, Yongming Rao, Dongsheng Zhang, et al. Hy-embodied-0.5-vla: From vision-language-action models to a real-world robot learning stack. *arXiv preprint arXiv:2606.14409*, 2026.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE/CVF conference on computer vision and pattern recognition*, pp. 586–595. IEEE, 2018.

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, et al. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. In *International Conference on Learning Representations*, 2026.

Chuning Zhu, Raymond Yu, Siyuan Feng, Benjamin Burchfiel, Paarth Shah, and Abhishek Gupta. Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets. *arXiv preprint arXiv:2504.02792*, 2025.

APPENDIX: SUPPLEMENTARY MATERIAL

A TRAINING AND EVALUATION PROTOCOL

Training was performed on NVIDIA A800 GPUs.

RAP-WAM initializes its shared transformer from Wan2.2-TI2V-5B video weights (Team Wan et al., 2025) and jointly learns future video and robot actions from target RoboDojo demonstrations. Action/state encoders, queries, and CAP are initialized for target-task training, without initialization from a separately trained robot policy. The base architecture contains no additional memory, mixture-of-experts, or post-training extension. Thus "without embodied pretraining" excludes an intermediate pretraining stage on external embodied data, whether for action-policy learning or joint world action modeling. It retains Wan video pretraining, pretrained text embeddings, and joint world action training on target robot demonstrations. The collection has 3,500 episodes, with 3,499 used for training and one for validation. The fixed RAP-WAM base is the reference for local mechanism analyses. Table 1 reports the complete post-training system, whose retained task policies build on this substrate. The settings in Table 2 describe the base learner, not the cumulative computation of the entire improvement process.

Table 2: Training settings. Physical training uses the same core with task-specific interfaces; physical success rates appear in Figure 4.

Setting	RoboDojo	Physical-robot data
Video initialization	Wan2.2-TI2V-5B	Wan2.2-TI2V-5B
Input	Head/two-wrist 384×320 mosaic	Head 224×224
Action/state dimension	14/14	Phone 28/28; numbers 32/32
Language	Precomputed instruction embeddings	None
Observation/future/action length	1/16/16	1/16/16
Queries/alignment width	16/128	16/128
Per-device/global batch	1/96	1/120
Optimizer/peak LR	AdamW/2×10^{-4}	AdamW/2×10^{-4}
Precision/parameter EMA	BF16/none	BF16/none
Auxiliary warmup (updates)	18,000	6,000
Rehearsal probability/denoising	0.25/four iterations	0.25/four iterations
Conditional state substitution	0.5	0.5
response-based credit probability	0.5	0.5
response-based credit video/action coefficients	0.5/0.3	0.5/0.3
$\mathcal{P}, \mathcal{S}, \mathcal{E}$ coefficients	1/1/0.5	1/1/0.5
VAE	Cached recorded futures; predicted inputs encoded live	Live

RoboDojo policy evaluation uses 30 denoising steps and executes eight actions before replanning. Action smoothing uses a Savitzky–Golay filter with window 9 and polynomial order 2 on all 14 action coordinates. Four-action boundary blending uses new-chunk coefficients $(i+1)/5$ for $i = 0, 1, 2, 3$. This filter is distinct from classifier-free guidance. Physical deployment additionally depends on robot-specific joint mappings, units, and control rates.

Generalization has 12 tasks, Precision 8, Long-Horizon 8, Memory 6, and Open 8. Binary success rate (SR) and partial-completion Score are averaged within capabilities and then equally across the five capability means (Chen et al., 2026b):

$$\mathrm{AvgSR} = \frac{1}{5} \sum_{d=1}^{5} \frac{1}{|\mathcal{T}_d|} \sum_{j \in \mathcal{T}_d} \mathrm{SR}_j. \tag{12}$$

Here $\mathcal{T}_d$ contains the evaluated tasks in capability d; Score uses the same aggregation. Row-specific coverage is given in the table notes and Appendix D. Standard and random variants are combined into their parent Generalization task before aggregation. Both the base and agentic system cover all 42 task units: 12 Generalization, eight Precision, eight Long-Horizon, six Memory, and eight Open.

For task t, SR is the number of successful valid episodes divided by its valid-episode count n_t, and Score is the mean partial-completion score over the same episodes. Task means are averaged within capability and then equally across capabilities; tasks are not reweighted by n_t. Layout identities and per-task denominators are retained in the result records.

The target alignment projection is stopped along with its target features. The direct attention mask permits state-mediated information and shared-parameter updates. At per-device batch one, the optional local min–max difficulty correction vanishes. Spatial $\mathcal{P}$ broadcasts uniformly in time, so its normalized temporal contribution also vanishes. The prior is zero until its first valid rehearsal update. Rehearsal and response-based credit are invoked according to their respective sampling probabilities. Only successfully computed auxiliary losses with nonzero weights enter the training objective. For each branch, the effective participation rate is the fraction of training updates satisfying these conditions.

B Public References and Pretraining Attribution

G0.5 pretrains on robot demonstrations and web-scale vision–language data, with 14 real/simulated embodiments mapped to a shared action space (Liu et al., 2026). G0.5 therefore uses external embodied pretraining. Xiaomi-Robotics-1 and Hy use action-policy pretraining on handheld UMI demonstrations (Xiaomi Robotics Team et al., 2026; Zhang et al., 2026); these are action-labeled embodied trajectories, not necessarily robot-collected data. OpenWAM-α jointly pretrains future-video and action prediction on egocentric and robot data (Wang et al., 2026). Its final mixture contains 6,369 hours of egocentric human and robot trajectories; human video is not itself labeled robot-action supervision. Our no-external-embodied-pretraining flag still includes Wan video initialization, pretrained text embeddings, and joint world action training on target robot demonstrations. Table 1 therefore defines "Pretrained" as an intermediate stage on external embodied data, including robot/UMI action-policy and joint world action pretraining, rather than asserting that every pretraining trajectory was robot-collected.

External embodied pretraining. The pretraining column summarizes the method's reported use of external embodied data. InternVLA-A1.5 uses robot-episode pretraining (Ma et al., 2026); VLAct and X-VLA use cross-embodiment robot pretraining, and MolmoAct2 uses action-data pretraining (Yang et al., 2026b; Zheng et al., 2026; Fang et al., 2026). Spatial Forcing inherits the Pi-05 base, so both carry a positive embodied-pretraining flag (Physical Intelligence et al., 2025).

X-WAM reports more than 5,800 hours of robot-data pretraining (Guo et al., 2026). GigaWorld-Policy introduces embodied-video pretraining using robot and egocentric human data before target-task action training (Ye et al., 2026a). AHA-WAM reports no robot-data pretraining in its default simulation setting, but its real-world comparison pretrains both AHA-WAM and Fast-WAM on 24,600 RoboCOIN trajectories (approximately 165 hours) (Cai et al., 2026). The AHA-WAM Yes flag refers to this reported real-world variant; it does not establish the initialization of the RoboDojo checkpoint. X-WAM and GigaWorld-Policy-0 are marked Yes based on their reported pretraining. Fast-WAM's reported simulation configuration uses Wan initialization without external embodied pretraining (Yuan et al., 2026).

SimpleMemVLA training and comparison source. SimpleMemVLA uses timestamped visual history with a Qwen3.5-4B backbone and a flow-matching action head (Yin et al., 2026). Its released training recipe initializes the backbone from Qwen3.5-4B, constructs a new action head, and trains on each benchmark's demonstrations without a separate embodied-pretraining stage. We therefore mark No under the definition above. The RoboDojo values use the official 23 September 2026 leaderboard snapshot, including separate standard and randomized Generalization results.

World-model classification. Table 1 separates VLA and other action policies from world-model-based policies. The latter group includes OpenWAM-α, X-WAM, GigaWorld-Policy-0, AHA-WAM, Fast-WAM and our policy, which learn an explicit predictive world-model component alongside action generation. InternVLA-A1.5 is grouped with VLAs: its foresight tokens receive latent supervision from a frozen pretrained video model, but the policy does not learn pixel-level generation and discards the video branch at inference (Ma et al., 2026). Its ‡ marker records this supervision source without classifying the policy as a world action model.

Table 3: Post-training system results across 42 tasks. SR is in percent; Score measures partial completion. Each task is averaged over its valid evaluation episodes.

Task	Capability	SR	Score
Align blocks	Open	0.00	0.00
Arrange largest number	Generalization	4.00	5.80
Build tower	Precision	72.00	79.60
Classify objects	Long-Horizon	2.00	6.50
Classify objects by language	Open	0.00	1.60
Cover blocks	Memory	22.00	26.10
Deposit coin	Precision	10.20	14.69
Fasten screws	Precision	8.00	27.60
Fill egg holder	Long-Horizon	0.00	12.95
Fill pen holder	Long-Horizon	18.00	43.40
Fold clothes	Generalization	12.00	16.00
General pickup	Open	6.00	6.00
Hang mugs	Generalization	2.00	7.30
Imitate sorting sequence	Memory	2.00	3.80
Insert key	Precision	0.00	15.00
Insert tubes	Precision	58.00	72.40
Make kong	Long-Horizon	82.00	82.00
Make toast	Generalization	10.00	18.00
Match and pick from conveyor	Memory	32.00	32.00
Organize table	Long-Horizon	10.00	48.50
Pack objects into box	Generalization	6.00	17.20
Pick from conveyor by image	Open	0.00	0.00
Play xylophone	Precision	2.00	2.00
Play stacking toy	Long-Horizon	0.00	2.00
Play tic tac toe	Long-Horizon	86.00	96.00
Plug in charger	Precision	4.00	4.00
Pour balls into vase	Precision	14.00	14.00
Pour by language	Open	0.00	0.00
Pour liquid into cup	Generalization	4.00	4.00
Press by number	Memory	0.00	0.00
Push t	Generalization	0.00	0.00
Put bottles into dustbin	Long-Horizon	94.00	96.10
Solve equation	Open	0.00	0.00
Sort nesting dolls by size	Generalization	8.00	8.00
Stack blocks	Generalization	24.00	28.20
Stack blocks by language	Open	0.00	1.20
Stack bowls	Generalization	42.00	43.20
Store laptop and headphones	Generalization	14.00	24.80
Store tools in toolbox	Open	0.00	0.50
Swap t	Memory	0.00	0.00
Swap blocks	Memory	0.00	0.00
Sweep blocks	Generalization	0.00	0.00

Xiaomi, Hy, Pi-05, VLAct, X-VLA and MolmoAct2 document action-generation objectives without an explicit predictive world-model component (Xiaomi Robotics Team et al., 2026; Zhang et al., 2026; Physical Intelligence et al., 2025; Yang et al., 2026b; Zheng et al., 2026; Fang et al., 2026). Spatial Forcing aligns observed geometry, not future scene evolution (Li et al., 2026); its benchmark identity is the Pi-05-based variant. Action flow matching, current depth prediction, and textual subtask planning alone do not qualify as world modeling here. KinRT uses kinematics-supervised expert routing for action generation and is grouped with non-world-model policies (Yang et al., 2026c). SimpleMemVLA is grouped with VLAs: its video context represents observed history, rather than an explicitly predicted future world state. These method-level labels neither make all architectures identical nor explain performance causally.

Table 4: RAP-WAM base results across 42 tasks, using the single step-742500 checkpoint before post-training. Task order and metric definitions match Table 3. SR (%) and Score are computed over valid evaluation episodes.

Task	Capability	SR	Score
Align blocks	Open	0.00	0.00
Arrange largest number	Generalization	2.00	3.80
Build tower	Precision	60.00	69.80
Classify objects	Long-Horizon	0.00	4.40
Classify objects by language	Open	0.00	1.60
Cover blocks	Memory	16.00	20.60
Deposit coin	Precision	4.08	7.76
Fasten screws	Precision	2.00	18.60
Fill egg holder	Long-Horizon	0.00	12.95
Fill pen holder	Long-Horizon	6.00	32.00
Fold clothes	Generalization	6.00	9.20
General pickup	Open	6.00	6.00
Hang mugs	Generalization	0.00	3.80
Imitate sorting sequence	Memory	0.00	1.20
Insert key	Precision	0.00	14.40
Insert tubes	Precision	54.00	69.20
Make kong	Long-Horizon	74.00	74.00
Make toast	Generalization	4.00	14.00
Match and pick from conveyor	Memory	26.00	26.00
Organize table	Long-Horizon	10.00	48.50
Pack objects into box	Generalization	2.00	14.70
Pick from conveyor by image	Open	0.00	0.00
Play xylophone	Precision	0.00	0.00
Play stacking toy	Long-Horizon	0.00	0.60
Play tic tac toe	Long-Horizon	80.00	94.50
Plug in charger	Precision	0.00	0.00
Pour balls into vase	Precision	12.00	12.00
Pour by language	Open	0.00	0.00
Pour liquid into cup	Generalization	0.00	0.00
Press by number	Memory	0.00	0.00
Push t	Generalization	0.00	0.00
Put bottles into dustbin	Long-Horizon	90.00	94.00
Solve equation	Open	0.00	0.00
Sort nesting dolls by size	Generalization	2.00	2.00
Stack blocks	Generalization	24.00	27.00
Stack blocks by language	Open	0.00	0.40
Stack bowls	Generalization	40.00	42.10
Store laptop and headphones	Generalization	10.00	17.60
Store tools in toolbox	Open	0.00	0.50
Swap t	Memory	0.00	0.00
Swap blocks	Memory	0.00	0.00
Sweep blocks	Generalization	0.00	0.00

C Mathematical Properties and Objective Details

C.1 Predictive representation and training update

Joint flow-matching substrate. We use the linear interpolation and velocity-regression formulation of flow matching (Lipman et al., 2023).

A frozen VAE encodes observed and subsequent demonstration video, with fixed camera mosaics for multi-view observations. Actions and states use separate training-set normalizers. Video and actions share a noise level σ but receive independent noises:

$$z_\sigma = (1-\sigma)z + \sigma\epsilon_z, \quad a_\sigma = (1-\sigma)a + \sigma\epsilon_a, \quad (v_z^*, v_a^*) = (\epsilon_z - z, \epsilon_a - a). \tag{13}$$

Observed visual latents are clamped and excluded from the future-only video loss. The video-initialized trunk is trained together with newly initialized action/state encoders, queries, and CAP on target demonstrations.

Tokenization, conditioning, and action projection. Action and state encoders use Linear–SiLU–Linear projections, learned position embeddings, and LayerNorm. All four token groups share each block's self-attention projections, text cross-attention, and feed-forward layers; there are no modality-specific transformer experts. Rotary position encoding applies only to video-token queries and keys. Predictive queries use action-side diffusion-time modulation. State tokens receive no additional diffusion-time modulation, but still undergo the block's learned modulation and all updates. The mask in Equation 3 is applied identically at training and sampling: for $i \in \mathcal{A}, j \in \mathcal{V}$, the logit is suppressed; state-mediated paths remain.

The two projections in Equation 1 have output width 128. The entire target branch, including P_t, is evaluated without gradients. Future latents are pooled over time and onto a spatial 4×4 grid; the resulting targets supervise 16 distinct query slots. They are neither an online memory bank nor an independently trained teacher. In Equation 4, the MLP output projection and bias are zero-initialized; the time-modulated branch starts from the newly initialized sequence head. Its exact form is

$$H_r(h, e) = W_r[\mathrm{LN}(h) \odot (1 + e + m_s) + (e + m_b)] + b_r, \tag{14}$$

with learned offsets m_s, m_b. The two branches have separate normalization operators. The construction introduces no low-pass filter, task routing, or independently supervised coarse/fine decomposition.

Generated-image and state conditions. Rehearsal is invoked with $p_c(k) = \bar{p}_c r_c(k)$, where $r_c(k) = \min(1, k/W_c)$. Conditional state substitution has probability $\bar{p}_s \min(1, k/W_s)$. In the reported configuration, $\bar{p}_c = 0.25$ and $\bar{p}_s = 0.5$. The first chunk uses S_c gradient-free denoising steps, with $S_c = 4$, before RGB decoding. Its last image replaces the next observed image; this prefix and the recorded next future frames are jointly re-encoded by the frozen VAE. Thus the recorded RGB targets are retained, but their temporal VAE latents need not equal those encoded with the recorded initial image. For action and state normalizers N_a, N_s, the substituted normalized state is

$$\widetilde{s}_1 = N_s\big((1 - b_s)s_1^{\mathrm{raw},*} + b_s \,\mathrm{sg}\big[N_a^{-1}(\widehat{a}_{0,H_a-1})\big]\big)\,. \tag{15}$$

Here $\widehat{a}_0$ is the sampled normalized action chunk and b_s selects state substitution. This mapping assumes compatible absolute-joint action and state schemas. It does not model actuator dynamics or contact feedback. The base loss includes query alignment; rehearsal and credit auxiliaries supervise video/action flow outputs without additional alignment losses.

The persistent prior is initially zero, so credit remains defined before a rehearsal update. The first valid rehearsal-error map initializes the prior directly; subsequent maps use the moving average. The reference predictions use the original demonstration batch with fresh noising, not the rehearsed input. Each credit invocation uses one gradient-carrying reference prediction and one gradient-free perturbed forward, in addition to the base and any rehearsal computations.

The algorithm assumes valid training inputs, with a zero prior before its first valid rehearsal update. It specifies the update rule, not the empirical frequency with which each auxiliary branch contributes during training (Appendix A). It neither sends actions to an environment nor requests new corrective demonstration labels at predicted states.

Joint sampling and deployment. Inference begins with independent future-video and action noise. At adjacent noise levels, Euler integration updates both modalities:

$$(z_{n+1}, a_{n+1}) = (z_n, a_n) + (\sigma_{n+1} - \sigma_n)(v_\theta^z, v_\theta^a). \tag{16}$$

The observed visual prefix is clamped at every iteration. Queries, directional attention, and CAP remain in each forward; the action output is $N_a^{-1}(a_N)$. Returning actions does not decode future RGB, but it does retain future-latent computation. Alignment targets, rehearsal, prior updates, and perturbation-based credit construction are absent at deployment.

C.2 Local interpretation of the perturbation response

Fix θ, observation, state, language, video noise, and diffusion level. Let $f_i : \mathbb{R}^d \to \mathbb{R}^C$ map the flattened normalized noisy action chunk to the predicted video velocity at location i. For $g \sim \mathcal{N}(0, I_d)$

Algorithm 1 One training update of RAP-WAM

Require: Adjacent demonstration chunks, model θ, spatial prior $\mathcal{P}$ (initially zero), step k

Sample base noising; compute joint loss $\mathcal{L} \leftarrow \mathcal{L}_0$
if $b_c \sim \mathrm{Bernoulli}(p_c(k))$ is one **then**
    Generate the first chunk without gradients using S_c denoising steps
    Form the second-chunk image from the last predicted frame
    Optionally substitute the last raw predicted joint action; normalize as state
    Compute video/action flow loss $\mathcal{L}_1$ against the recorded future/actions
    Update $\mathcal{P}$ from detached second-chunk spatial errors
    $\mathcal{L} \leftarrow \mathcal{L} + \lambda_c r_c(k)\mathcal{L}_1$
end if
if $b_P \sim \mathrm{Bernoulli}(p_P)$ is one **then**
    Fresh auxiliary noising; compute gradient-carrying (v^z, v^a)
    Evaluate the action perturbation without gradients; detach $\mathcal{P}, \mathcal{S}, \mathcal{E}$
    Compute $\mathcal{W}, c, w$ and normalize their masses to q^v, q^a
    $\mathcal{L} \leftarrow \mathcal{L} + \widetilde{\mathcal{L}}_P(\theta; \bar{\theta})$
end if
Backpropagate $\mathcal{L}$ and update the trainable parameters

and $\delta = \tau g$, $\mathcal{S}_i = C^{-1}\|f_i(a+\tau g) - f_i(a)\|^2$. At the noisy action input a_σ, the local interpretation is

$$\mathbb{E}_\delta[\mathcal{S}_i] = \frac{\tau^2}{C}\|J_a f_i(a_\sigma)\|_F^2 + O(\tau^4). \tag{17}$$

This perturbation argument (Bishop, 1995) motivates a forward-only sensitivity probe without explicit Jacobians or second-order derivatives, under the regularity conditions stated below.

Proposition 1 (response interpretation). Suppose f_i is three times continuously differentiable on $\mathbb{R}^d$ and its third derivative has globally bounded operator norm. At fixed a,

$$\mathbb{E}\mathcal{S}_i = \frac{\tau^2}{C}\|J_a f_i(a)\|_F^2 + O(\tau^4) \quad \text{as } \tau \to 0. \tag{18}$$

For an affine f_i, the expression without the remainder is exact for every τ. This is a standard local sensitivity calculation, not a new control guarantee or an assertion that a finite-precision model satisfies the assumptions.

Proof. Write the Taylor expansion as

$$f_i(a+\tau g) - f_i(a) = A + B + R, \quad A = \tau J g, \quad B = \tfrac{1}{2}\tau^2 H[g,g], \quad \|R\| \le M_3\tau^3\|g\|^3/6. \tag{19}$$

Here M_3 bounds the third derivative, $M_1 = \|J\|_{\mathrm{op}}$, and $M_2 = \|H\|_{\mathrm{op}}$ at a. Gaussian second moments give $\mathbb{E}\|A\|^2 = \tau^2\|J\|_F^2$, and symmetry makes the cubic linear–quadratic cross term vanish. With $m_p = \mathbb{E}\|g\|^p$, the remaining terms satisfy

$$\left|\mathbb{E}\mathcal{S}_i - \frac{\tau^2}{C}\|J\|_F^2\right| \le \frac{1}{C}\left[\tau^4\left(\frac{M_2^2}{4} + \frac{M_1 M_3}{3}\right)m_4 + \tau^5\frac{M_2 M_3}{6}m_5 + \tau^6\frac{M_3^2}{36}m_6\right]. \tag{20}$$

All Gaussian moments are finite, proving the asymptotic bound. Affine maps have $H = R = 0$. The constants can depend on dimension and the fixed predictor. □

Single-direction and scale limitations. For an affine map,

$$\mathrm{Var}(\mathcal{S}_i) = \frac{2\tau^4}{C^2}\,\mathrm{tr}[(J^\top J)^2]. \tag{21}$$

Thus shrinking τ does not eliminate the relative uncertainty of a single direction. The implementation uses one perturbation, and generally $\mathbb{E}\mathcal{N}(\mathcal{S}) \neq \mathcal{N}(\mathbb{E}\mathcal{S})$; its normalized map is not an exact Jacobian-norm ranking. Nor does per-coordinate $\tau = 0.1$ imply a small whole-chunk perturbation: the RMS norm is $\tau\sqrt{d}$, approximately 1.50 for $d = 16\times14$ and 2.12 for $d = 16\times28$ in normalized coordinates. The analysis motivates perturbation-scale and repeatability checks, not an environment-causality interpretation or a prediction of task success.

C.3 Response and prediction error need not agree

Figure 5 illustrates how prediction error, action response, and historical error yield different supervision weights. For $f_1(a) = b$ and $f_2(a) = \kappa a$, suitable targets give

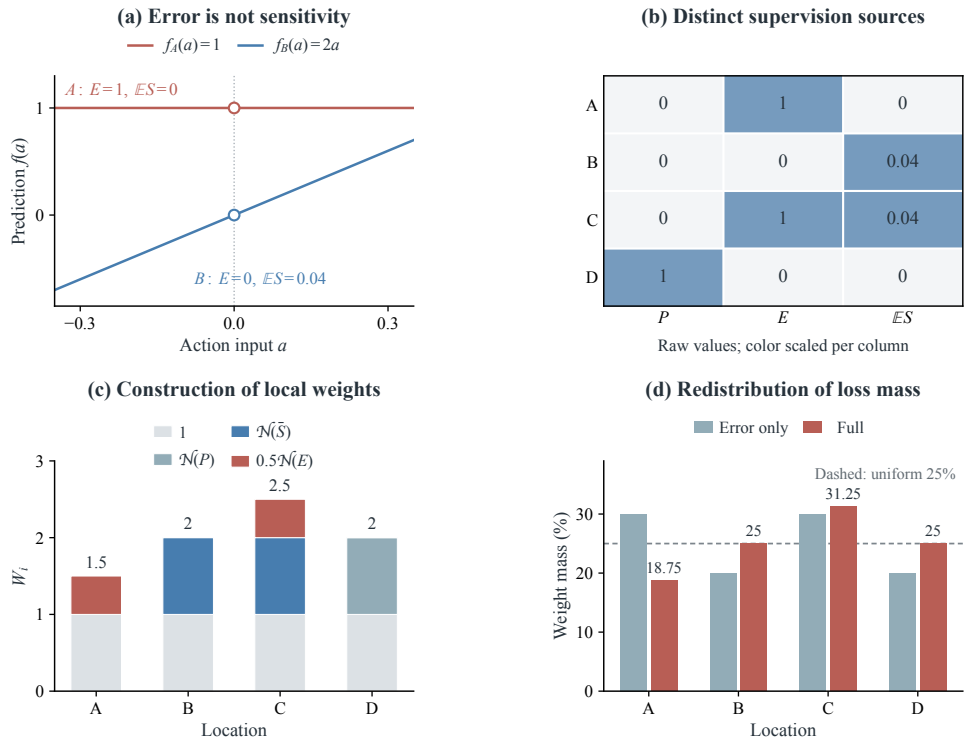


Figure 5: **Why error, response, and rehearsal prior provide different supervision.** An exact constructed example, not measured robot results. At four locations, let $f_A(a) = 1$, $f_B(a) = 2a$, $f_C(a) = 1 + 2a$, and $f_D(a) = 0$, with $a = 0$, target zero, and prior $P = (0, 0, 0, 1)$. For $\delta \sim \mathcal{N}(0, 0.1^2)$, use the expected response $\bar{S} = \mathbb{E}_\delta S$. (a) A has error without action sensitivity; B has sensitivity without error. (b) The four locations separate residual, response, combined, and historical signals. (c) Applying the spatial rule to $P, \bar{S}, E$ gives positive weights $(1.5, 2, 2.5, 2)$. (d) After normalization, error-only allocation gives $(30, 20, 30, 20)\%$, whereas the full rule (Full) gives $(18.75, 25, 31.25, 25)\%$. Thus current error alone cannot reproduce the allocation. This illustration uses expected response; training uses a single sampled perturbation, as discussed in Appendix C.2.

$$\mathcal{E}_1 = (b - y_1)^2 > 0, \quad \mathcal{E}_2 = 0, \qquad \mathbb{E}\mathcal{S}_1 = 0, \quad \mathbb{E}\mathcal{S}_2 = \tau^2\kappa^2. \tag{22}$$

A constant action-independent output can have nonzero error but zero response; an accurately predicted linear output can have zero error but nonzero expected response. This demonstrates non-equivalence of signals, not an empirical improvement in robot success. The finite-perturbation and normalization qualifications above still apply.

C.4 Frozen-weight surrogate and redistribution identity

For the main visual objective, bars denote uniform empirical averages and Cov_u the corresponding covariance over active locations:

$$\sum_i q_i^v e_i^z = \bar{e}^z + \frac{\mathrm{Cov}_u(\mathcal{W}, e^z)}{\bar{\mathcal{W}}}, \tag{23}$$

This separates prediction error from emphasis on particular locations. The general form and its gradient analogue follow below. Let i enumerate all active batch/spatiotemporal locations, with

channel-mean errors $e_i(\theta)$ and positive weights $\mathcal{W}_i$. At current $\bar{\theta}$, freeze $q_i = \mathcal{W}_i / \sum_j \mathcal{W}_j$. The local surrogate and its gradient are

$$\widetilde{F}(\theta; \bar{\theta}) = \sum_i q_i(\bar{\theta}) e_i(\theta), \qquad g_q = \sum_i q_i(\bar{\theta}) \nabla e_i(\bar{\theta}). \tag{24}$$

This is the precise backward-pass meaning of stop-gradient. At differentiable points, the fully adaptive value $F(\theta) = \sum_i q_i(\theta) e_i(\theta)$ has the additional derivative $\sum_i e_i \nabla q_i$, which the implementation intentionally omits. Ties in min–max normalization do not prevent defining the frozen-weight surrogate.

Proposition 2 (redistribution identities). Let bars and Cov_u use uniform empirical averaging over n locations, with covariance denominator n, and let $g_i = \nabla e_i(\bar{\theta})$. Then

$$\sum_i q_i e_i = \bar{e} + \frac{\mathrm{Cov}_u(\mathcal{W}, e)}{\bar{\mathcal{W}}}, \qquad g_q - \bar{g} = \frac{\mathrm{Cov}_u(\mathcal{W}, g)}{\bar{\mathcal{W}}}. \tag{25}$$

The second covariance is defined componentwise. These are explanatory identities, not original theorems of optimization or closed-loop improvement.

Proof. For either scalar $x_i = e_i$ or vector $x_i = g_i$, expanding the empirical covariance gives $n^{-1} \sum_i \mathcal{W}_i x_i = \bar{\mathcal{W}} \bar{x} + \mathrm{Cov}_u(\mathcal{W}, x)$. Divide by $\bar{\mathcal{W}} > 0$ and use the definition of q. □

Consequently, normalized weighting can change gradient direction and/or magnitude when priorities correlate with location-wise gradients. It need not decrease the fully adaptive value F, and it is not a corrected importance sampler for the uniform risk. Within a single sample, $\mathcal{E}_i = e_i$, so its spatial/temporal empirical covariance satisfies $\mathrm{Cov}_{u,s}(\mathcal{N}(\mathcal{E}), e) = \mathrm{Var}_{u,s}(e) / \max(\max_i e_i - \min_i e_i, \epsilon_{\mathrm{n}}) \geq 0$: part of response-based credit explicitly increases coefficients at high-error locations. Whether adding response $\mathcal{S}$ or persistent prior $\mathcal{P}$ improves control is an empirical question.

Bounded emphasis without exclusion. Let $\kappa_v = 1 + \alpha_P + \alpha_S + \alpha_E$ and $\kappa_a = 1 + \alpha_S$. For N_v video positions and N_a action positions in their respective normalizations, nonnegative source coefficients imply

$$\frac{1}{\kappa_v N_v} \leq q_i^v \leq \frac{\kappa_v}{N_v}, \qquad \frac{1}{\kappa_a N_a} \leq q_j^a \leq \frac{\kappa_a}{N_a}. \tag{26}$$

Our settings give $\kappa_v = 3.5$ and $\kappa_a = 2$. Thus selectivity retains every location. Detachment removes direct gradients through credit while allowing it to adapt across updates. These bounds regulate emphasis, not gradient norms, unbiased estimation, or preservation of previously acquired capabilities.

C.5 Limits of the interpretation

The identities characterize the weighting rule, not its causal effect on success. No matched-budget, independently evaluated component ablation is reported here. A claim of signal-specific control benefit would require matching auxiliary computation, initialization, data, and checkpoint selection, and separating weighting information from gradient-scale changes. Normalizing weight sums does not match the full AdamW update. Numerical checks on affine and smooth toy maps verify the identities, not robot performance.

D Task-level Base and Post-training Results

Tables 3 and 4 report the post-training system and the fixed base on the same 42 tasks. The base achieves 12.85% SR and 17.86 Score under the capability-balanced aggregation. Base and system use identical task coverage and aggregation rules; per-task SR and Score come from one policy's valid evaluation records. Generalization variants are combined within their parent task.

Figure 6 compares the per-task changes in SR and Score relative to the base.

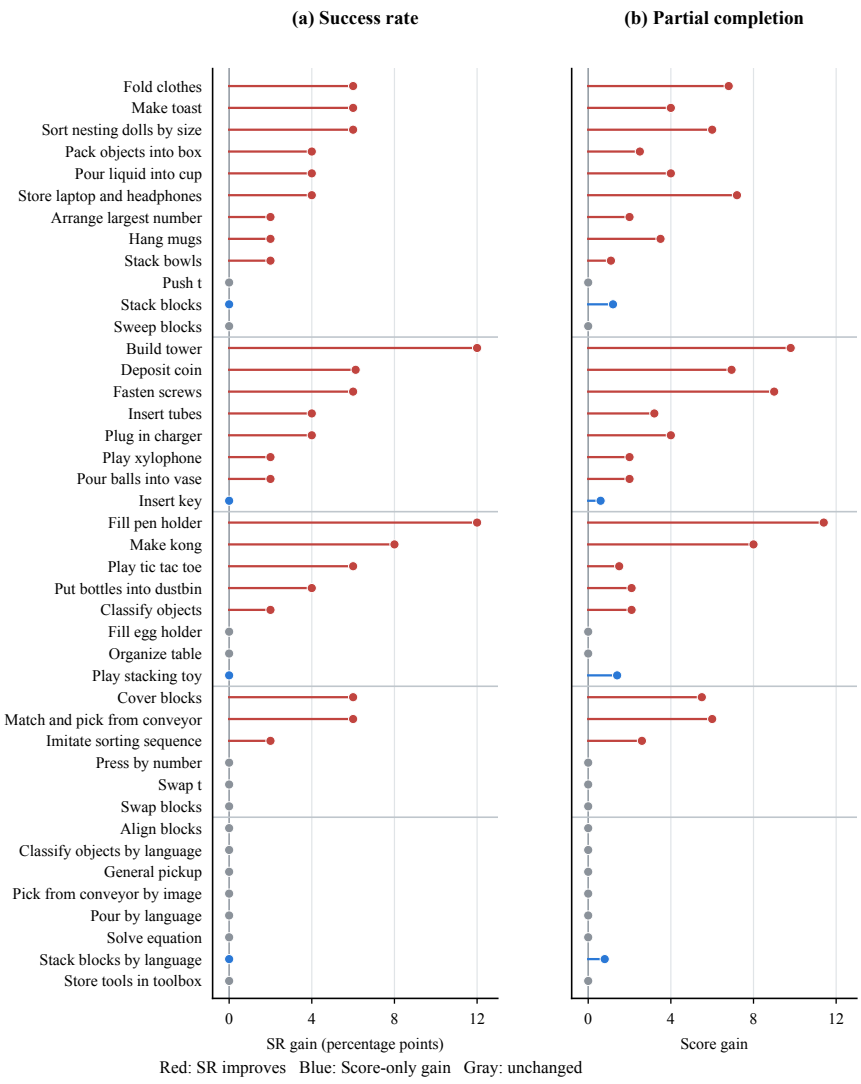


Figure 6: **Task-level changes under the selected assignment.** Aligned rows show all 42 tasks, grouped by capability: (a) SR changes and (b) partial-completion Score changes relative to the base. Red indicates 24 tasks with higher SR, blue four tasks with higher Score only, and gray 14 unchanged tasks. These are observed results under the selected task-policy assignment, not an independent test of regression-free improvement.

E Supervision Allocation and Response Analysis

The analyses use fixed-policy predictions to quantify supervision allocation. Simulation statistics use the RAP-WAM base and 35 conditions of nine demonstration windows each, separate from benchmark evaluation episodes. Physical-demonstration sources are specified in Appendix F.

Field notation. $\mathcal{P}, \mathcal{S}, \mathcal{E}, \mathcal{W}$ denote the cross-chunk prior, action response, prediction error, and composite visual-loss coefficient, respectively. The three simulation component fields used in the analyses below are normalized.

Arithmetic source-removal analysis. Figure 2a–b uses the same simulation component maps, with model outputs held fixed. Each of 315 windows has $N = 4 \times 24 \times 20 = 1{,}920$ native positions, giving 604,800 positions in total, not that many independent observations. Cast the normalized $\mathcal{P}, \mathcal{S}, \mathcal{E}$ arrays to float64 and reconstruct $\mathcal{W} = 1 + \mathcal{P} + \mathcal{S} + 0.5\mathcal{E}$ instead of using the rounded stored composite. For each $X \in \{\mathcal{P}, \mathcal{S}, 0.5\mathcal{E}\}$, remove that source within the original window and recompute the sum-normalization:

$$q_i^{-X} = \frac{\mathcal{W}_i - X_i}{\sum_j (\mathcal{W}_j - X_j)}, \qquad \mathrm{TV}_X = \frac{1}{2} \sum_i \left| q_i - q_i^{-X} \right|. \tag{27}$$

The unit floor and all other normalized components remain fixed. All 315 windows share the same saved $\mathcal{P}$ map; its removal is not removal of a separate learned task-specific prior.

For panel a, average tied ranks separately within each window and divide each rank by 1,920. The horizontal coordinate is the $\mathcal{E}$ rank and the vertical coordinate is the $\mathcal{S}$ rank. Hexagonal binning with grid size 18 reports the arithmetic mean of $\Delta_i = 1{,}920(q_i - q_i^{-\mathcal{S}})$ over positions in each displayed hexagon. Color measures signed change relative to the uniform per-position coefficient $1/N$, not a ratio to $q_i^{-\mathcal{S}}$ or a density count. The diverging scale is symmetric about zero and covers all displayed means; no smoothing is applied. The native changes sum to zero in every window; recovering this property from bin means requires weighting by bin counts, not an unweighted mean of the displayed hexagons. Pooling preserves equal window representation overall, but the conditional mean in a hexagon weights conditions by their number of positions falling there.

For panel b, TV_X equals the sum of positive coefficient changes, the sum of negative changes in magnitude, and the largest absolute change in total coefficients assigned to any subset of native positions. Writing $\alpha_X = \sum_i X_i / \sum_i \mathcal{W}_i$ and $p_i^X = X_i / \sum_j X_j$ for $\sum_i X_i > 0$ gives

$$q - q^{-X} = \alpha_X (p^X - q^{-X}), \qquad \mathrm{TV}_X = \alpha_X \, \mathrm{TV}(p^X, q^{-X}). \tag{28}$$

Thus removal sensitivity combines the source's summed coefficients with its normalized spatial shape. It is neither a causal importance estimate nor an additive decomposition across sources. If $\sum_i X_i = 0$, removal instead leaves the normalized coefficients unchanged and $\mathrm{TV}_X = 0$. Compute TV on each window, take its median across nine windows per condition, and show all 35 condition medians. The cross-condition median [Q1,Q3] in percent is 5.4707 [5.2891,5.7790] for $\mathcal{P}$, 2.2159 [1.9935,2.5100] for $\mathcal{S}$, and 1.1405 [1.0422,1.2434] for $0.5\mathcal{E}$. Each density is normalized to its own peak. Density smoothing is display-only; the marks are descriptive IQRs, not confidence intervals. These arithmetic removals hold model outputs fixed, do not retrain a policy, and measure neither gradient magnitude nor success rate.

Panel c instead uses raw float32 $\mathcal{S}$ from one phone-packing validation demonstration, at window starts 63, 126, and 188. These windows use the physical-image diagnostic model and data described in Appendix F. Spatial means of raw $\mathcal{S}$ are normalized over the four native future bins using the implemented 10^{-6} denominator floor. Their temporal ranges are respectively 3.6025×10^{-6}, 5.6887×10^{-6}, and 1.8406×10^{-6}, so the floor is inactive in all three retained windows. The display shows four bin weights, each repeated over four consecutive action steps, alongside the exact unit reference when $\mathcal{S}$ is removed. It does not use a normalized-map temporal proxy or 16 independently probed actions. Because $\mathcal{P}$ is time-constant and this branch excludes $\mathcal{E}$, removing $\mathcal{S}$ leaves unit temporal coefficients. The same action perturbation thus provides spatial and temporal selectivity without a second coefficient predictor. The displayed w precedes the objective's normalization $q_j^a = w_j / \sum_k w_k$; even its unit reference is not a retrained ablation. The companion reproducibility materials retain the source correspondence and display specification.

Diagnostic conditions. Table 5 identifies the 35 conditions used for the response and error analyses in Figure 2. Each condition contains nine demonstration windows.

Priority-set flow and condition raincloud. In each simulation window, top-20% membership is defined on all 1,920 native positions, giving 384 effective members in either map. Boundary ties receive fractional membership. For memberships $a_i, b_i \in [0, 1]$ in $\mathcal{E}, \mathcal{S}$, the four joint masses are the position means of $(1 - a_i)(1 - b_i)$, $(1 - a_i)b_i$, $a_i(1 - b_i)$ and $a_i b_i$, followed by an equal mean over all 315 windows. These fractions are 0.709583, 0.090417, 0.090417 and 0.109583, respectively. Figure 2d conditions on response-top20 membership: dividing the shared and response-only masses by 0.2 gives the displayed 54.7916% and 45.2084% bars. Each percentage uses the

Table 5: Task identities for the 35 diagnostic conditions.

Index	Task
00	Arrange largest number
01	Build tower
02	Classify objects
03	Cover blocks
04	Deposit coin
05	Spell RoboDojo
06	Fasten screws
07	Fill egg holder
08	Fill pen holder
09	Fold clothes
10	Hang mugs
11	Imitate sorting sequence
12	Insert key
13	Insert tubes
14	Make kong
15	Make toast
16	Match + pick from conveyor
17	Organize table
18	Pack objects into box
19	Play xylophone
20	Play stacking toy
21	Play tic-tac-toe
22	Plug in charger
23	Pour balls into vase
24	Pour liquid into cup
25	Press by number
26	Push T
27	Put bottles in dustbin
28	Sort nesting dolls by size
29	Stack blocks
30	Stack bowls
31	Store laptop + headphones
32	Swap T pieces
33	Swap blocks
34	Sweep blocks

response top set as its denominator. Shared mass remains 10.9583% of all positions. The overlap compares signals within the same window.

Raincloud points use the within-condition median of each window's Spearman rank correlation. The cloud density is normalized to its own peak. The cloud and median/IQR marks describe 35 curated condition medians, not 315 independent evaluations. The separately summarized overlap median 0.559896 differs from the alluvial mean 0.547916. Distribution smoothing in the cloud is a display device, not an uncertainty model; no confidence interval, significance test, or success gain is inferred.

Figure 2c labels the three original phone-packing windows (starts 63, 126, 188) as W1, W2, W3. The five-panel arrangement retains the original underlying statistics.

Normalized auxiliary weight share. Appendix E evaluates $\sum_i b_i \mathcal{W}_i / \sum_i \mathcal{W}_i$ on the same windows under uniform $\mathcal{W}_i = 1$, post-hoc error-only $\mathcal{W}_i = 1 + 0.5\mathcal{E}_i$, and full response-based credit $\mathcal{W}_i = 1 + \mathcal{P}_i + \mathcal{S}_i + 0.5\mathcal{E}_i$ rules. The saved component maps are normalized; full weights are reconstructed from them rather than taken from float16-rounded stored $\mathcal{W}$. Each point is a median over nine windows in one condition, with median/IQR marks across 35 conditions. The respective medians are 20.00%, 20.9448% and 24.4003%; the error-only and response-based credit IQRs are 20.7050–21.2332% and 23.0325–25.3654%. Every condition-median share increases; the median paired difference is 3.7816 percentage points (range 0.8839–5.7004), not the difference of group medians. Adding both $\mathcal{P}$ and $\mathcal{S}$ prevents attribution to either signal alone; enrichment partly follows from including $\mathcal{S}$ in both the weights and the selected set. This measures normalized auxiliary coefficients, not gradient magnitude, task gain, or a retrained error-only control.

Figure 7 shows that the full credit rule assigns a larger weight share to response-top20 positions than error-only reweighting in every displayed condition. The medians across conditions are 24.40% and

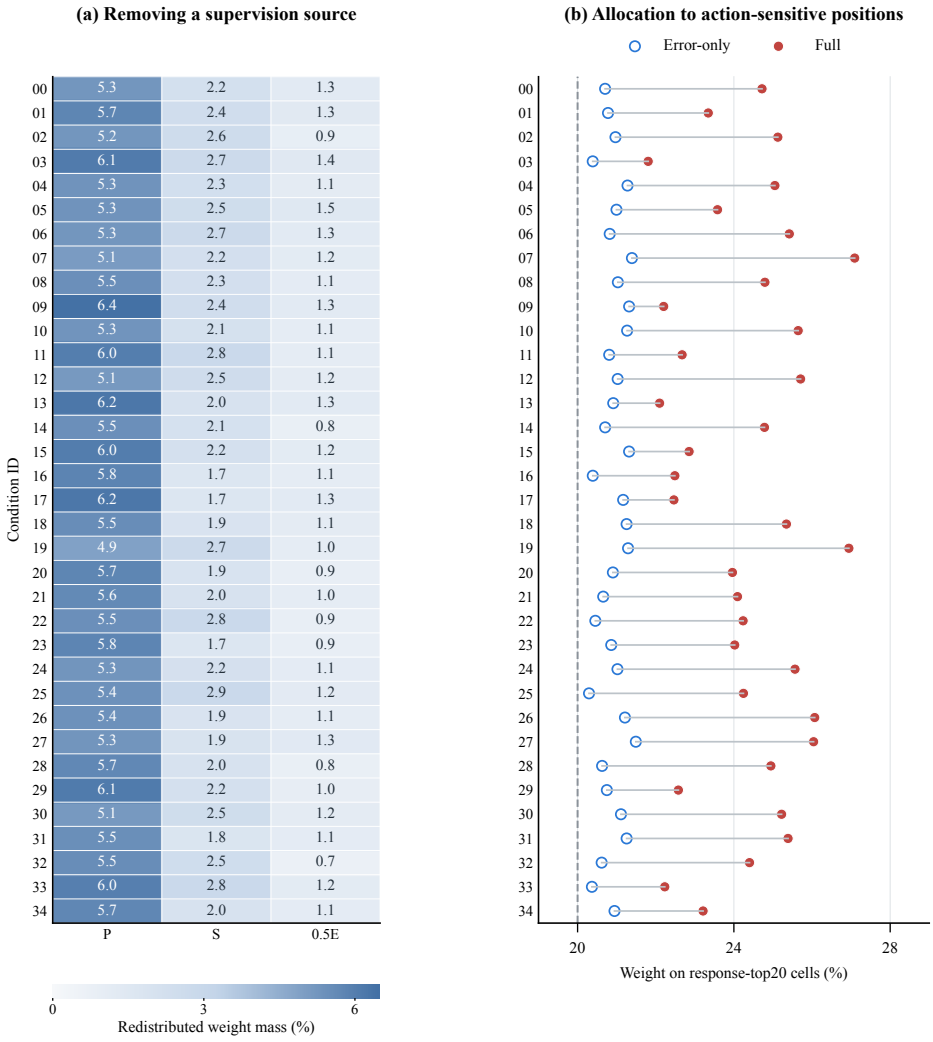


Figure 7: **Condition-resolved supervision allocation.** Both panels retain all 35 conditions, each with nine windows; IDs follow Table 5. (a) Median total-variation distance after removing each source from the normalized spatial weights, in percent. (b) Median normalized weight mass assigned to response-top20 positions under error-only reweighting (blue) and the full credit rule (red, labeled Full); each connector pairs the same condition. The dashed line is uniform allocation, 20%. The fields and predictions are fixed throughout.

20.94%, respectively. This supports the mechanism's ability to alter supervision beyond error-only weighting; it does not measure a retrained success-rate gain. The source-removal distances also vary across conditions, showing that the same coefficient rule induces different allocations for different inputs. The 315 windows are descriptive samples, not independent policy evaluations; no confidence intervals or significance claims are inferred from these plots.

F Spatiotemporal Diagnostic Protocol

The simulation analysis uses 35 conditions with nine distinct demonstrations each. Conditions are defined using initial image, state, and logarithmic trajectory length; each demonstration contributes one middle valid window and one seeded draw of diffusion noise and action perturbation. Each map contains four future bins on a 24×20 grid. Within-window ranks use all 1,920 positions;

condition summaries aggregate nine windows. Pixels and windows from the same condition are not independent policy evaluations.

The temporal-credit example in Figure 2c uses a phone-packing demonstration with window starts 63, 126, and 188. Its raw response maps have four 14×14 bins; spatial averaging and the normalization in Equation 8 produce the displayed action weights. Figure 3 instead uses the three-can tray-placement run, with checkpoint and window details below. Both analyses use recorded demonstrations, not autonomous physical success trials.

Three-timescale display and component accounting. Panel a of Figure 3 reads the native 14×14 EMA prior directly from the same can-placement run at steps 690, 4140, 8280, 12420, 16560, and 20700. Percentages are relative to the final step. The upper row selects steps 690, 12420 and 20700, independently normalized to show spatial shape. The lower row reports the arithmetic mean of the raw stored EMA at all six checkpoints on a logarithmic axis. The curve measures the training prior rather than held-out prediction error. The final prior is the one used in panels b–c. Panels b–c use the verified step-20,700 can-placement checkpoint and recorded training demonstration 8574282; these are not physical executions. Panel b retains start 112 (RGB offsets 114, 118, 122, 126), selected after screening 32 archived windows. Its square crop is $x \in [48, 176)$, $y \in [96, 224)$; top40 membership and transitions are computed over the full 14×14 grid before cropping. Thus the crop can show fewer than 40 selected cells. The archived full-field tensors retain background responses.

Panel c instead includes all eleven exported windows, starts 0,16,. . .,160, without localization-based selection. For each window let $C_j \in \{\mathcal{P}, \mathcal{S}, 0.5\mathcal{E}\}$ and $\mathcal{W} = 1 + \sum_j C_j$. The upper row reports $100 \sum C_j / \sum(\mathcal{W} - 1)$, with sums over all four future groups and all 196 spatial positions. The unit floor is excluded only from this composition ratio. The lower row normalizes $\mathcal{W}$ and $\mathcal{W} - C_j$ separately over the same positions and reports $100\,\mathrm{TV}$, where $\mathrm{TV}(p, q) = \frac{1}{2} \sum |p - q|$. The unit floor is retained in both distributions; no remaining component is recomputed or renormalized before this final distribution normalization. This measures sensitivity of the allocation rule at fixed outputs, not the effect of training without a component. Shared row scales are 0–100% and 0–5%; lower-row points denote available windows and lines connect adjacent observations. Upper-row stacked shares are linearly connected between adjacent windows.

Component share measures weight magnitude, whereas removal TV measures its effect on the normalized allocation. These complementary statistics describe one demonstration at fixed model parameters.

G Scope of Physical-Robot Evidence

Sixteen task videos provide recorded manipulation sequences. Tasks 04, 06, 08, and 11 appear in Figure 4; Figures 8–10 cover the twelve other recorded tasks. Each row contains four chronologically ordered frames from one video: initial configuration, task-specific intermediate events, and a visible terminal configuration. Selection follows actions rather than equal time intervals; timestamps are source-video times. Task 15 places a hamburger in a takeaway box and closes the lid; its final frame shows both arms returned. Task 16 packs a phone into its box and fits the lid. This supplementary task is excluded from the fifteen-task main comparison.

All 64 frames are decoded directly from the sixteen original videos at the reported timestamps. Tight opaque masks cover incidental packaging and equipment text, including chest branding; additional masks cover the visible torso while preserving foreground arms and hands. Task-defining numerals and mahjong symbols, timestamps, and stage labels are retained. Original cloth marks, object details, and robot poses are retained. Display frames are downsampled and JPEG-compressed; no generative editing or inpainting is applied. Original videos and unmasked extracted frames are retained.

Supplementary phone packing. For packing a phone into its box and closing the lid, the supplied results report SRs of 5% for $\pi_{0.5}$ and 85% for Ours.

H Post-training Implementation and Evaluation

Agent implementation and experimental workflow. Completed task results and failure evidence are supplied to an improvement agent based on Claude Opus 5 or GPT 5.6 Sol. The agent returns

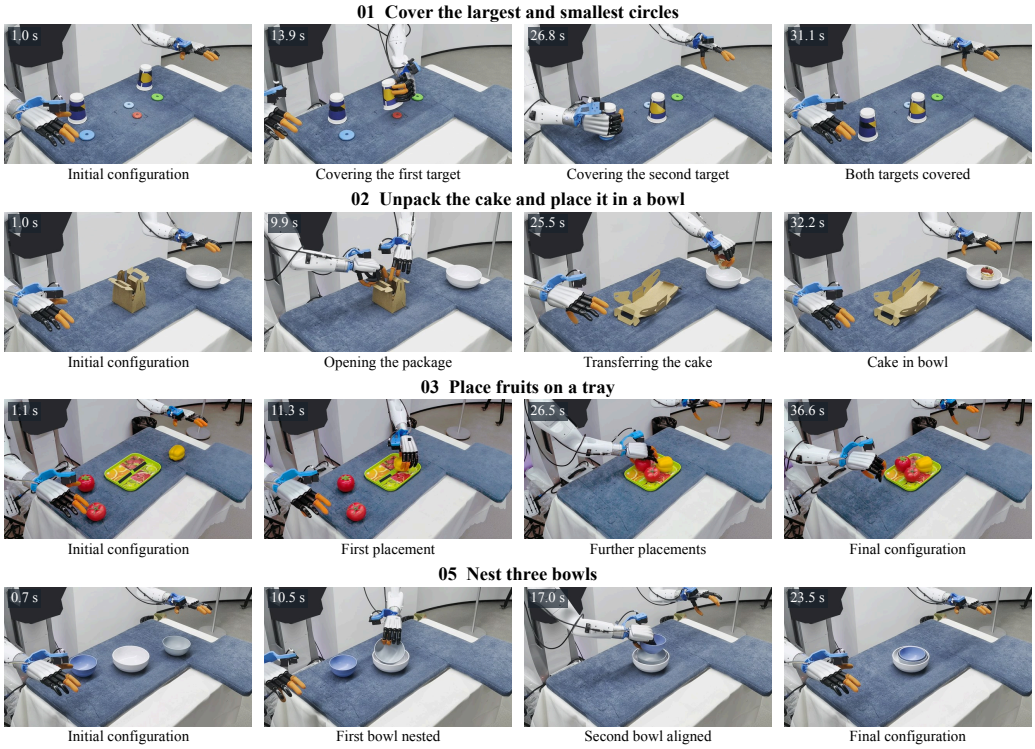


Figure 8: **Additional recorded task sequences (I).** Tasks 01, 02, 03, and 05 show circle-size selection, cake unpacking and placement, fruit placement, and bowl nesting. Frames are event-aligned within each source video; torsos and incidental text are masked before display downsampling and compression.

a post-training proposal specifying the parent policy, target behaviors, training-data allocation, and parameter-update scope. Training produces a candidate for task-level evaluation. Improved task capabilities enter the repertoire; unsuccessful candidates are excluded from active execution. Evaluation outcomes feed the next proposal, linking policy learning to repeated experimental decisions.

Evaluation object. The full system combines task-specialized policies developed on the RAPWAM substrate. Its task-to-policy assignment is part of the evaluated system specification. SR and Score for a task refer to the same policy execution record; parameters and within-episode action fragments are not averaged across policies. The numerical source manifest preserves task identities, policy identities, and original summary paths.

Aggregation. The complete system has Generalization SR/Score of 10.50/14.375, Precision 21.0255/28.6617, Long-Horizon 36.5000/48.4318, Memory 9.3333/10.3167, and Open 0.75/1.1625. Applying Equation 12 gives 15.6218% SR and 20.5895 Score, rounded to 15.62% and 20.59 in the main text. Table 1 shows Generalization's standard and random columns separately, reconstructed from the corresponding task-policy records. The local base comparison uses identical capability denominators.

Inference configuration. Evaluation uses 30 denoising steps, replanning every eight actions, nine-step Savitzky–Golay smoothing, and four-action boundary blending. Each task assignment identifies the policy architecture, checkpoint, normalizer, and inference path.

Component comparisons. Separating policy learning from agentic decision-making requires matched data, training compute, interaction counts, and retry budgets. The relevant comparisons are continued training without agent-guided intervention, refinement with history reset, and the

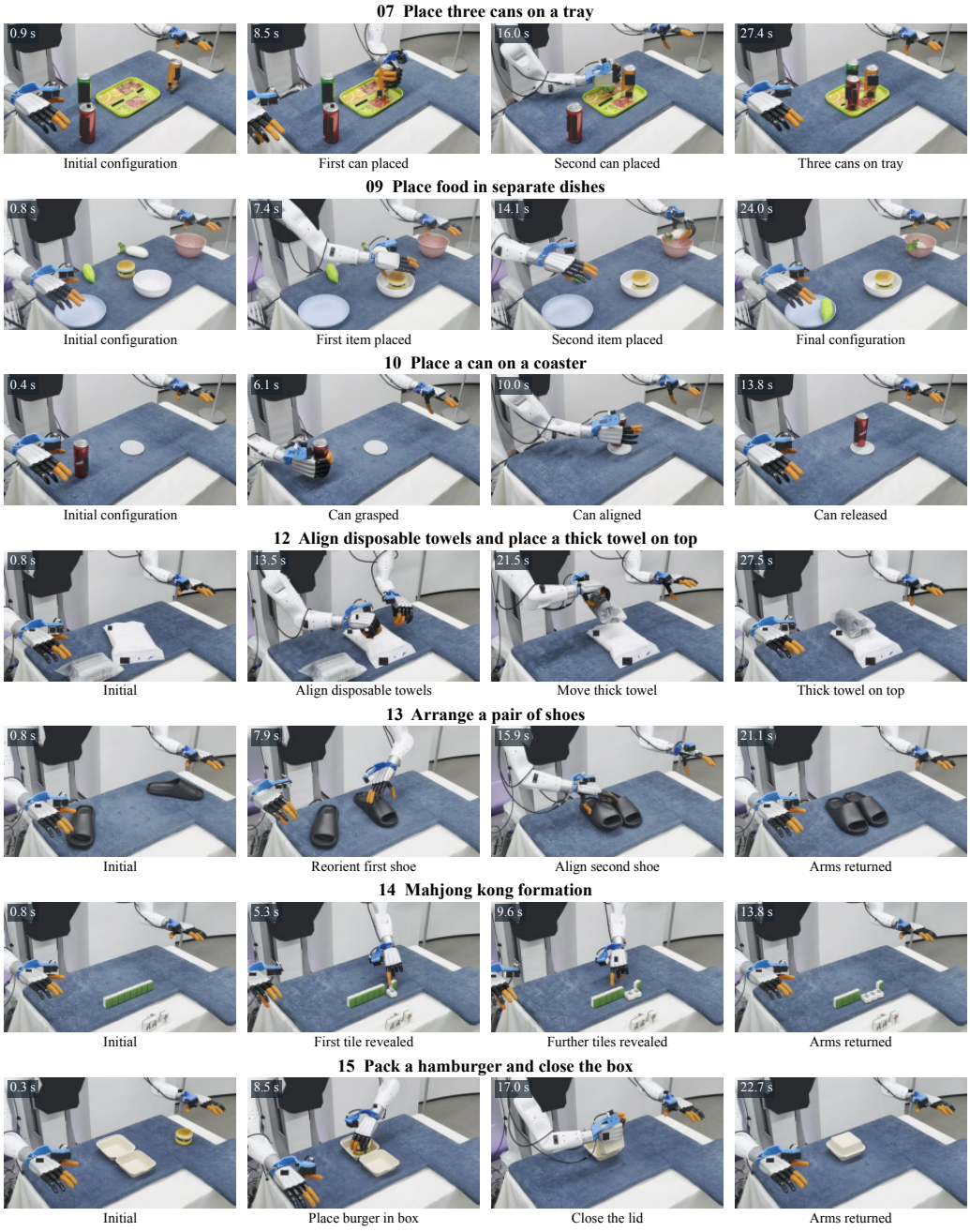


Figure 9: **Additional recorded task sequences (II).** Tasks 07, 09, and 10 show can placement, food placement, and coaster placement. Task 12 aligns disposable towels and places a thick towel on top; the torso and packaging text are masked; the recorded scene is otherwise retained. Tasks 13 and 14 show shoe organization and mahjong kong formation. Task 15 places a hamburger in a box and closes its lid, ending with both arms returned. Timestamps and frame-selection conventions follow Figure 8.

complete feedback loop. Removing response-based credit tests the local learning rule; resetting experiment history tests the role of cross-round evidence. The reported system comparison evaluates the combined procedure.



Figure 10: **Additional recorded task sequence (III).** Task 16 packs a phone into its box and fits the lid. The four original-video frames show the initial configuration, phone positioning, lid placement, and the final configuration with both arms returned. Torso and incidental packaging and equipment text are masked; timestamps refer to the source video.

Post-training implementation. In the continuation experiment with demonstration replay, we start from base step 742500, updating all action-generating parameters with AdamW and cosine decay. Learning-rate warmup lasts 2,000 updates; rehearsal, state-substitution, and credit schedules restart over 5,000 updates. Its immutable demonstration sampler uses 50% natural replay, 20% task-balanced replay, 15% reliable failure-boundary windows, and 15% late-progress windows. The source covers 35 training tasks; failure-boundary evidence is available for 21. Five virtual epochs correspond to ten source-dataset equivalents, with a final training step of 145,585. The final repertoire uses intermediate checkpoints at 12,500, 25,000, and 50,000 steps; the last training checkpoint is not automatically the retained policy for every task.

Task-level improvement. The final assignment retains the base for 14 of 42 tasks and uses 16 specialist checkpoints for the other 28, drawing on 30 recorded candidate identities. Nine specialists are obtained within 5,000 additional optimization steps, indicating that useful task-specific refinements can emerge from short training runs. Success improves on 24 tasks; four other substitutions improve Score at unchanged SR. The capability-balanced SR gain is 2.7697 percentage points, from 12.8520% to 15.6218%. These checkpoint budgets describe the retained solutions; total development cost also includes longer training branches and candidate evaluation.

The base serves 14 tasks, while 16 other checkpoint identities serve 28. Different specialists serve different capability groups. Assignments use the observed evaluation results, as specified in Appendix H.

Evaluation scope. The RAP-WAM base and post-trained repertoire use no external embodied pretraining. Task-policy selection uses observed evaluation results; the reported evidence does not include an independent held-out policy-selection test. Runs also differ in RNG and live-VAE metadata, so they are not paired stochastic replicates. This RoboDojo comparison evaluates integrated simulation performance. Real-robot results are reported separately in Figure 4.

Replay-intervention parameter values. The sampler uses $\tau_{\mathrm{rel}} = 0.15$, $b_{\min} = 0.08$, $b_{\max} = 0.22$, $\kappa = 0.5$, $\tau_{\mathrm{boundary}} = 0.30$, $K_{\mathrm{fallback}} = 128$, and $\tau_{\mathrm{late}} = 0.65$. The replay proportions are $(0.50, 0.20, 0.15, 0.15)$ in the order used in Section 3.4, giving $2N$ virtual windows. Learning rates are $\eta_{\mathrm{backbone}} = 10^{-5}$ and $\eta_{\mathrm{interface}} = 3 \times 10^{-5}$. AdamW uses betas $(0.9, 0.95)$ and weight decay 10^{-4} for decayed parameter groups, with zero decay for the excluded groups; warmup lasts 2,000 optimizer steps, followed by cosine decay. Candidate checkpoints are saved every 2,500 steps. These values instantiate the replay intervention used in the experiments.

Execution evidence to a training distribution. For each task, the execution profile records the fraction of failed episodes with positive partial-completion scores, together with a center c_t and quartiles $q_{25,t}, q_{75,t}$. Boundary replay is enabled for eligible tasks whose positive-failure fraction reaches a reliability threshold τ_{rel}. A demonstration window at offset k has normalized progress $p = k/\max(1, L - H)$, with $p = 1$ if no offset range is available; L is episode length and H the required sequence span. Set $b_t = \mathrm{clip}(b_{\min} + \kappa(q_{75,t} - q_{25,t}), b_{\min}, b_{\max})$ and proximity $w = \exp[-(p - c_t)^2/(2b_t^2)]$. The boundary pool contains windows with $w \geq \tau_{\mathrm{boundary}}$; an empty pool falls back to the K_{fallback} highest-proximity windows, capped by the available count. The

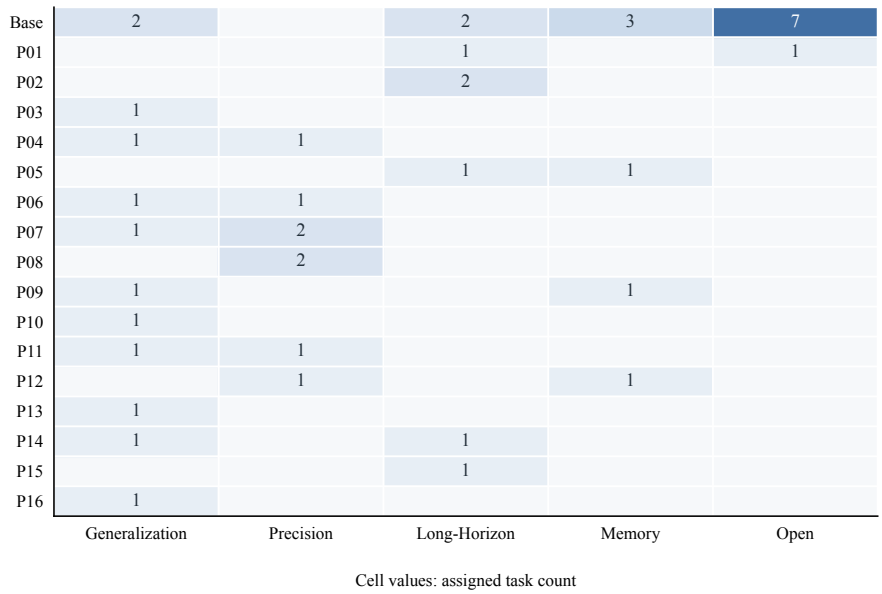


Figure 11: **Task assignments in the retained repertoire.** Each cell counts tasks assigned to the base or a retained specialist within one capability group; blank cells denote zero assignments. P01–P16 identify existing checkpoints, with identities recorded in the accompanying manifest. This is the final repertoire snapshot, not an iteration history or a policy-removal experiment.

late-progress pool uses $p \geq \tau_{\text{late}}$. This transfers a task-level score profile to demonstration progress as a sampling heuristic, rather than estimating a physical failure time.

Continued training and candidate generation. The training distribution combines natural replay, task-balanced replay, boundary replay, and late-progress replay with nonnegative proportions $\alpha_{\text{natural}}, \alpha_{\text{balanced}}, \alpha_{\text{boundary}}, \alpha_{\text{late}}$ summing to one. A virtual epoch preserves one complete pass over the source windows and adds the other pools in these proportions. Additional samples are balanced across eligible tasks and drawn uniformly within each task pool, with replacement only when the requested count exceeds pool size. Images and action labels remain unchanged. The selected parent is then optimized using the existing RAP-WAM objectives. The backbone and action interface use separate learning rates η_{backbone} and $\eta_{\text{interface}}$; the latter covers the action head, action/state encoders, query tokens, and projections. The optimizer runs within a specified update budget, saving candidates at a prescribed interval for evaluation. The replay-intervention parameter values above specify the thresholds, mixture proportions, optimizer configuration, and checkpoint schedule.

Per-task episode accounting. All reported aggregates use the same 42-task definition. Each task uses its valid evaluated layouts; the evaluation records contain 49 for Deposit coin and 44 for Fill egg holder, shared by the base and candidate policies. The source summaries and per-layout records were cross-checked; duplicate records were counted once. Candidate selection uses observed SR first and Score to break ties, with both metrics taken from one policy. Deposit coin uses P07 at 12,500 post-training steps (5/49 successes, Score 14.6939); Fill egg holder retains the base (0/44, Score 12.9545). Metrics are computed directly over these valid episodes. The unified base aggregate is 12.8520% SR / 17.8602 Score, and the system aggregate is 15.6218% / 20.5895, a gain of 2.7697 percentage points and 2.7293 Score.

The base has the highest aggregate SR and Score among the individual checkpoints in this candidate pool. Nevertheless, refinements improve subsets of tasks while regressing on others. The retained assignment combines these local gains and reaches 15.62% SR and 20.59 Score. This distinguishes a task-specialized improvement from a universal improvement of one checkpoint. All candidate results

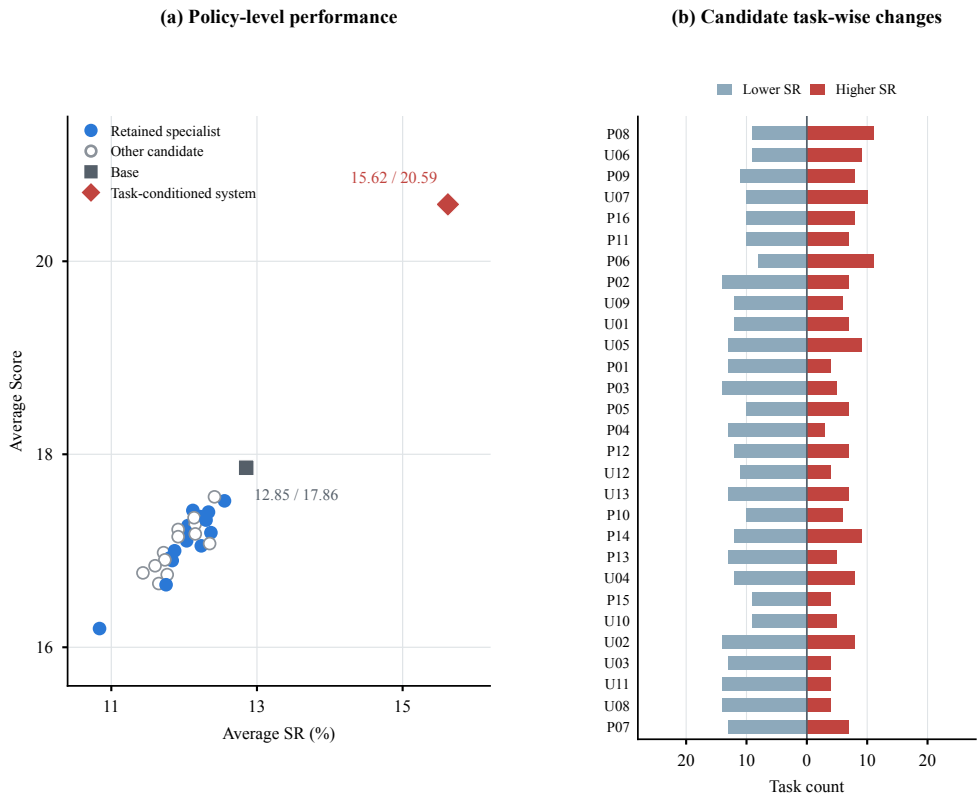


Figure 12: **Why retain task specialists?** (a) All 30 candidate checkpoint identities, evaluated using the same 42-task aggregation. Filled blue circles are retained specialists, open circles other candidates, the square is the base, and the diamond is the task-conditioned system. Axis origins are truncated to display the observed range. (b) For each of the 29 non-base candidates, tasks with lower SR (left) or higher SR (right) relative to the base; unchanged tasks are omitted from the bars. Candidates are ordered by average SR, with Score breaking ties. P identifiers match Figure 11; U identifiers denote unused candidates.

are shown, including candidates excluded from the repertoire. The comparison uses the observed results involved in policy selection; the system point is reconstructed from its task assignments, rather than an independent validation or evidence that the same gains persist under new seeds. Inference metadata differences are described in Appendix H.

H.1 Post-training and policy-retention details

Building on execution-feedback approaches (Xiao et al., 2026; Lu et al., 2026), the outer improvement loop uses experiment results to guide continued training, retain useful task specializations, and combine them through task-conditioned execution. Its state $X_r = (\mathcal{B}_r, g_r, \mathcal{H}_r)$ contains the policy repertoire, task assignment, and experiment history; the repertoire initially contains RAP-WAM. Training creates candidates, while evaluation and retention determine which capabilities remain available.

From execution evidence to continued training. At round r, completed experimental results are supplied to a language-model-based improvement agent. Its input $\mathcal{F}_r$ contains task outcomes, execution trajectories, and available model diagnostics; the agent uses these results to design the next post-training procedure. The agent compares weak task behavior with previous interventions in $\mathcal{H}_r$ to formulate a hypothesis about a learning bottleneck. It then specifies u_r: the parent policy,

target tasks or execution phases, the data or training-code change, and the trainable parameter set. This is a decision about the learning procedure; the policy optimizer carries out the resulting update: The intervention and optimization are defined by Equation 10. Agent-guided refinement yields multiple task specializations with only a few thousand fine-tuning steps, reusing existing policies and demonstrations. The operator $\mathcal{U}_{\mathrm{RAP}}$ updates designated parameters from a parent checkpoint while preserving that policy. The replay intervention below instantiates this process; implementation details and workflow provenance appear in Appendix H.

Evidence-guided continued training. The intervention mixes ordinary, task-balanced, failure-boundary, and late-progress demonstrations: $q_r = \sum_{j\in\{\mathrm{natural,balanced,boundary,late}\}} \alpha_j q_j$, with $\alpha_j \geq 0$ and $\sum_j \alpha_j = 1$. Boundary windows are selected by proximity to a score-derived task profile; reliability thresholds and fallback rules handle weak evidence or empty pools. Ordinary replay maintains demonstration coverage. All pools retain recorded observations and action targets. Training optimizes the RAP-WAM objectives from the parent checkpoint, with intermediate checkpoints evaluated as candidates. Appendix H specifies window selection, optimizer settings, and budgets. The score-to-progress mapping is a sampling heuristic, not a measurement of physical failure time.

Retention and task-conditioned composition. Task-wise comparison with the current assignment g_r produces a result vector $\mathcal{E}_r$ of success, partial completion, and observed failures. Let $\mathcal{C}_r$ contain the tasks on which the candidate is accepted. The comparison retains a task-level change vector $\Delta_t = (\Delta\mathrm{SR}_t, \Delta\mathrm{Score}_t)$ relative to the incumbent on each task, so improvements and regressions are not hidden by an aggregate mean. Both metrics must come from the same candidate execution record. Their assignments switch to the candidate; other tasks retain their policies. A candidate with an accepted improvement enters the repertoire; if $\mathcal{C}_r = \varnothing$, it is discarded from the active policy set and g_r is unchanged:

$$\mathcal{B}_{r+1} = \mathrm{Retain}(\mathcal{B}_r, \pi_r^+, \mathcal{C}_r), \quad g_{r+1} = \mathrm{Assign}(\mathcal{B}_{r+1}, \mathcal{C}_r, g_r), \quad \mathcal{H}_{r+1} = \mathcal{H}_r \cup \{(\mathcal{F}_r, u_r, \mathcal{E}_r, \mathcal{C}_r)\}. \tag{29}$$

The base and previously useful policies remain available. For task t and observation history $o_{\leq k}$, the system executes

$$a_k \sim \pi_{g_r(t)}(\cdot \mid o_{\leq k}, t). \tag{30}$$

The outer assignment selects a complete learned policy; credit coefficients govern supervision within policy training. Task-level comparison determines retention, with evaluation accounting specified in Appendix H.

Closing the improvement loop. History retains interventions, task-wise outcomes, and acceptance decisions, including rejected candidates. Later rounds can revise unsuccessful hypotheses and continue from useful specialists. Each evaluation is supplied back to the agent to design the next post-training round. The resulting loop is evaluate, propose, train, re-evaluate, and retain or discard. Discarding a policy does not imply discarding the experimental evidence used for the next decision.

I Additional Related Work

Generalist representations and action interfaces. OpenVLA adapts vision–language representations to robot actions (Kim et al., 2024). G0.5 places reasoning and action in one autoregressive stream (Liu et al., 2026), while Xiaomi-Robotics-1 scales action-labeled pretraining (Xiaomi Robotics Team et al., 2026) and Hy-Embodied uses modality-specific computation with a flow-based action expert (Zhang et al., 2026). Spatial Forcing supplies geometric alignment targets (Li et al., 2026). Our model learns its action interface on the target data. Its loss-coefficient rule uses the joint model's own outputs and existing future targets, rather than requiring an additional pretrained robot expert or geometric-label pipeline.

Training with predicted inputs and adaptive supervision. Self Forcing trains video diffusion on self-generated context with sequence-level supervision (Huang et al., 2025). Our rehearsal instead supervises an adjacent recorded segment and retains its demonstration targets.

Scheduled sampling and interactive imitation learning address discrepancies between recorded demonstration inputs and histories produced by policy execution (Bengio et al., 2015; Ross et al., 2011). Our

adjacent-segment training replaces observed inputs with predictions while retaining recorded demonstration targets; it neither collects new interactions nor inherits interactive data-aggregation guarantees. Prioritized replay and focal loss emphasize selected errors (Schaul et al., 2016; Lin et al., 2017); importance sampling targets estimator variance (Katharopoulos & Fleuret, 2018), and value-aware model learning uses decision structure (Farahmand et al., 2017). response-based credit combines persistent error with action response, and projects the same response into spatial and temporal supervision. RoboDojo and RoboTwin offer complementary manipulation benchmarks (Chen et al., 2026b; 2025); we report RoboDojo and real-robot task outcomes, alongside prediction diagnostics.

Synthetic trajectories and joint world action pretraining. UWM couples video and action diffusion with independent modality timesteps (Zhu et al., 2025); DreamZero jointly models video and actions to transfer across tasks and embodiments (Ye et al., 2026b). Motus learns latent actions from optical flow (Bi et al., 2026), and MotuBrain extends joint modeling to multiview and cross-embodiment settings (MotuBrain Team et al., 2026). DreamGen generates synthetic task videos and recovers pseudo-actions using latent-action or inverse-dynamics models (Jang et al., 2025). Our rehearsal instead changes the conditioning context while retaining recorded demonstration targets; response-based credit then redistributes supervision within that training. These approaches provide architectural and data-generation context, not controlled evidence for the effect of our credit rule.

J Recorded and Predicted Futures in Real-Robot Training

Figures 14–28 show one reproducible training example for 15 of the 16 real-robot tasks with external-view video sequences in Appendix G. Task 16 uses the completed Tencent Cloud phone-training run; its configuration, checkpoint, and sample identity were independently verified. Each example uses the final checkpoint of its audited task-training run. We select the median-length eligible continuous training segment and its middle legal two-chunk window, before inspecting predictions or attention. The archived exports used previously are replaced by these identified samples so that predictions and attention come from the same forward trajectory. Recorded targets belong to the same training sample as the prediction; the separately filmed external-view execution need not be the same episode. Each model receives only the conditioning observation and recorded current state at inference. There are no weight updates. The first column repeats the condition; offsets $t+1, t+4, t+8, t+12, t+16$ index the sampled future sequence, not seconds.

Prediction metrics. Every predicted frame is labeled with RGB SSIM (Wang et al., 2004); figure-level SSIM and LPIPS (Zhang et al., 2018) are arithmetic means over the five displayed future frames, excluding the condition. SSIM uses an 11×11 Gaussian window with $\sigma = 1.5$, population covariance, and channel averaging, excluding a five-pixel image border. LPIPS uses the official AlexNet network, version 0.1, at native resolution with inputs mapped to $[-1, 1]$. Both metrics use decoded float images clipped to $[0, 1]$, before privacy masks, quantization, or JPEG display compression. The orange dashed baseline ("Repeat current frame") uses the current conditioning image itself as the prediction at every future position, without running the world model. For current image I_t and recorded future I_{t+h}, its plotted error is $\mathrm{LPIPS}(I_t, I_{t+h})$. Thus it measures how much the recorded scene departs from remaining visually unchanged. Higher SSIM and lower LPIPS indicate closer images. We also retain metrics for all 16 predicted frames in the accompanying data.

Attention and local error. The bottom strip shows the condition with a fixed task-region rectangle, predictive-query attention, mean absolute RGB prediction error over the five displayed future frames, and the horizon-wise LPIPS curve. Rectangles were marked from condition images and checked for visible task coverage, independently of attention magnitude; they are coarse task regions, not object segmentations, and do not track moving objects. Region SSIM and MAE use the same rectangle in each future frame. A read-only probe records the actual self-attention query/key tensors after normalization and rotary encoding, retaining the model's additive mask and full softmax denominator. We average all 24 heads and 16 predictive-query rows at layers 10, 20, and 30, and denoising iterations 4, 8, and 16. We then display attention to the conditioning-time visual tokens. The spatial grid is 7×7, displayed without smoothing. Attention color is density relative to a uniform spatial distribution after conditioning on the displayed view, using a common 0–$4\times$ scale; error uses a common $[0, 0.15]$ scale. Values beyond either scale are clipped for display only. Region enrichment

is its fraction of attention divided by its fraction of image area; 1 denotes uniform allocation. Raw attention masses and layer/iteration maps are retained.

Scope of the evidence. Across these 15 selected examples, mean SSIM is 0.879 versus 0.814 for persistence, and mean LPIPS is 0.018 versus 0.095. Under the fixed aggregation, 1 of 15 task regions have attention enrichment above 1. Thus these maps do not establish preferential attention to task objects or accurate object recognition. They are descriptive internal diagnostics, not causal explanations. The examples come from training data and different task checkpoints; they do not establish held-out prediction quality, task success, or the separate contribution of response-based credit or agentic refinement. Incidental text and visible background faces are masked at paired locations; task numerals and mahjong symbols remain. Display editing is limited to deterministic masking, display downsampling, and compression.

Horizon-wise comparison. The fourth diagnostic panel in each task figure plots LPIPS over all 16 predicted frames, against the copy-condition baseline, using a common vertical scale within each domain. Figure 13 aggregates all 15 real-robot and 35 simulation windows with equal task and frame weights. Full-horizon mean LPIPS decreases by 81.8% for real robots and 86.3% for simulation, computed as one minus the ratio of the model and baseline grand means. Every task has a lower full-horizon mean, although individual frames can favor persistence; all frames and tasks are retained. These comparisons demonstrate improved future-image fidelity over repeating the current observation on the selected training windows. They do not isolate the contribution of response-based credit or compare against an independently trained base model.

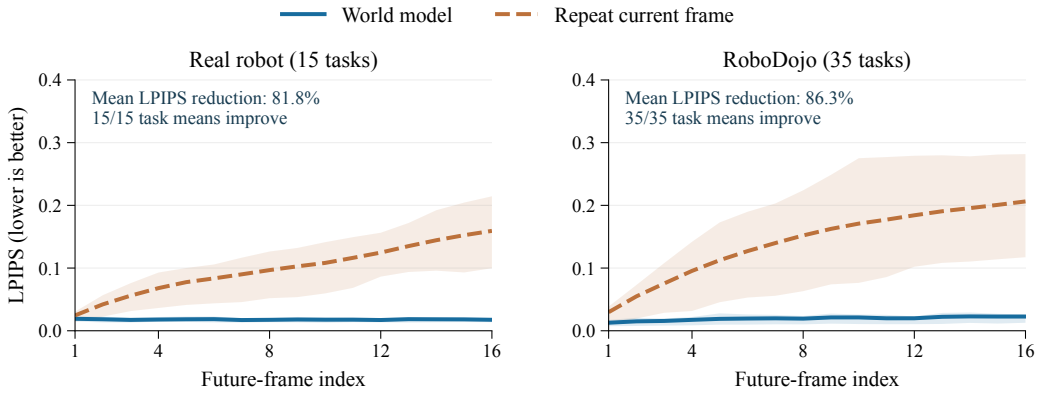


Figure 13: **Prediction error over the full future horizon.** Solid blue curves show the world model. Dashed orange curves show the LPIPS between the unchanged current image and each recorded future frame: the current image is reused as the future prediction, without model inference. Lines are task means, and shaded bands show the 25th–75th percentiles across the selected task windows, not confidence intervals. Both panels use the same vertical scale. Relative reductions use all 16 frames; the condition is excluded.

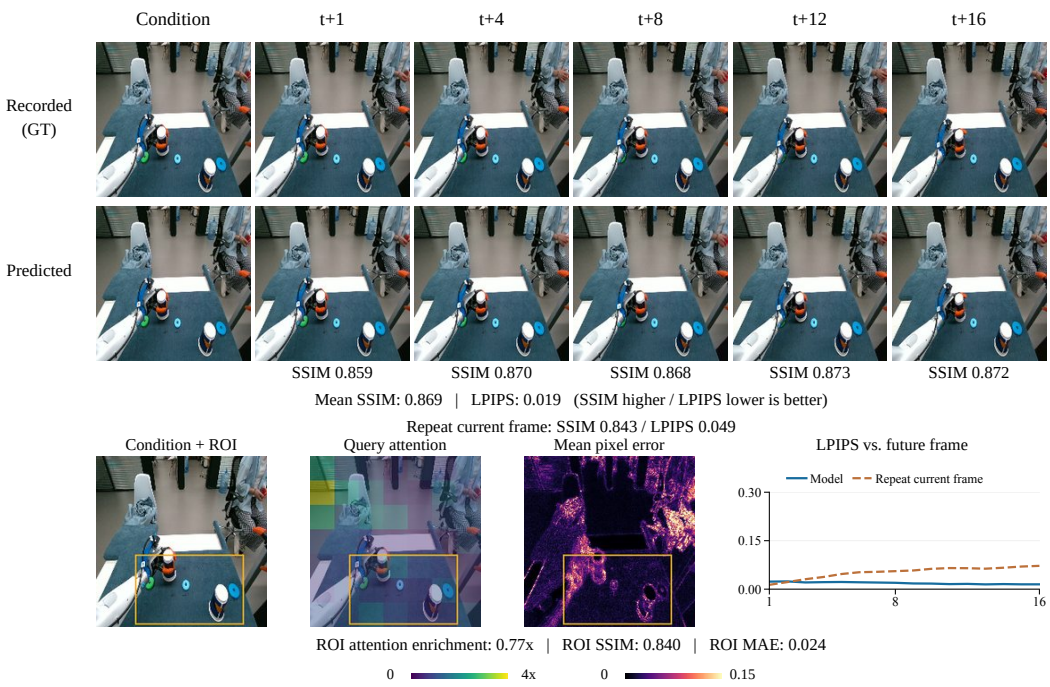


Figure 14: **Task 01: Cover the largest and smallest circles.**

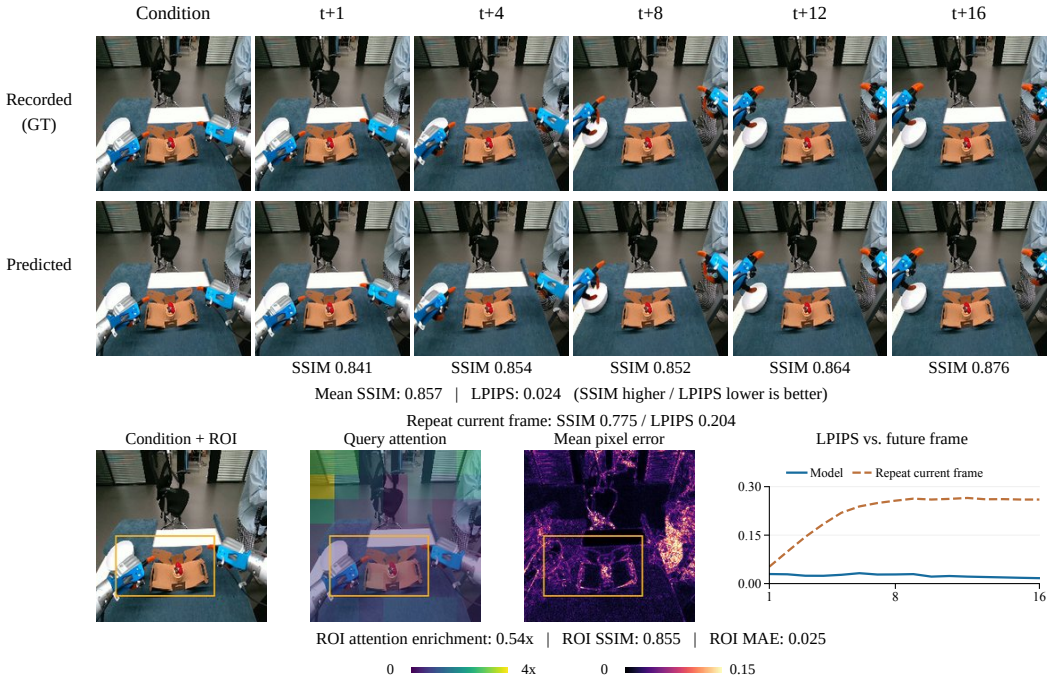


Figure 15: **Task 02: Unpack the cake and place it in a bowl.**

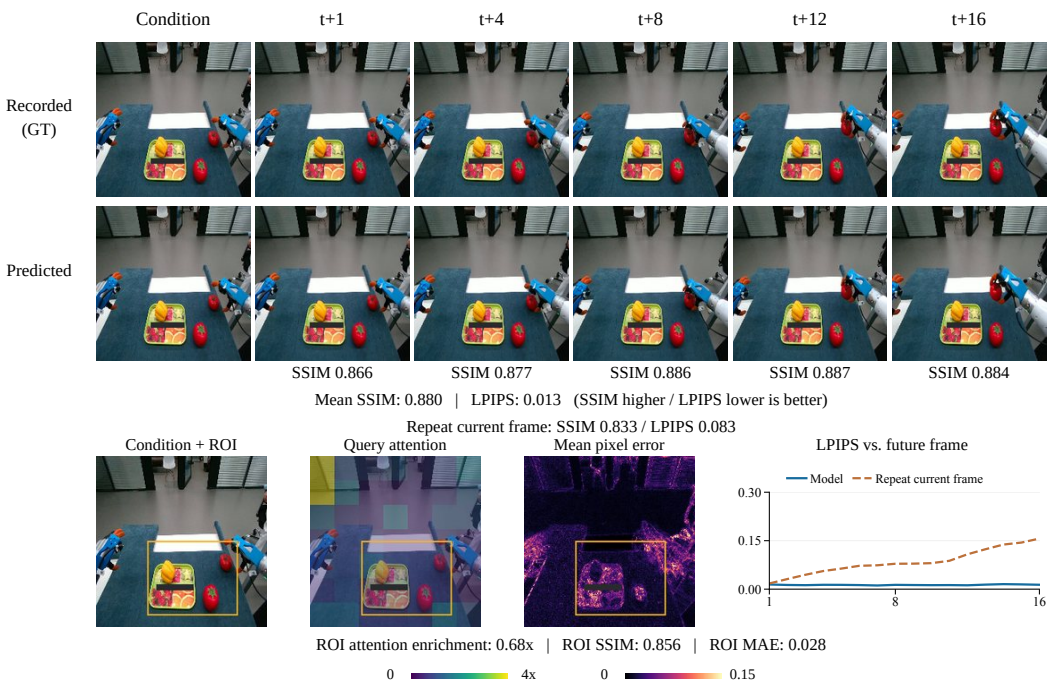


Figure 16: **Task 03: Place fruits on a tray.**

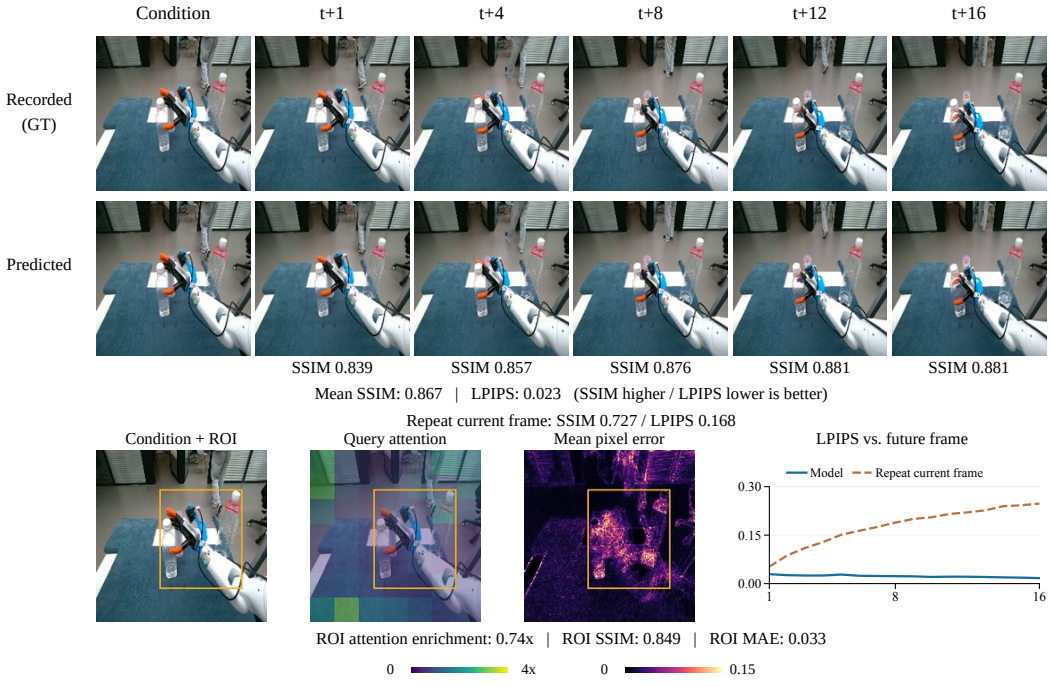


Figure 17: **Task 04: Reorder three different types of bottles by height.**

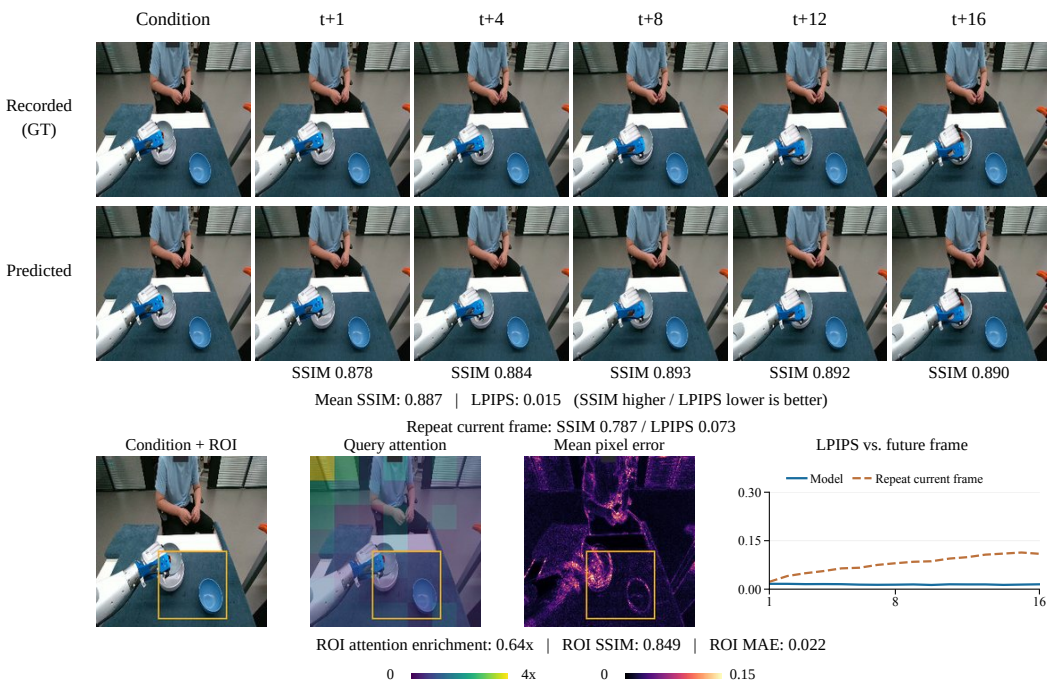


Figure 18: **Task 05: Nest three bowls.**

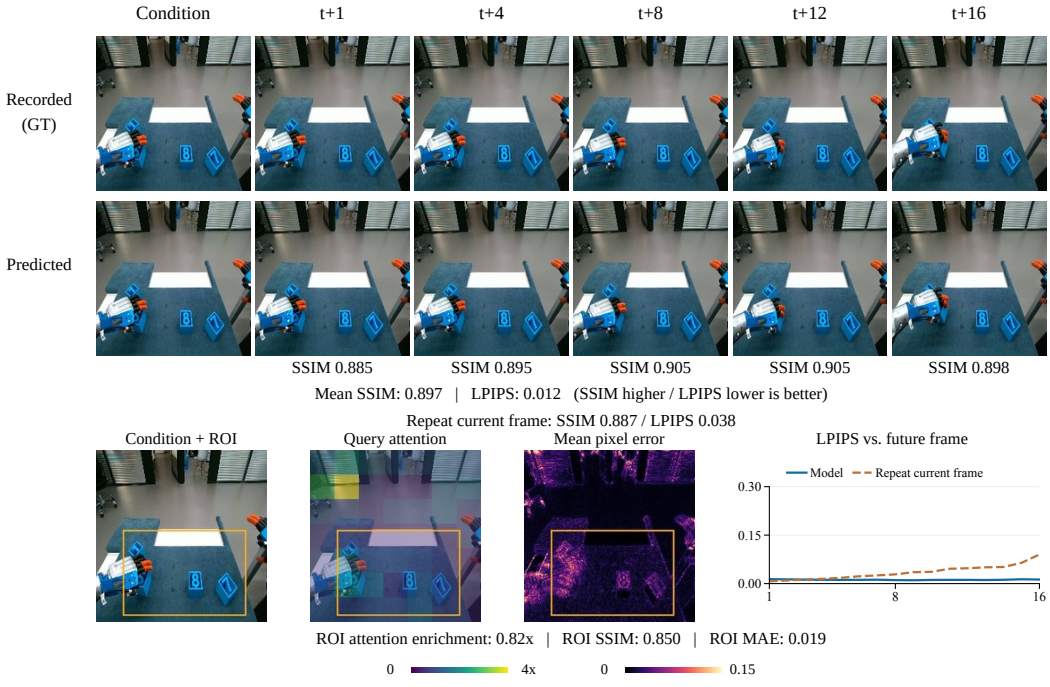


Figure 19: **Task 06: Rearrange numbered blocks.**

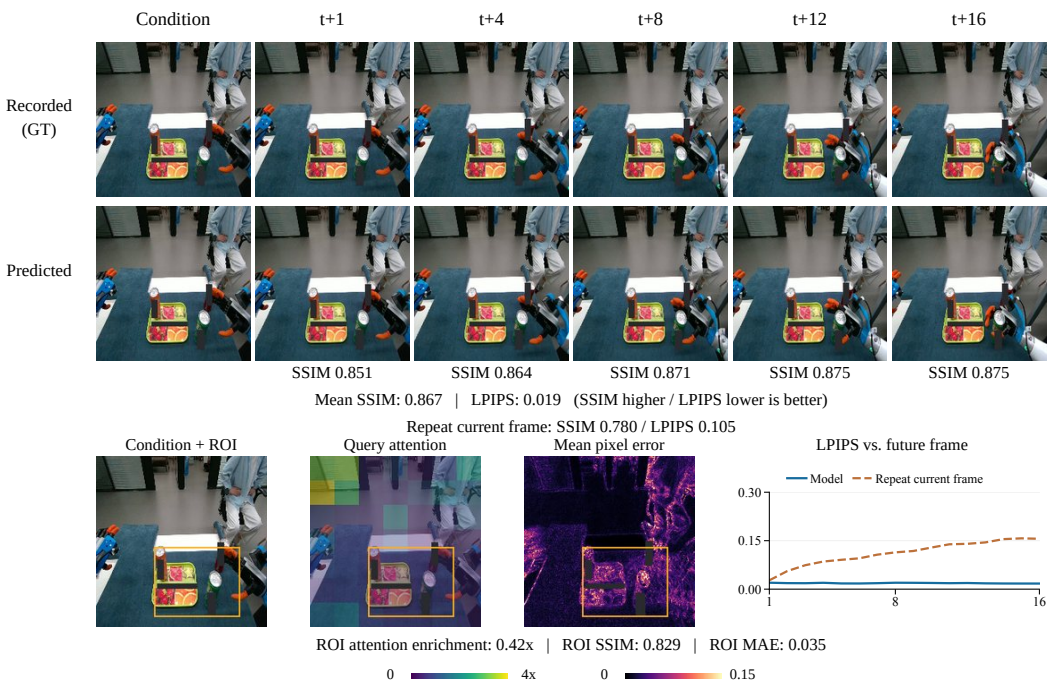


Figure 20: **Task 07: Place three cans on a tray.**

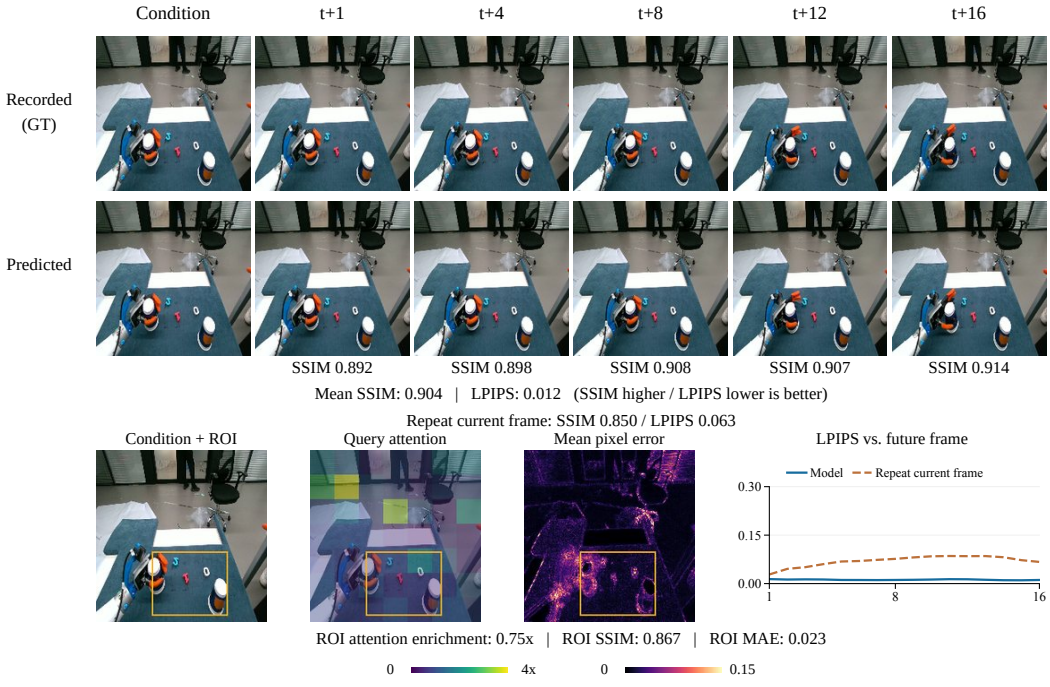


Figure 21: **Task 08: Cover the largest and smallest numbers.**

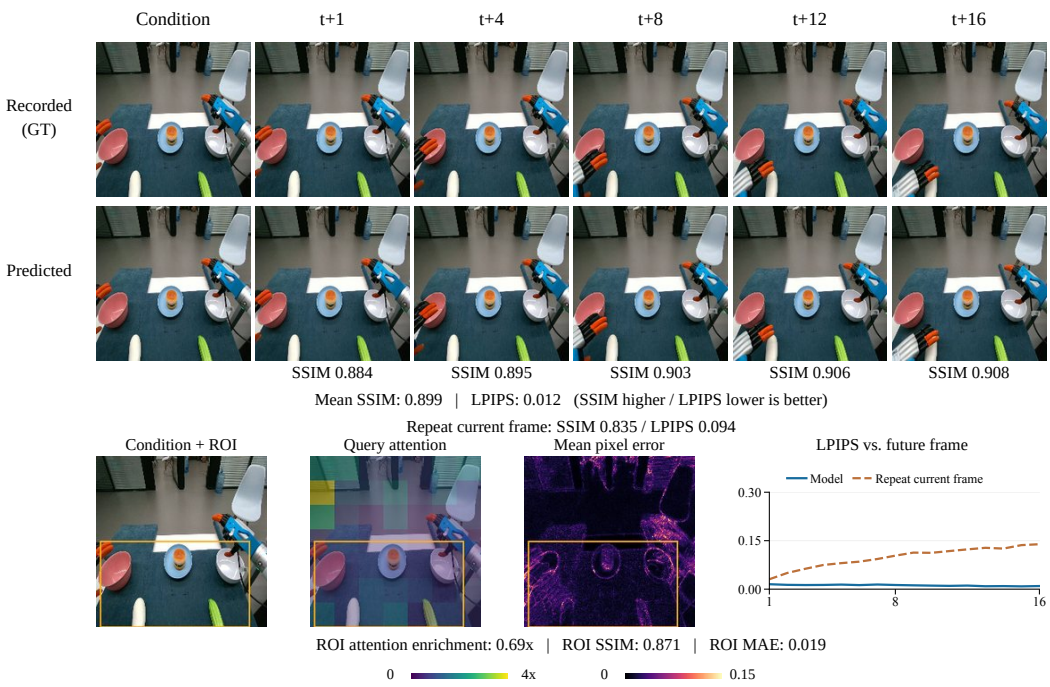


Figure 22: **Task 09: Place food in separate dishes.**

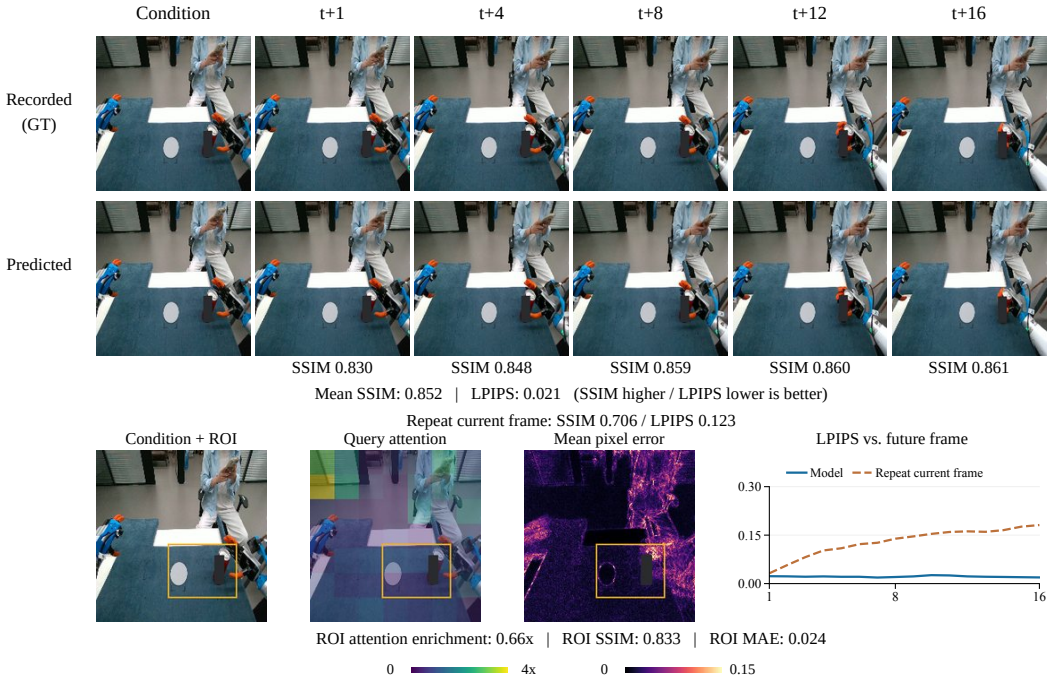


Figure 23: **Task 10: Place a can on a coaster.**

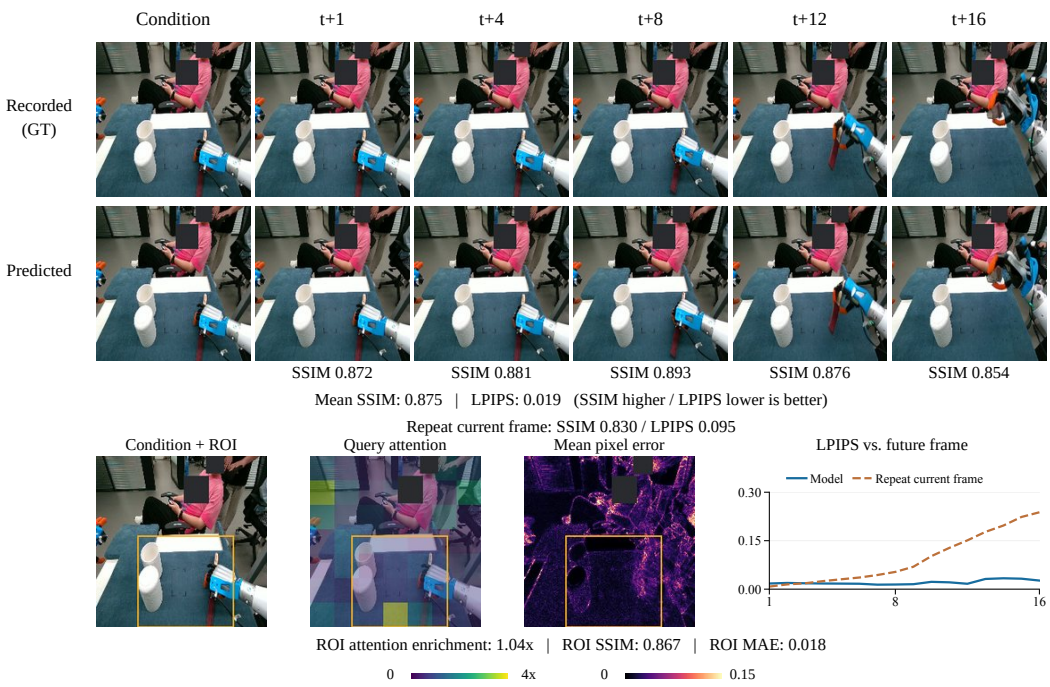


Figure 24: **Task 11: Insert a folding fan into an open-top vessel.**

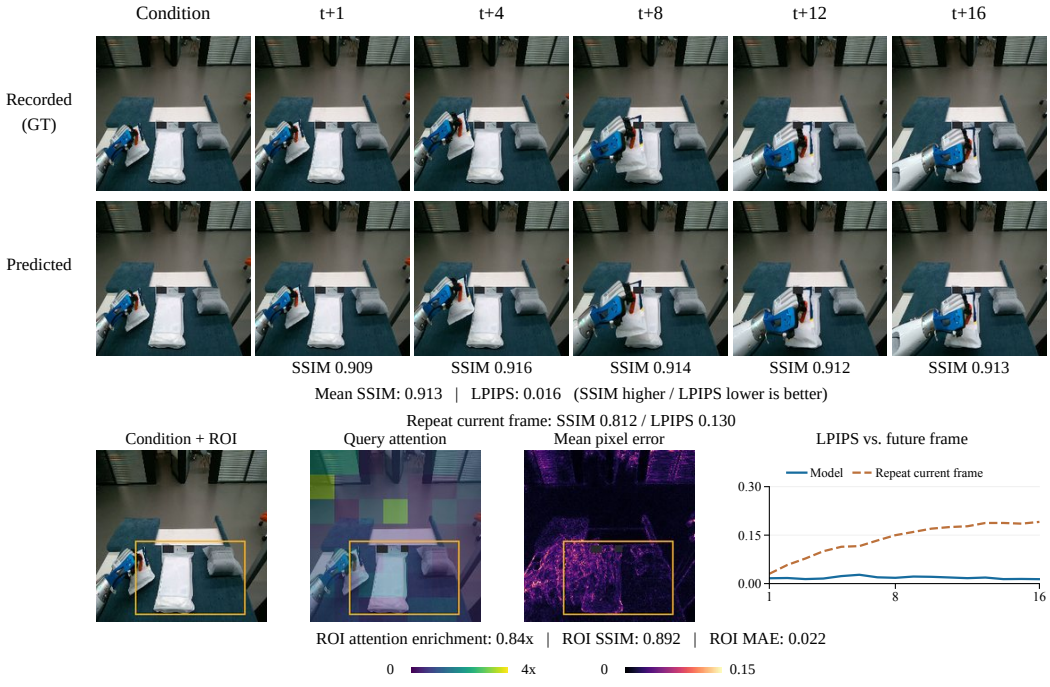


Figure 25: **Task 12: Align disposable towels and place a thick towel on top.**

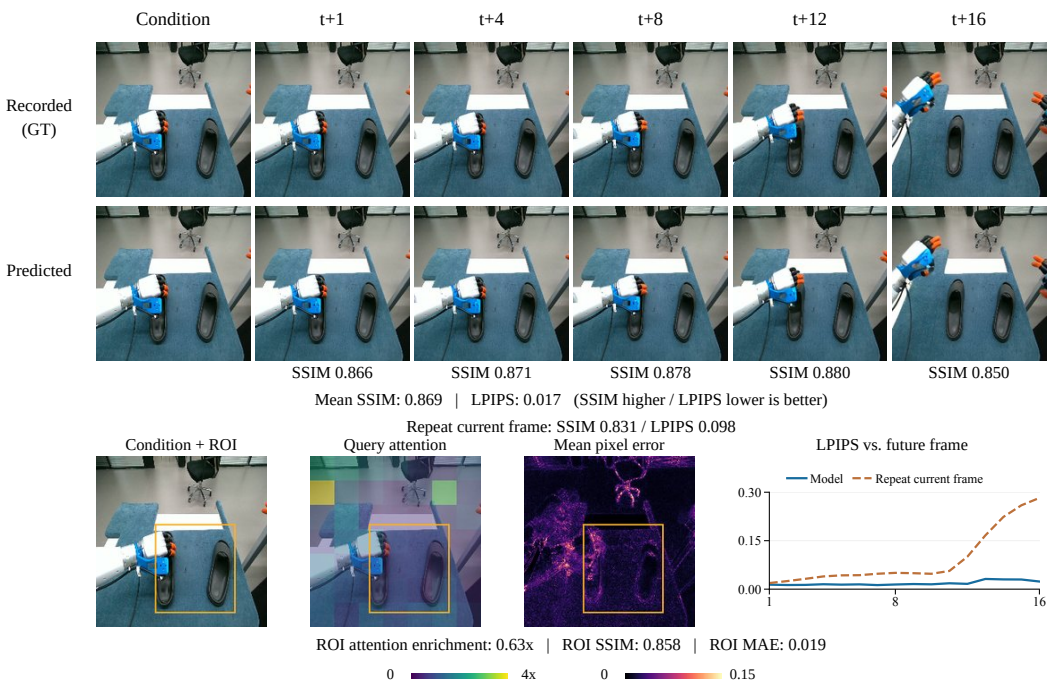


Figure 26: **Task 13: Arrange a pair of shoes.**

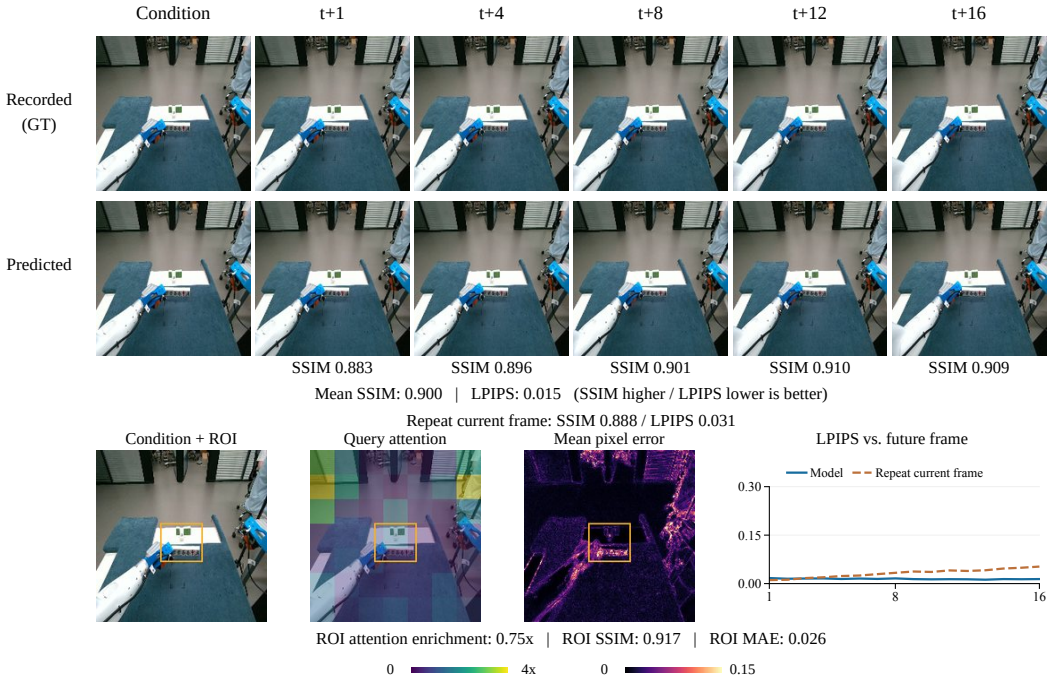


Figure 27: **Task 14: Mahjong kong formation.**

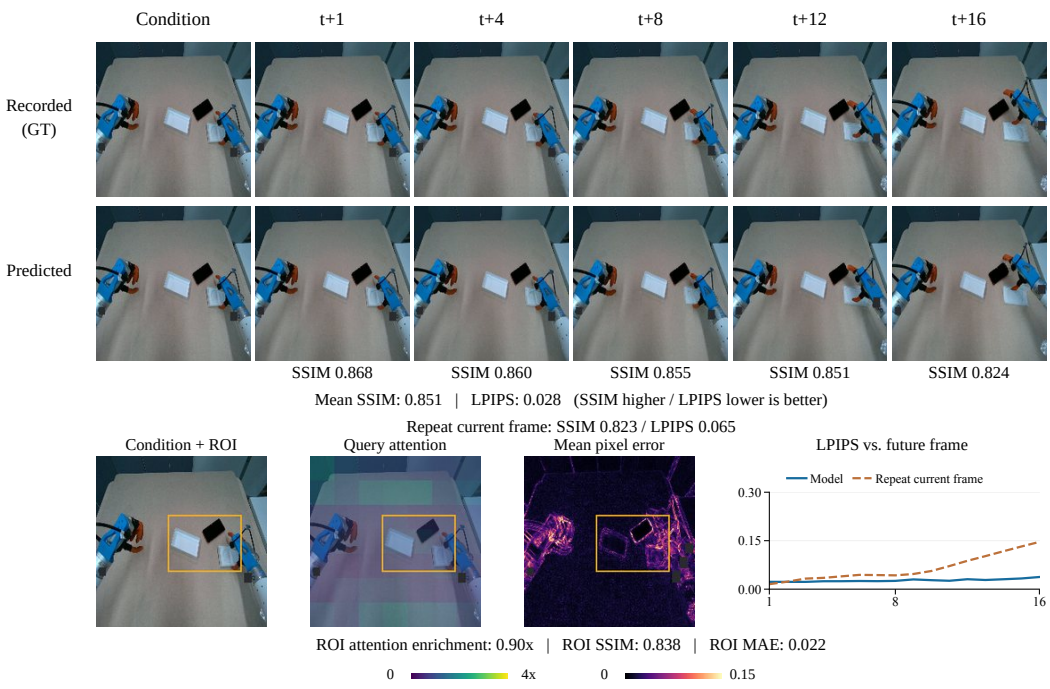


Figure 28: **Task 16: Pack a phone into its box.**

K Recorded and Predicted Futures in RoboDojo Simulation

Figures 29–63 show one identified training example for each of the 35 RoboDojo training tasks (task IDs 00–34). All examples use the retained post-training checkpoint P09 (step 50,000) described in Appendix H. We preserve the training loader's seed-42 validation exclusion, select the median-length eligible training episode per task, and use its middle legal two-chunk window. Task IDs are verified against dataset metadata, and the corresponding task-language embedding is explicitly supplied during inference. Each figure pairs the recorded future with a 16-frame prediction sampled using 16 denoising steps. Inputs are the conditioning image, current state, and task-language embedding. The 30-step setting in Appendix A applies to policy evaluation. The original head-and-two-wrist camera mosaic is retained; task numbers denote dataset IDs, not difficulty ranks.

SSIM, LPIPS, the persistence baseline, and the fixed attention aggregation follow Appendix J. Global similarity metrics use the full original 384×320 camera mosaic. The diagnostic bottom strip shows only the head view, the upper 256×320 region; region metrics and attention enrichment are restricted to that view. The original attention grid is 12×10 over the mosaic, with an 8×10 head-view subgrid. It is displayed without smoothing across camera boundaries, with the same attention and error color scales as the real-robot figures. Region rectangles were marked from conditions and checked for visible task coverage, independently of attention magnitude.

Across the 35 selected training examples, mean SSIM is 0.942 versus 0.828 for the copy-condition baseline, and mean LPIPS is 0.018 versus 0.134. SSIM improves on persistence in 34 of 35 examples and LPIPS in all 35. None of the 35 marked regions has attention enrichment above 1 under this fixed aggregation, so these maps do not support a claim of object-localized attention or object-recognition accuracy. Coverage of all training task types is not a held-out evaluation: one window per task cannot establish average trajectory fidelity, causal component gains, or rollout success. Incidental text is masked for display while task-defining numbers, letters, and mahjong symbols are retained; metrics are computed before masking and compression.

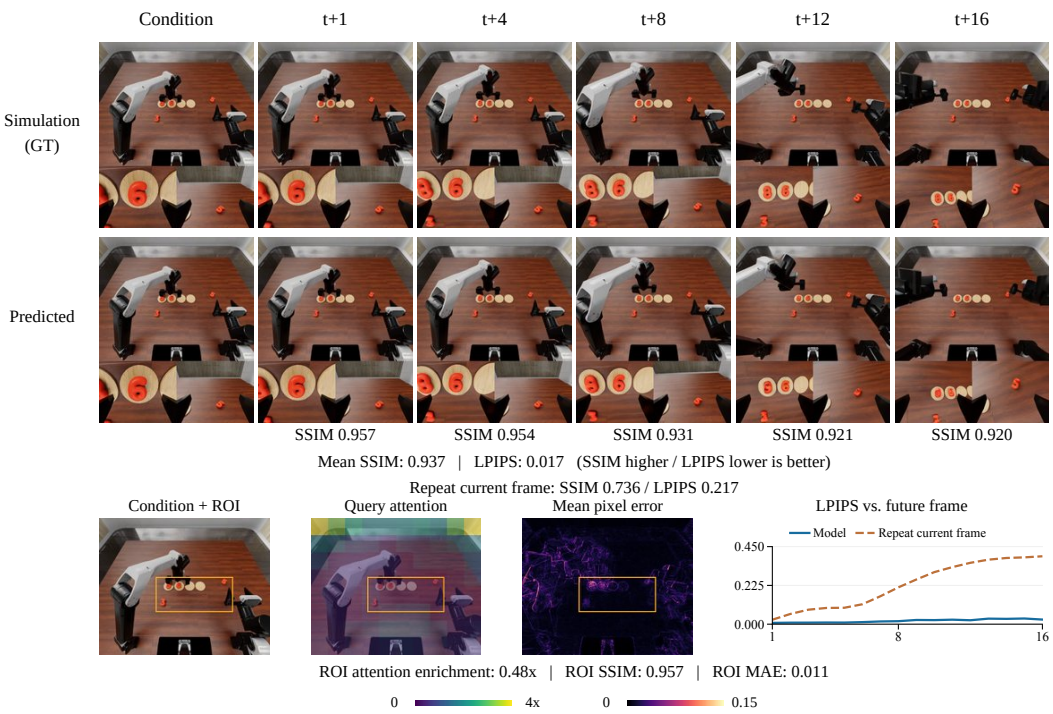


Figure 29: **RoboDojo task 00: Arrange numbers.**

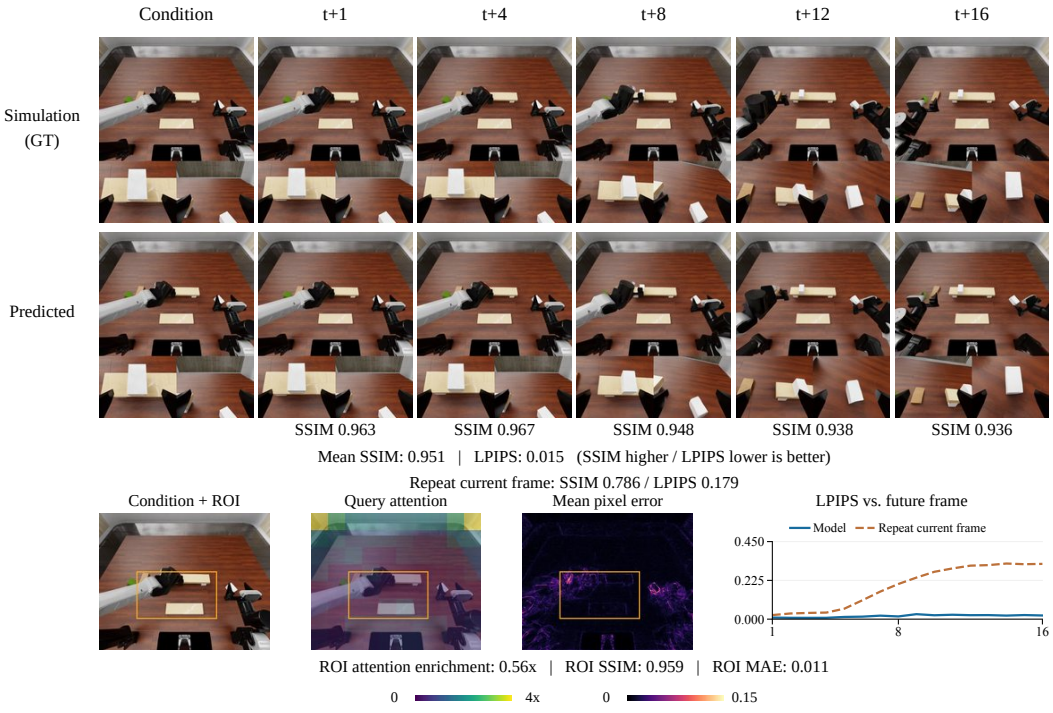


Figure 30: **RoboDojo task 01: Build a tower.**

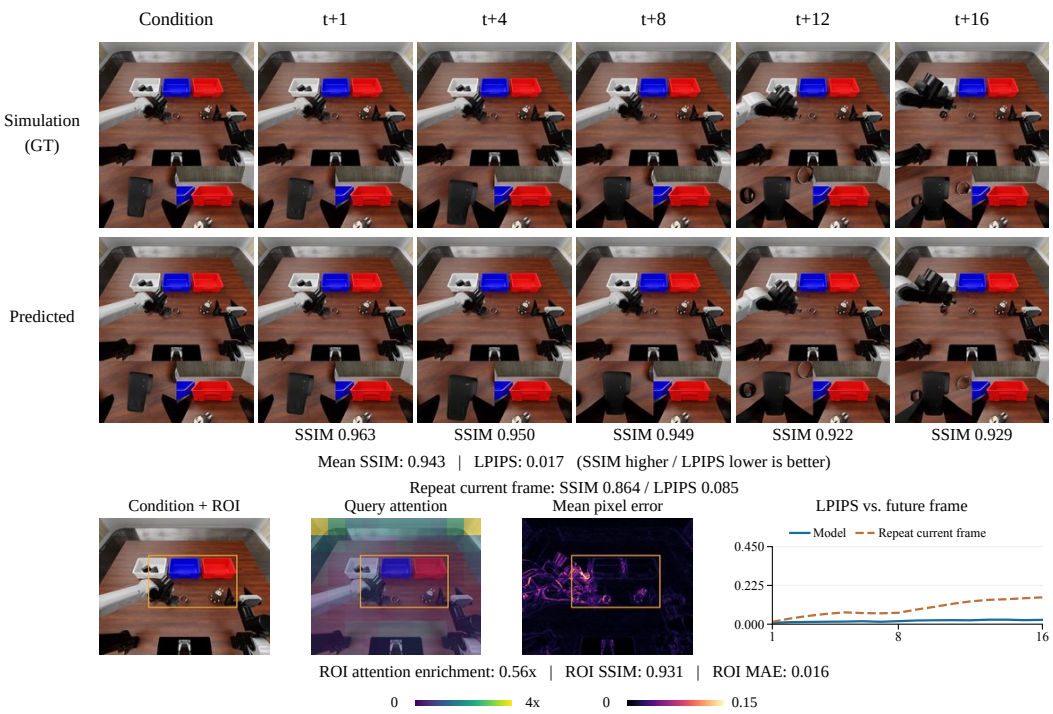


Figure 31: **RoboDojo task 02: Sort objects by category.**

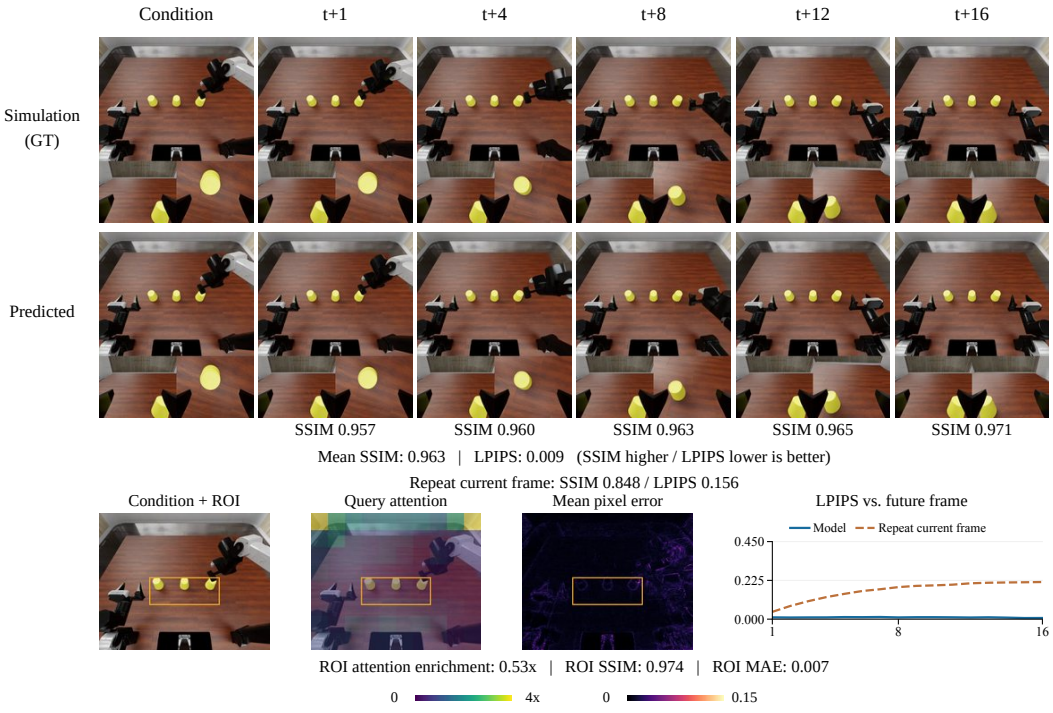


Figure 32: **RoboDojo task 03: Remember covered block colors.**

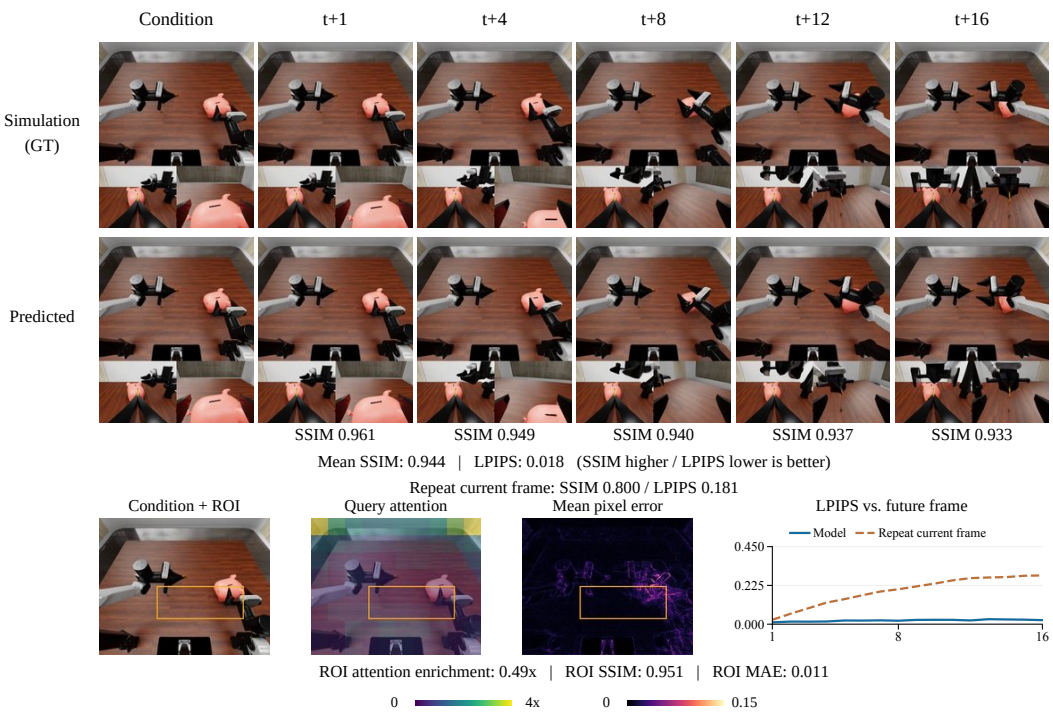


Figure 33: **RoboDojo task 04: Insert a coin.**

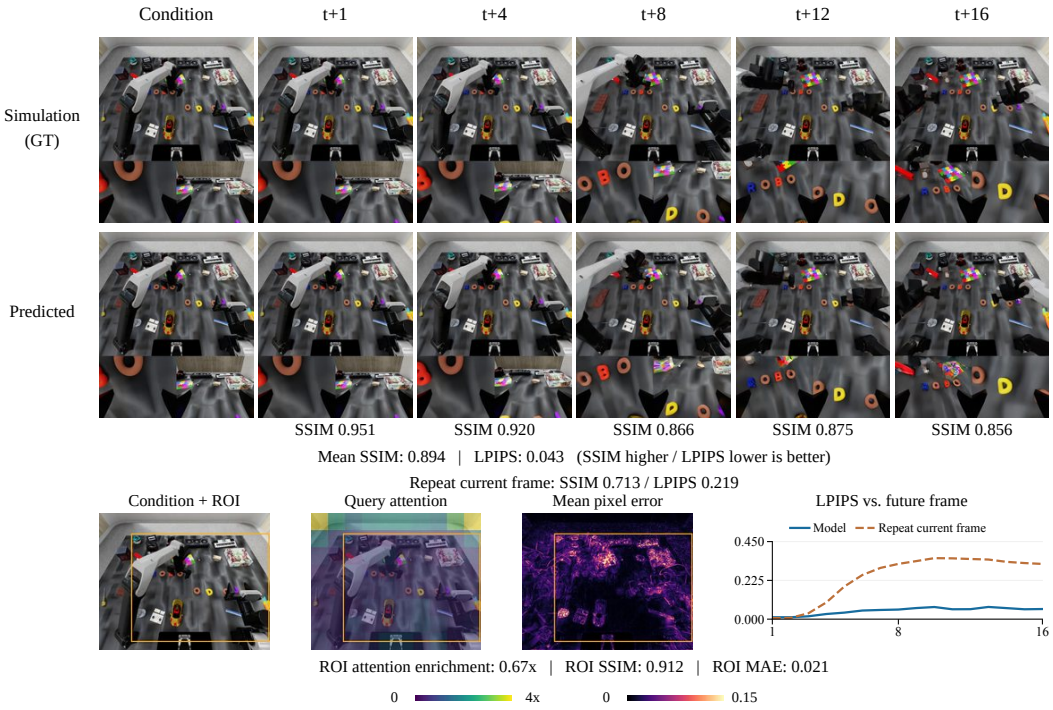


Figure 34: **RoboDojo task 05: Spell RoboDojo.**

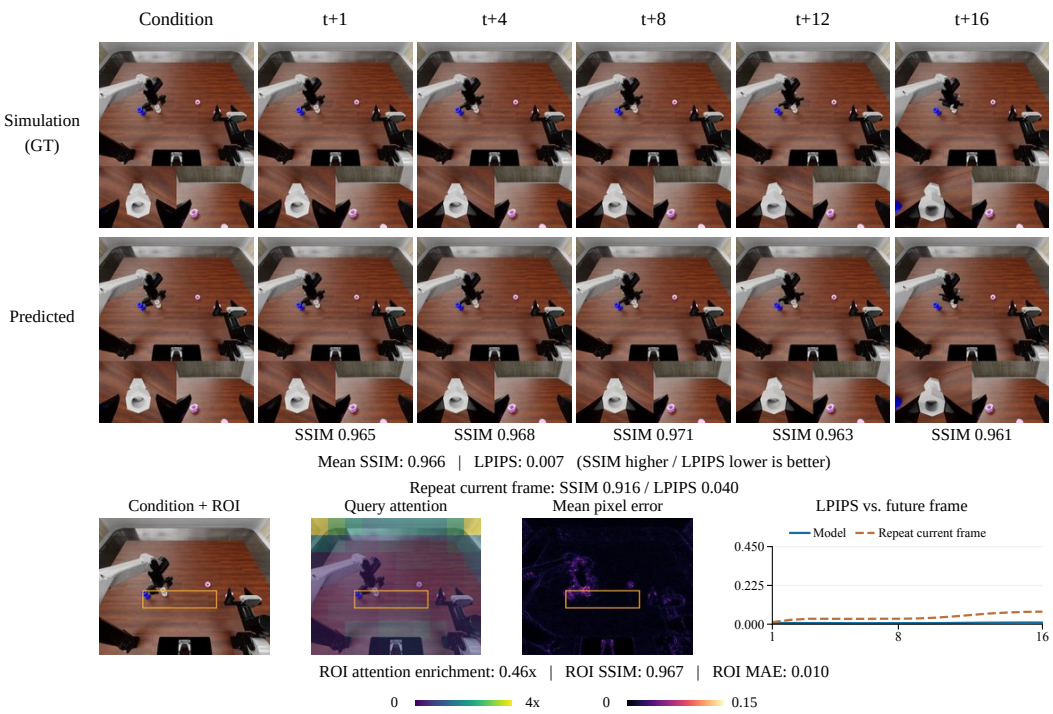


Figure 35: **RoboDojo task 06: Tighten color-matched screws.**

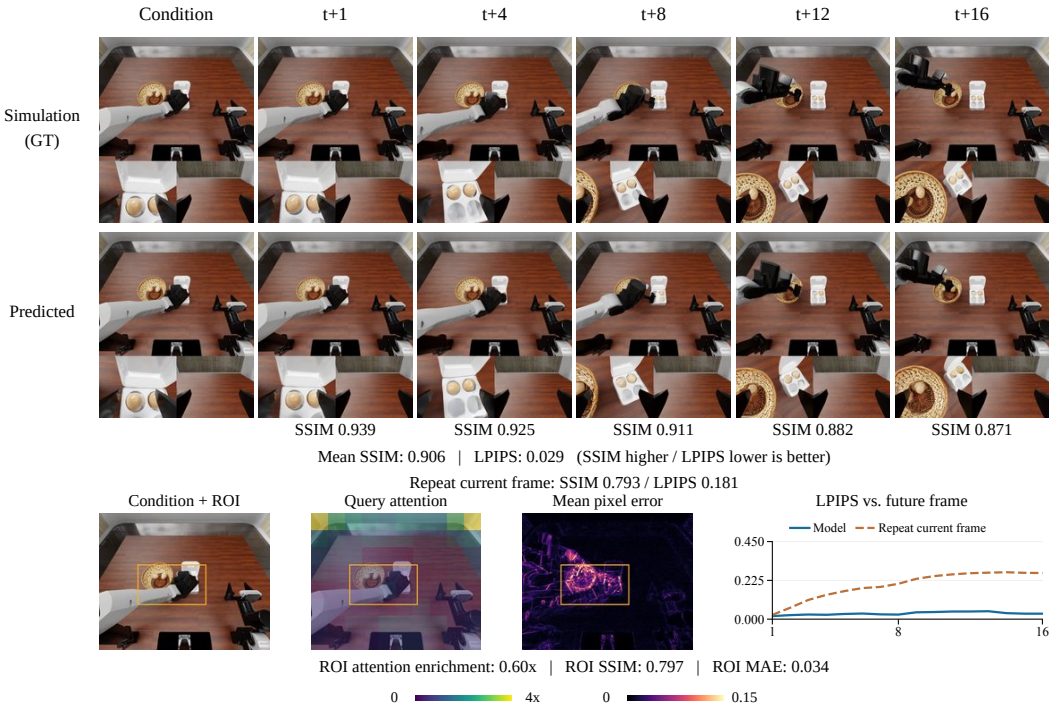


Figure 36: **RoboDojo task 07: Store eggs and close the holder.**

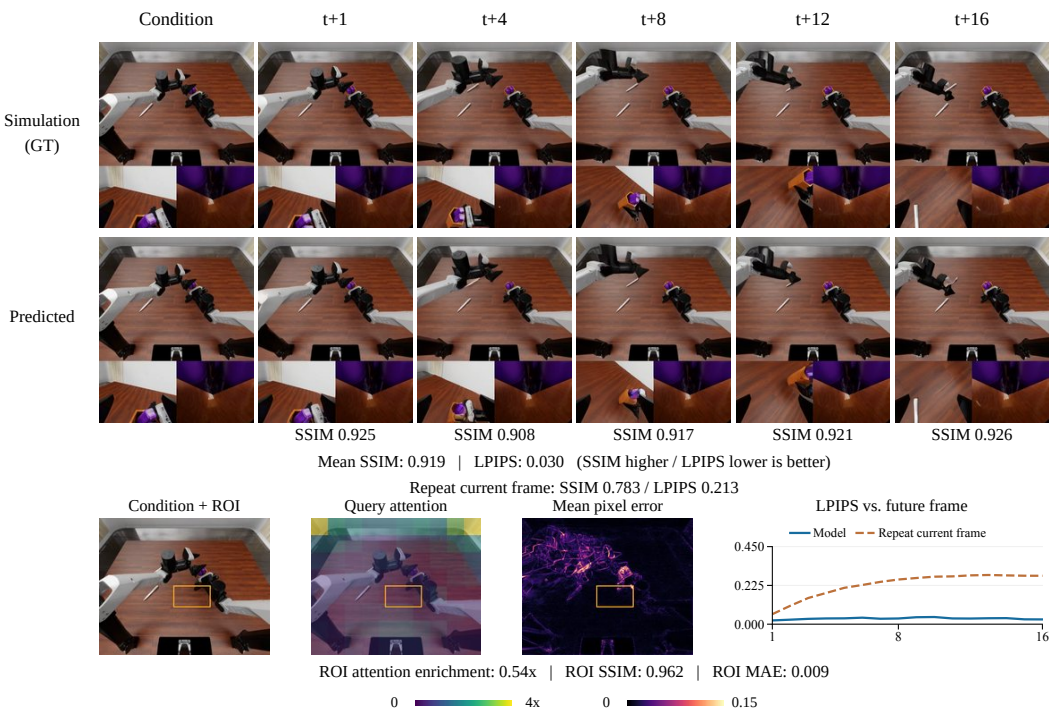


Figure 37: **RoboDojo task 08: Put pens into a holder.**

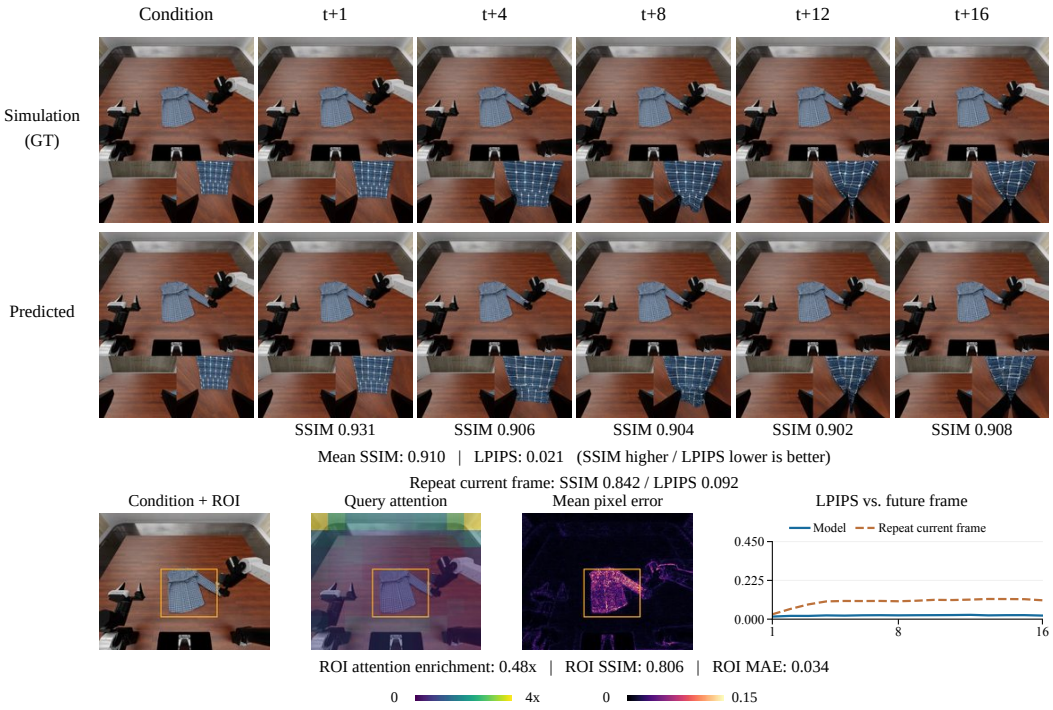


Figure 38: **RoboDojo task 09: Fold clothes.**

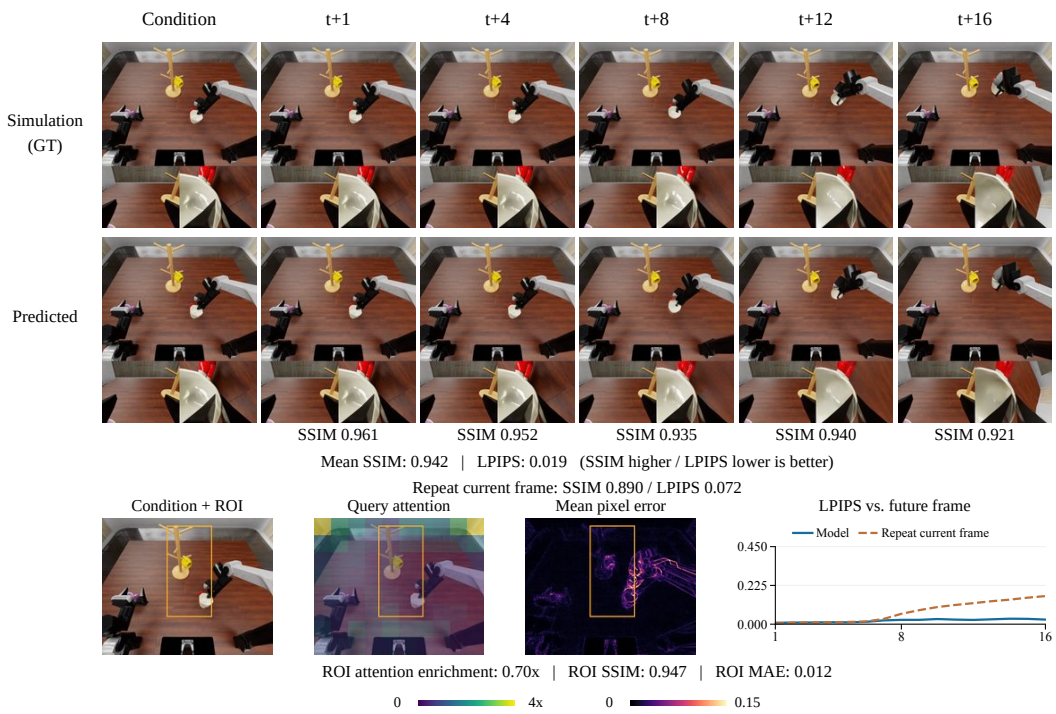


Figure 39: **RoboDojo task 10: Hang mugs.**

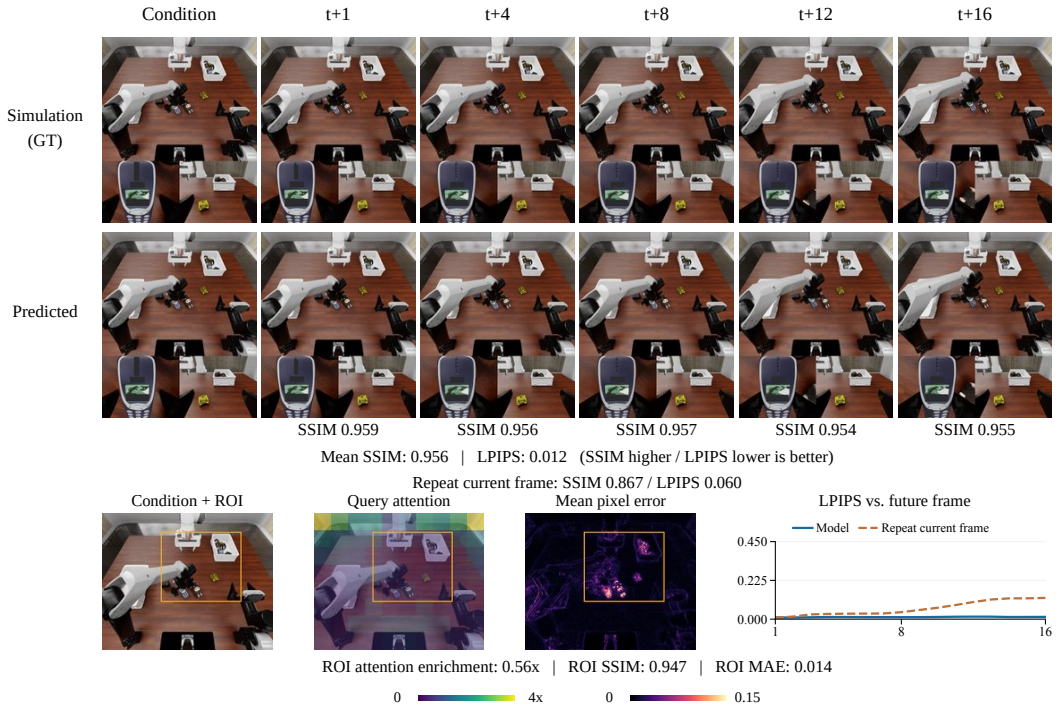


Figure 40: **RoboDojo task 11: Recall object placement order.**

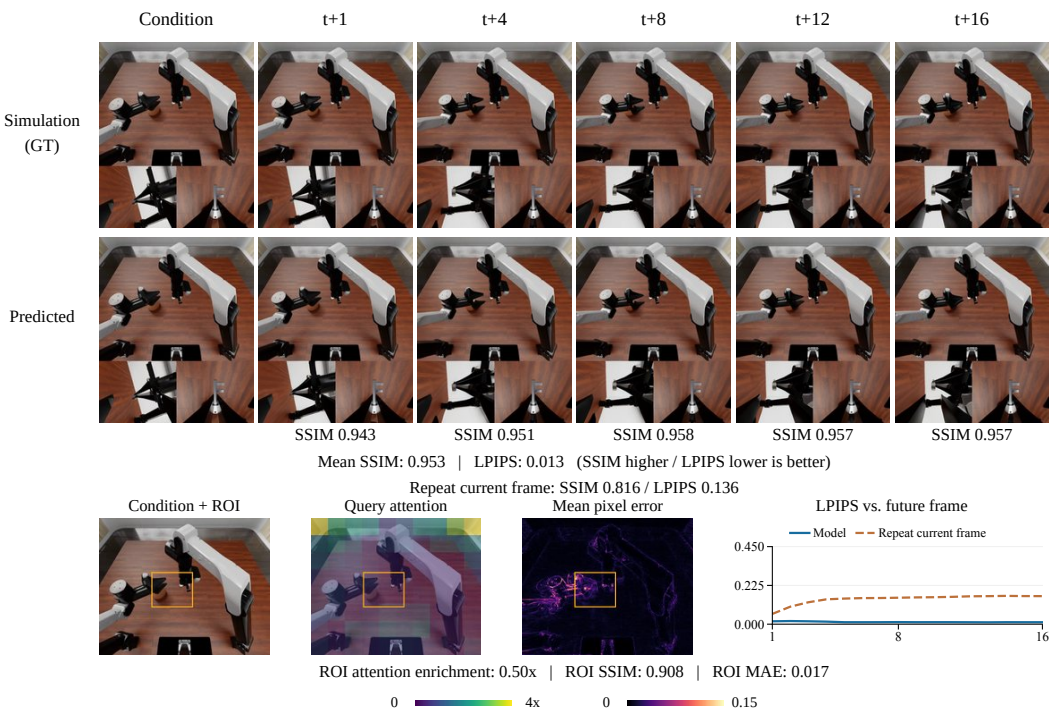


Figure 41: **RoboDojo task 12: Insert a thick card.**

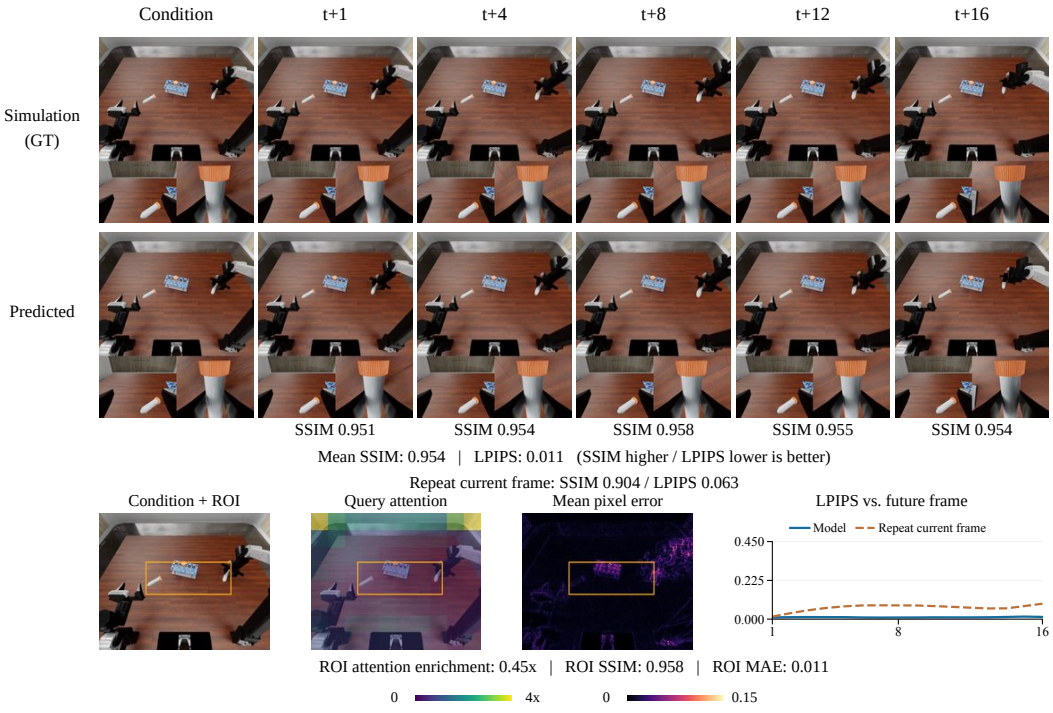


Figure 42: **RoboDojo task 13: Place tubes in a rack.**

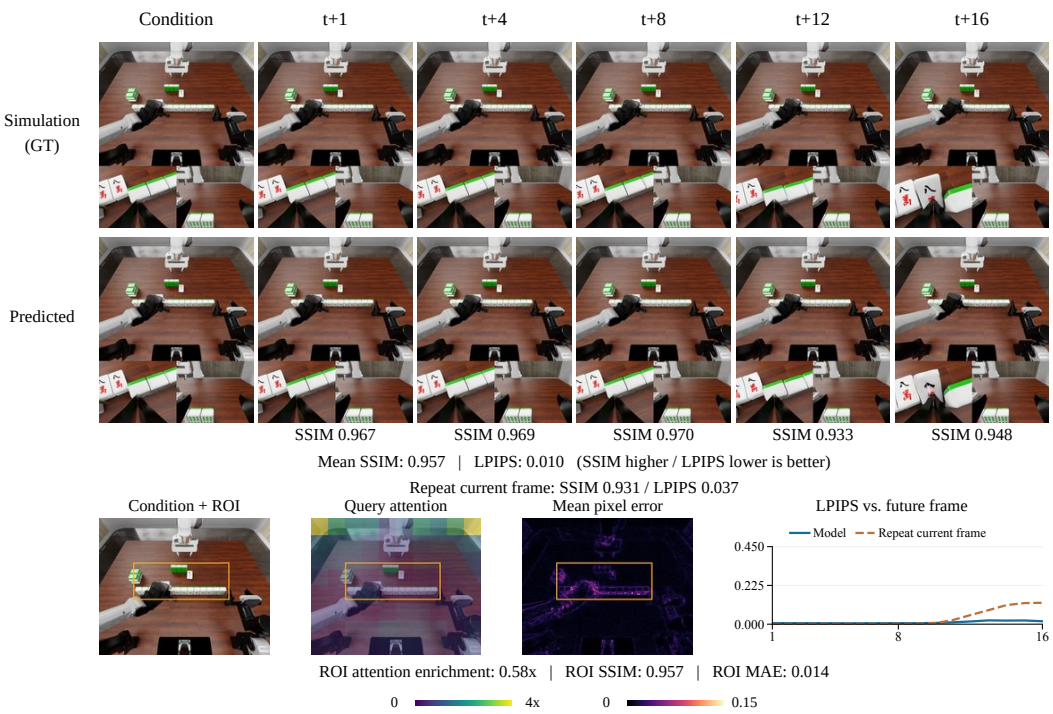


Figure 43: **RoboDojo task 14: Form a mahjong kong.**

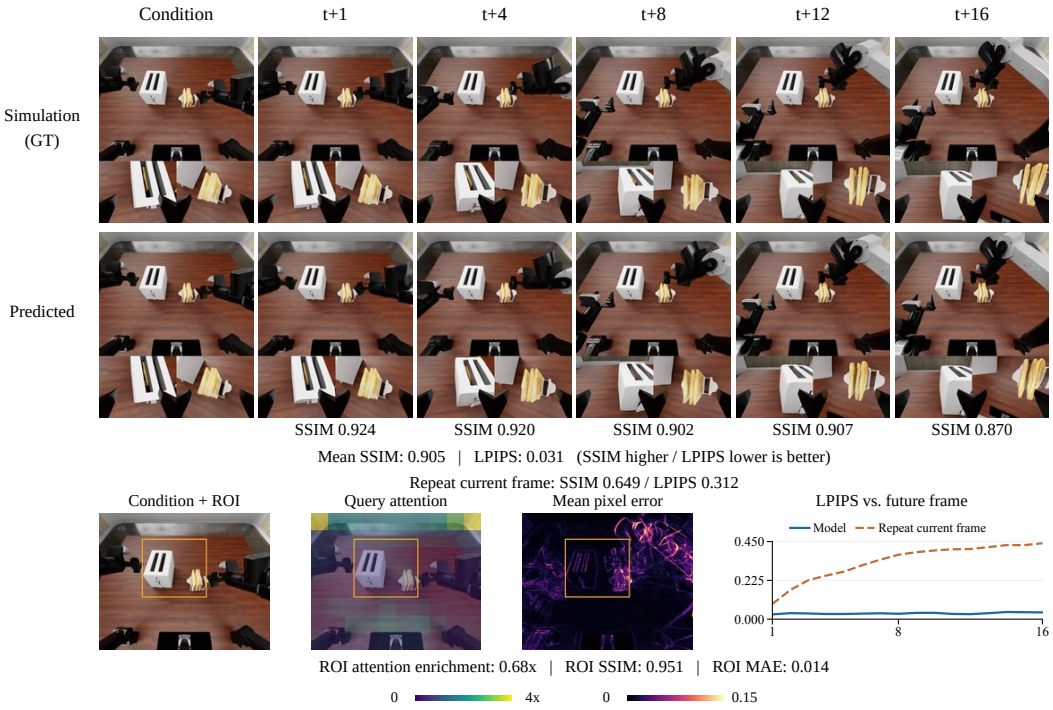


Figure 44: **RoboDojo task 15: Toast bread.**

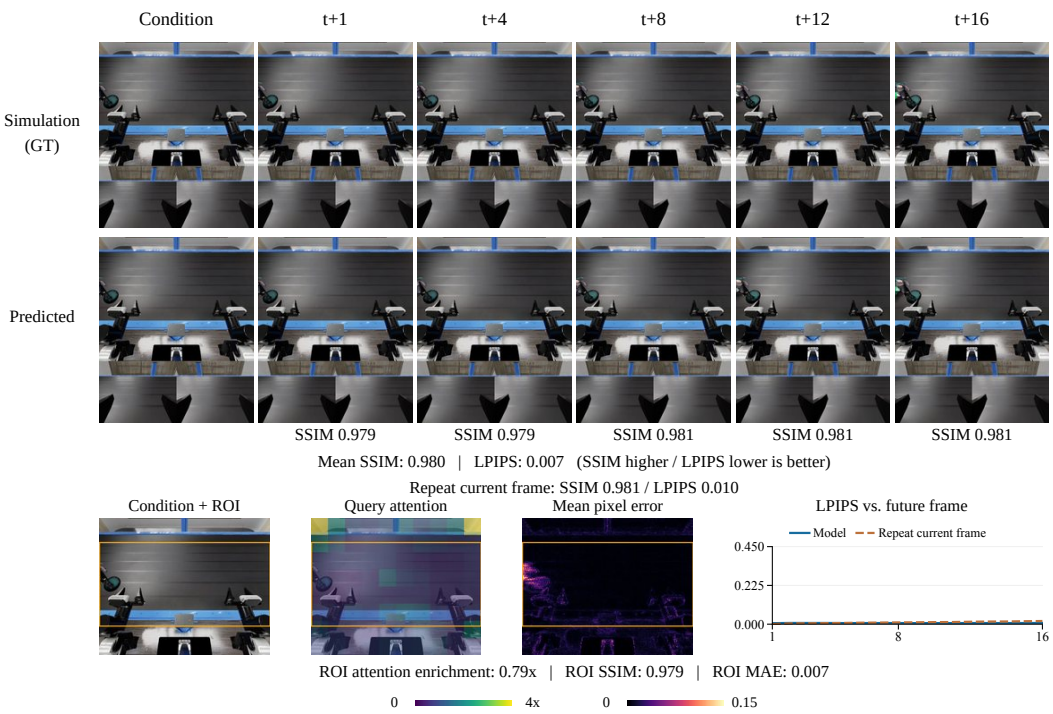


Figure 45: **RoboDojo task 16: Match a remembered conveyor object.**

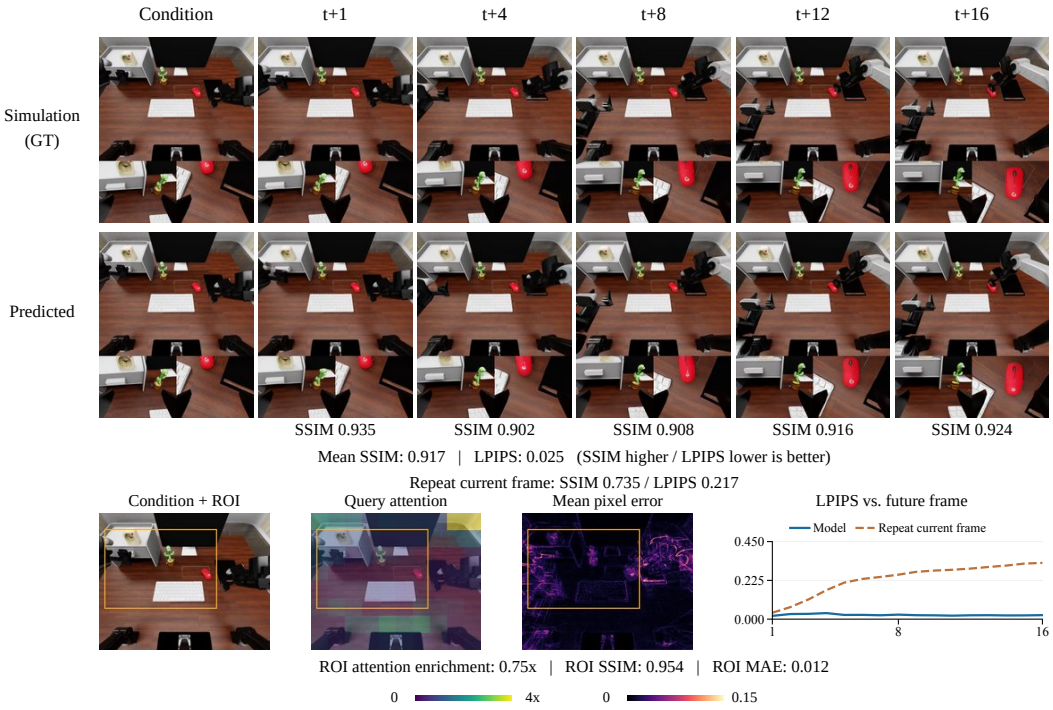


Figure 46: **RoboDojo task 17: Organize a desk.**

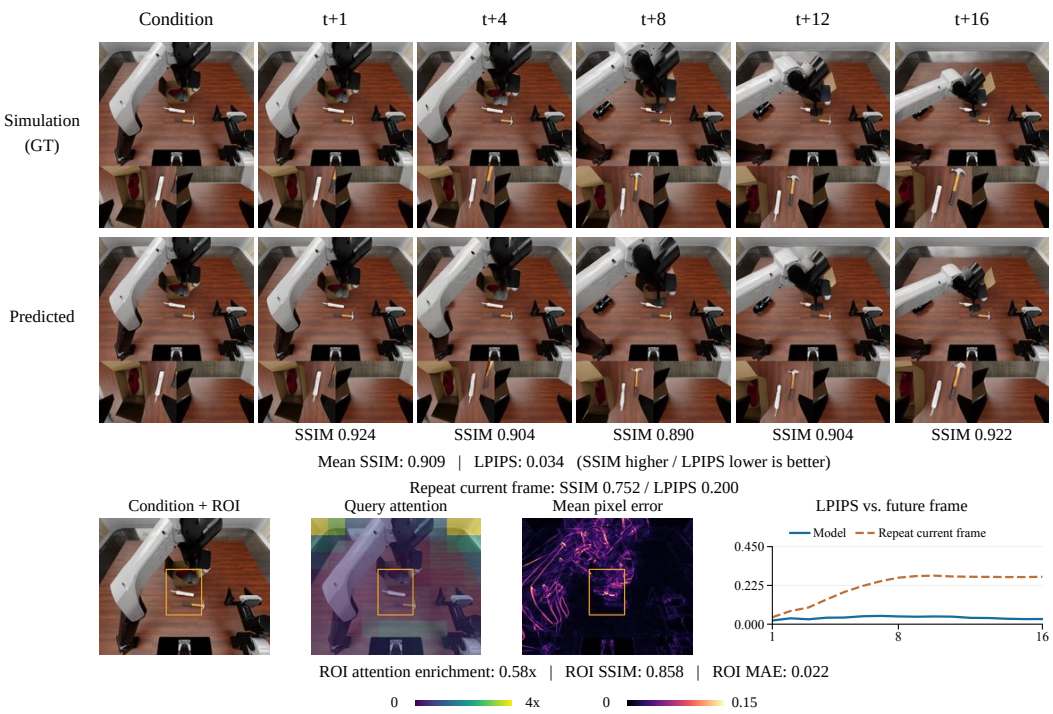


Figure 47: **RoboDojo task 18: Pack objects facing left.**

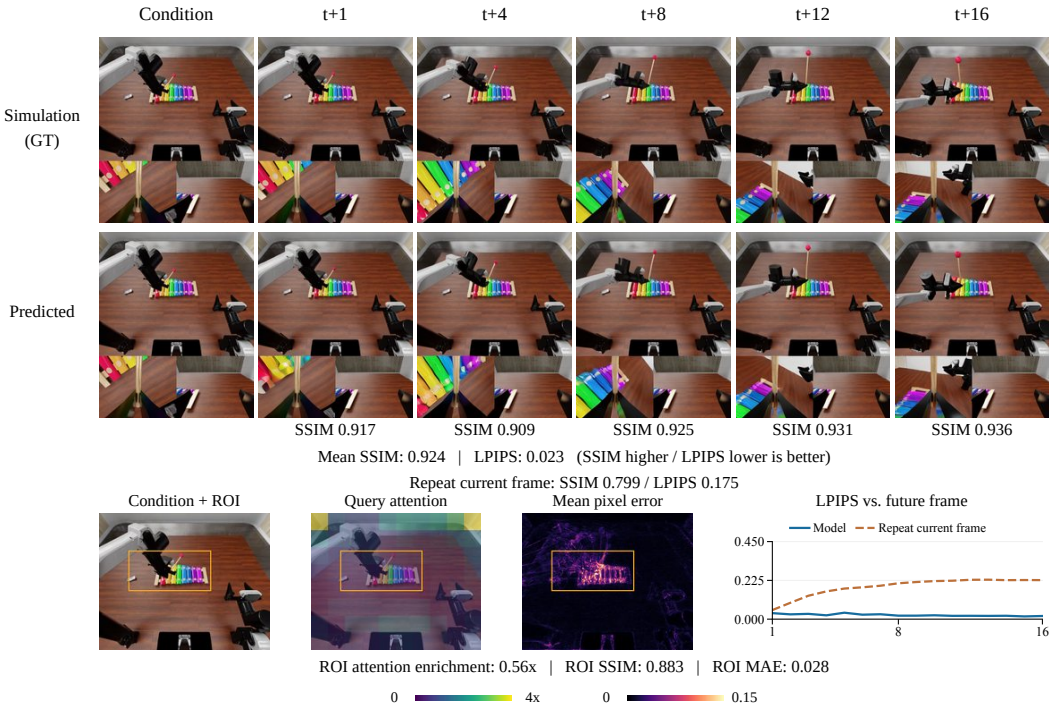


Figure 48: **RoboDojo task 19: Play the xylophone.**

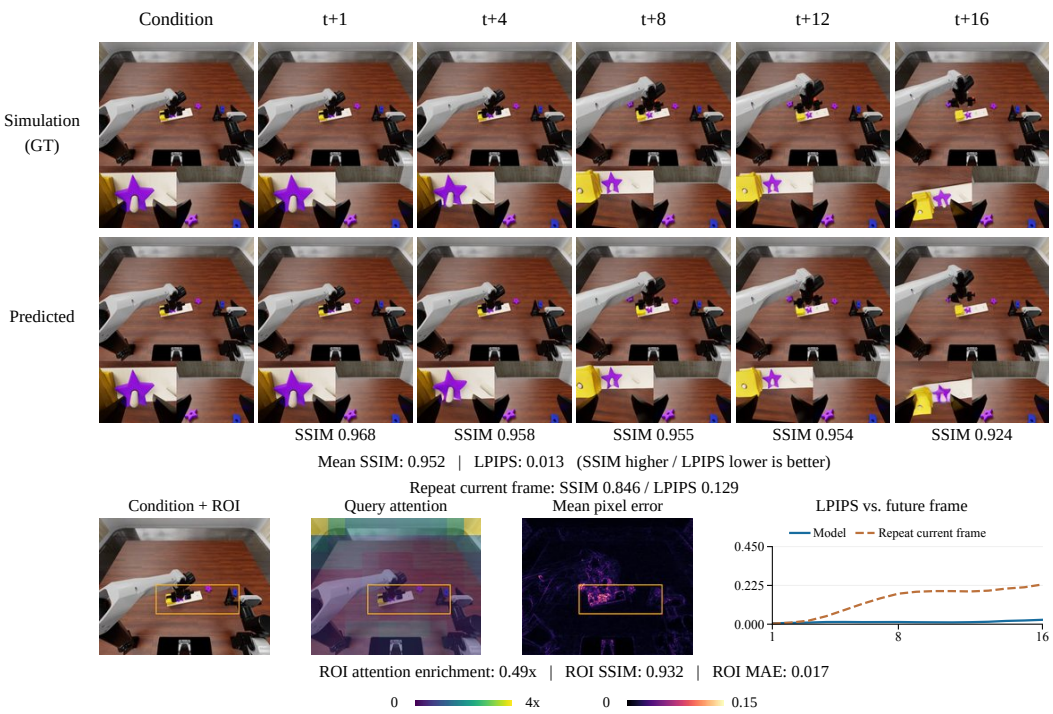


Figure 49: **RoboDojo task 20: Assemble a stacking toy.**

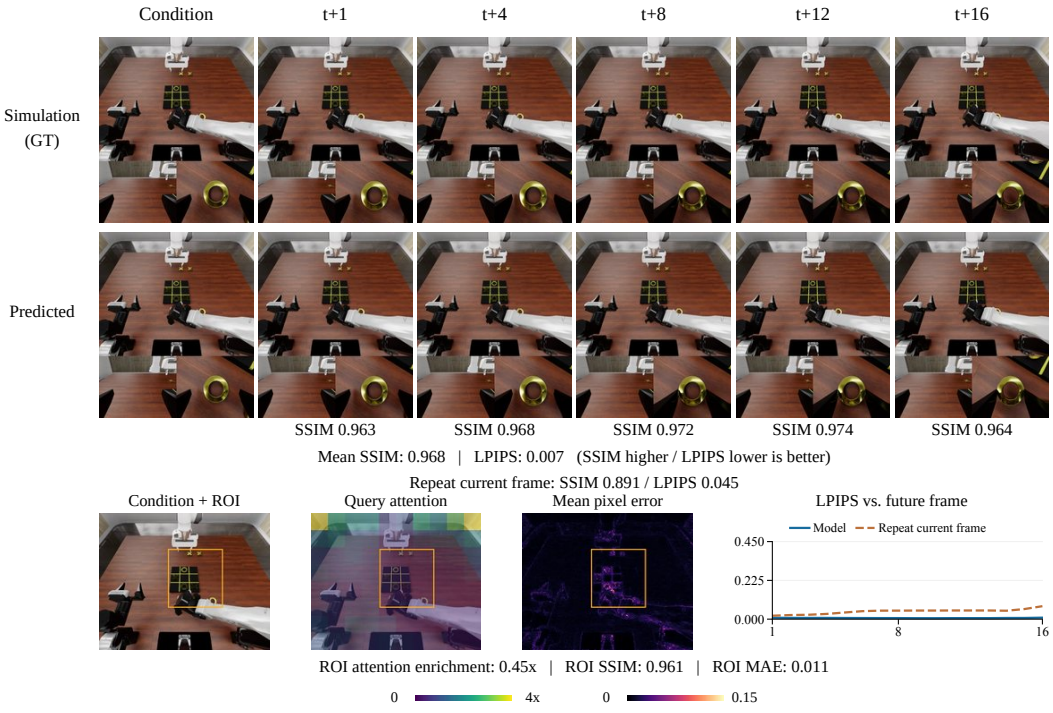


Figure 50: **RoboDojo task 21: Play tic-tac-toe.**

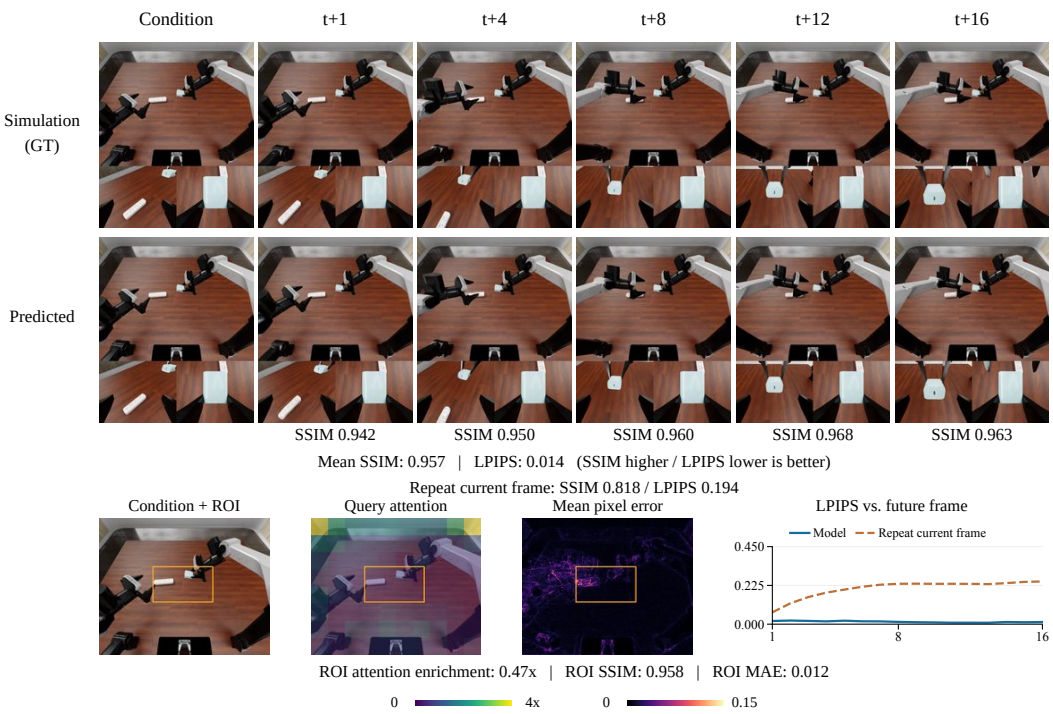


Figure 51: **RoboDojo task 22: Plug in a charger.**

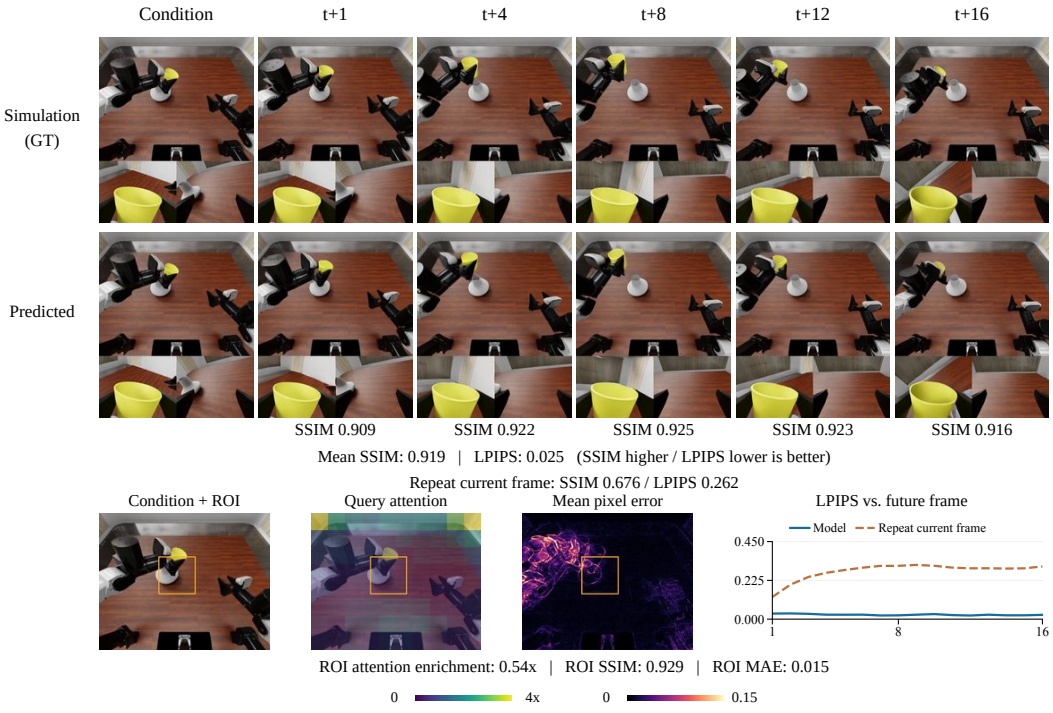


Figure 52: **RoboDojo task 23: Pour balls into a vase.**

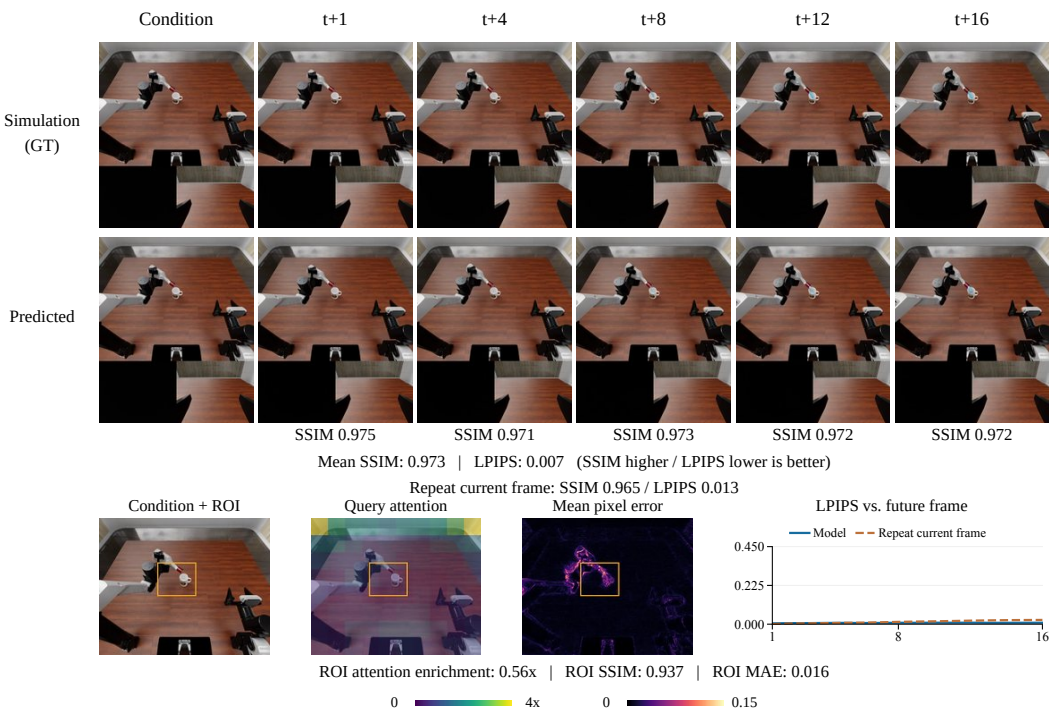


Figure 53: **RoboDojo task 24: Pour liquid into a cup.**

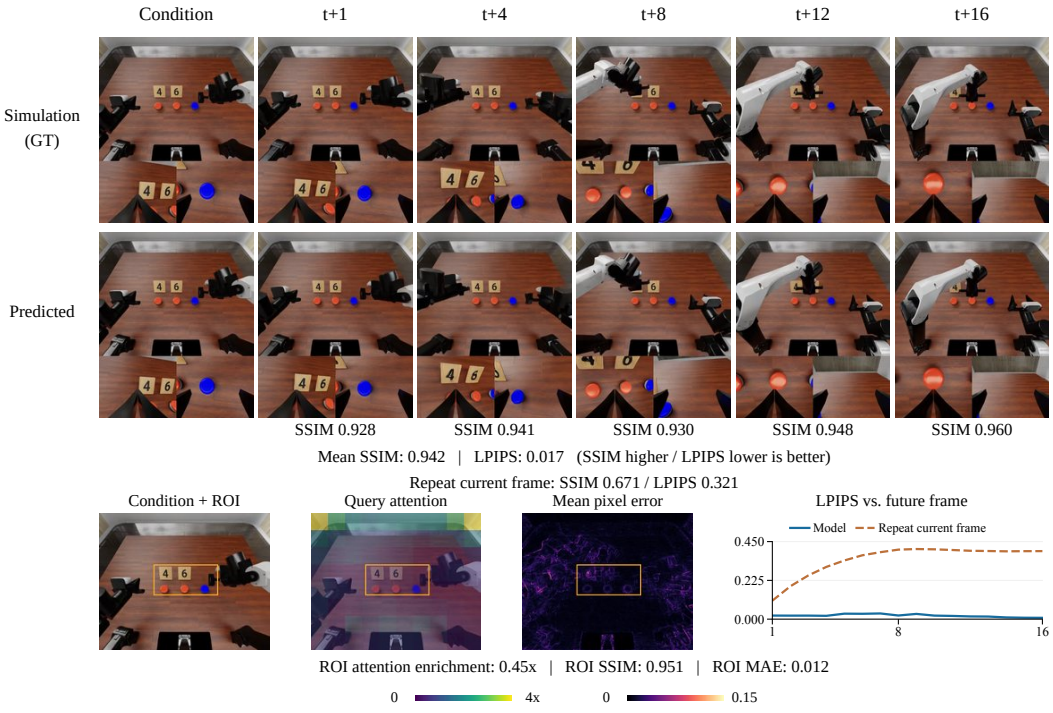


Figure 54: **RoboDojo task 25: Count button presses.**

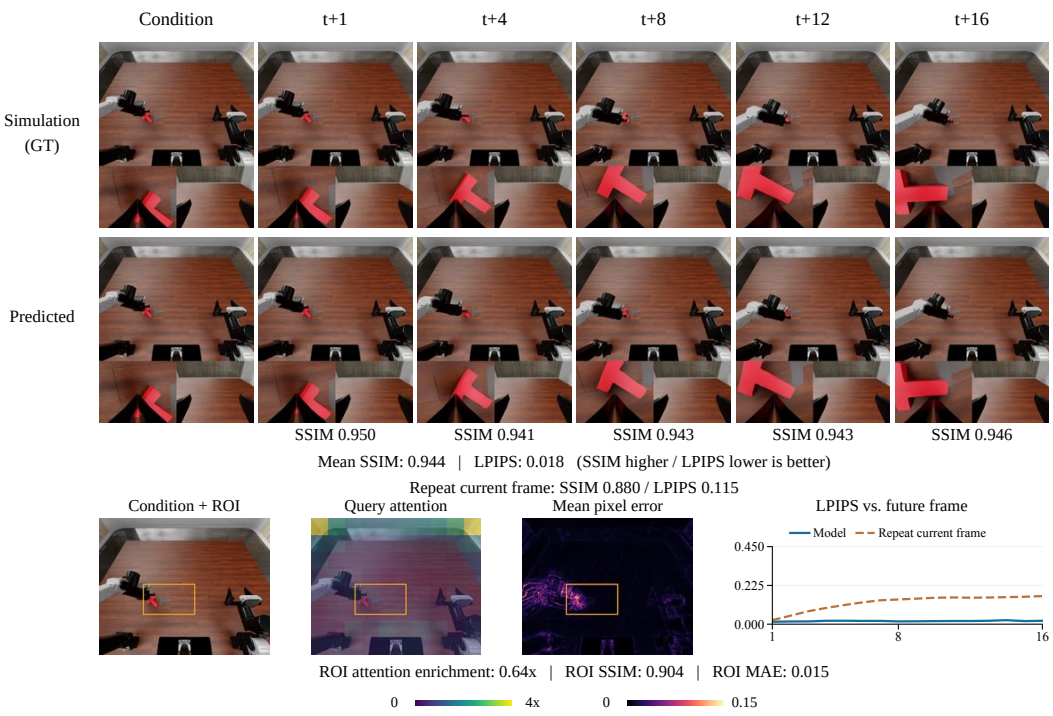


Figure 55: **RoboDojo task 26: Align a T-shaped block.**

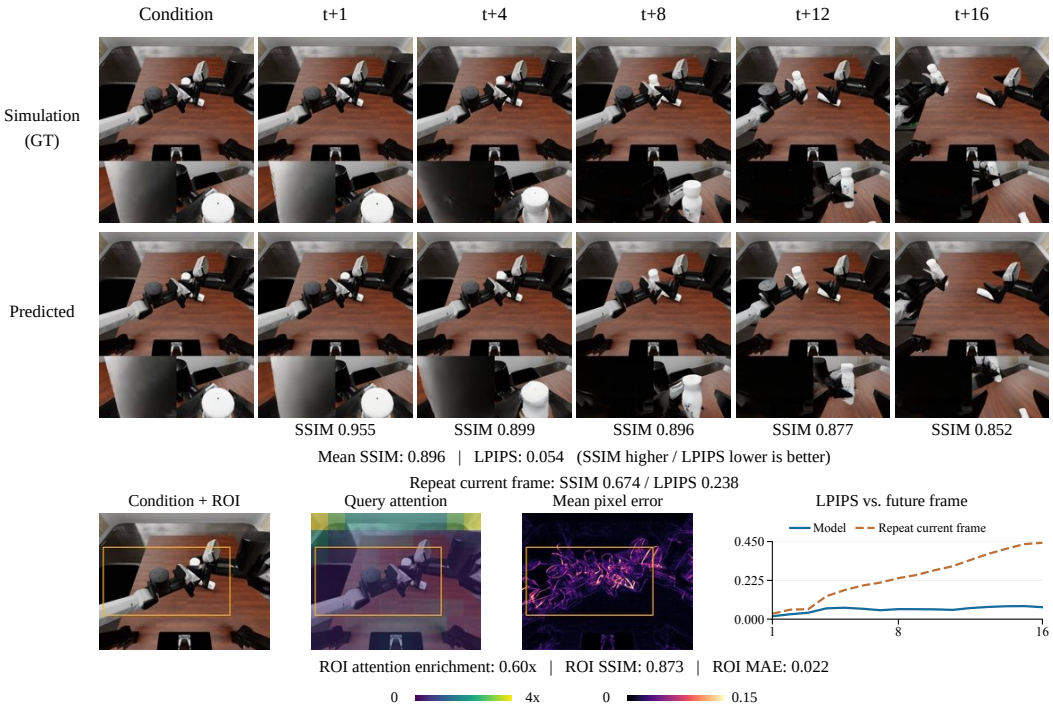


Figure 56: **RoboDojo task 27: Discard bottles.**

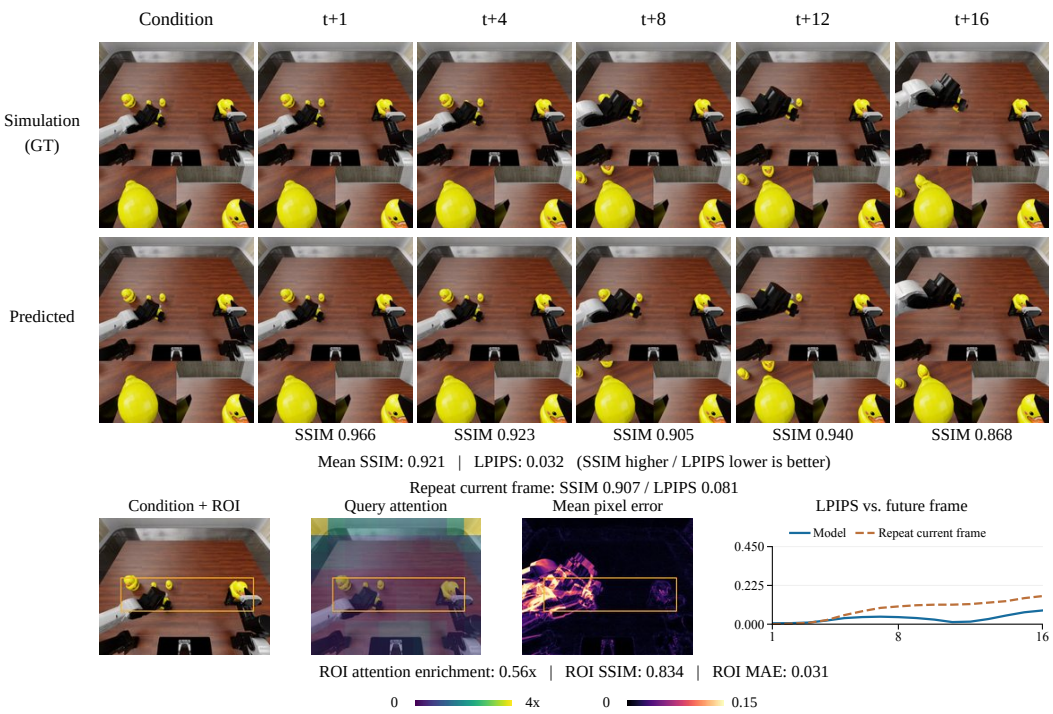


Figure 57: **RoboDojo task 28: Order nesting dolls.**

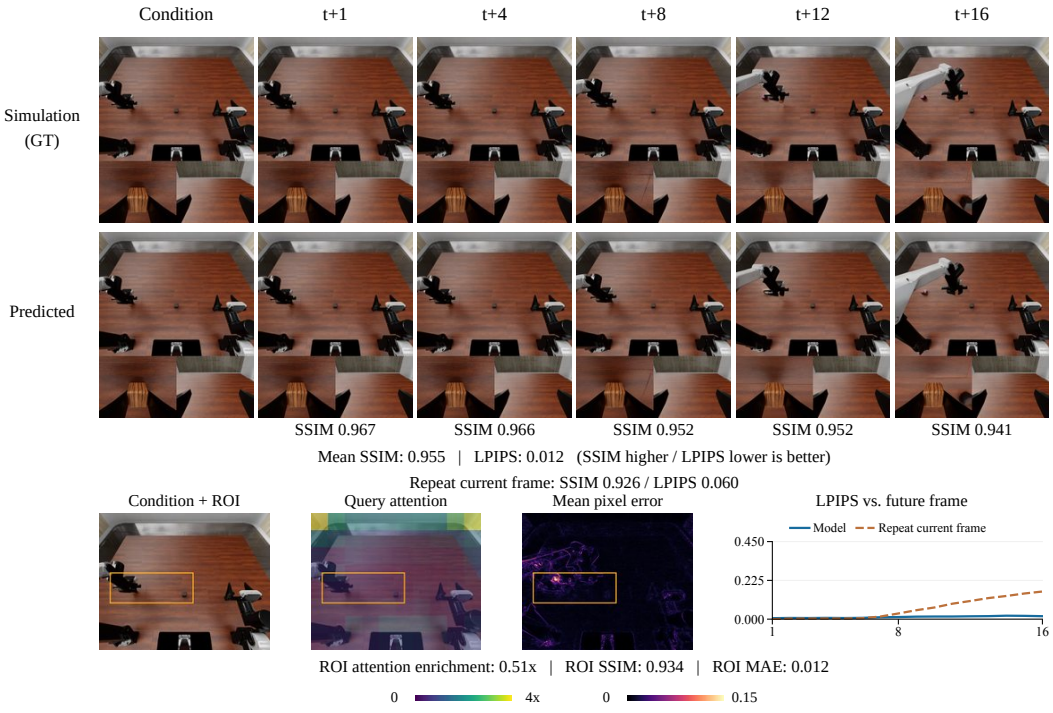


Figure 58: **RoboDojo task 29: Stack textured blocks.**

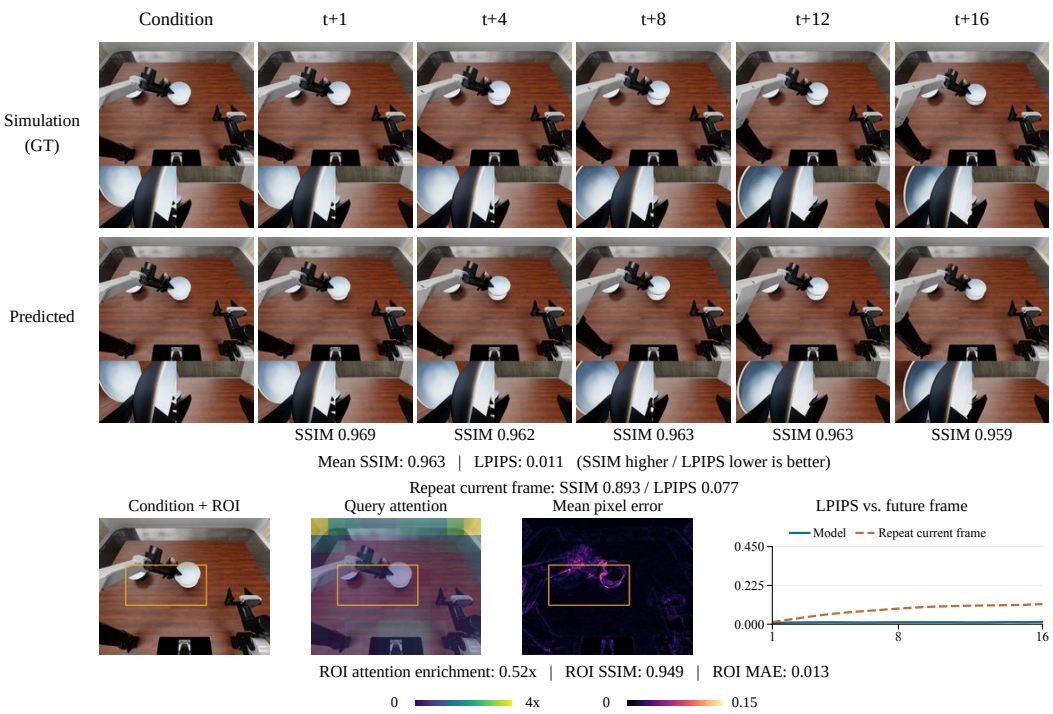


Figure 59: **RoboDojo task 30: Stack bowls.**

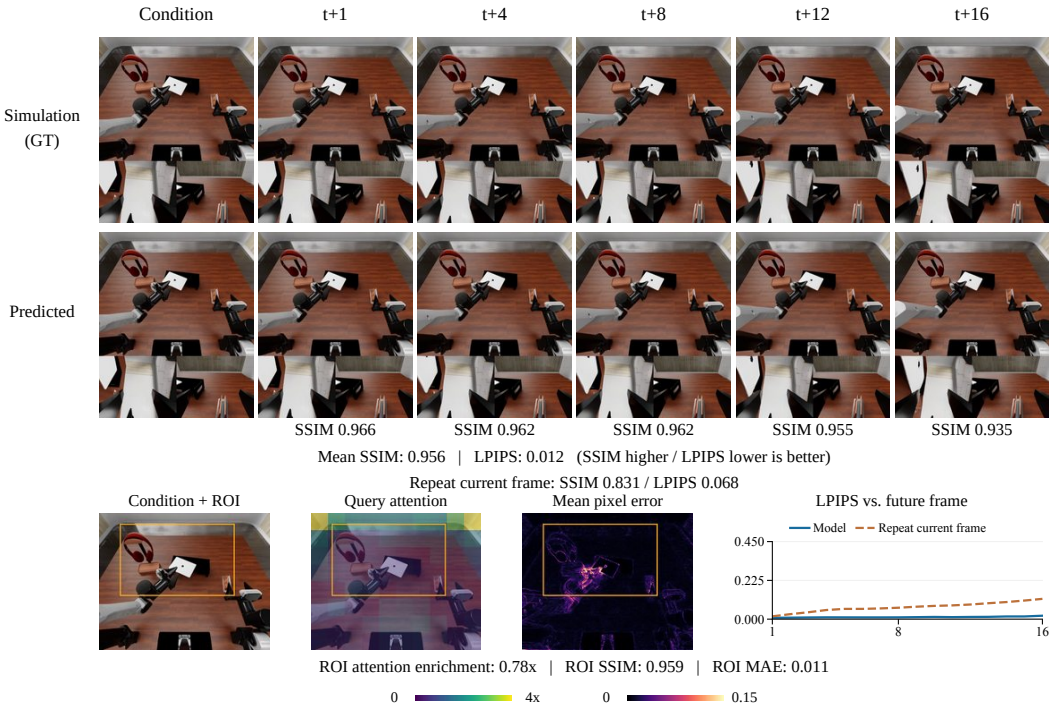


Figure 60: **RoboDojo task 31: Store headphones and a laptop.**

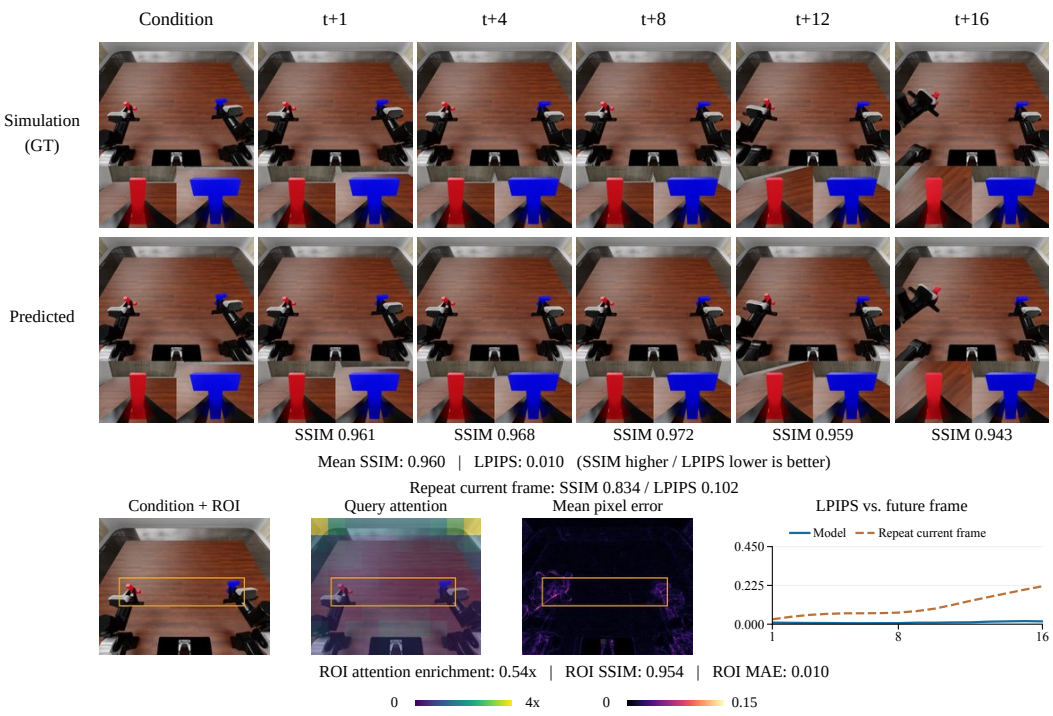


Figure 61: **RoboDojo task 32: Swap T-shaped blocks.**

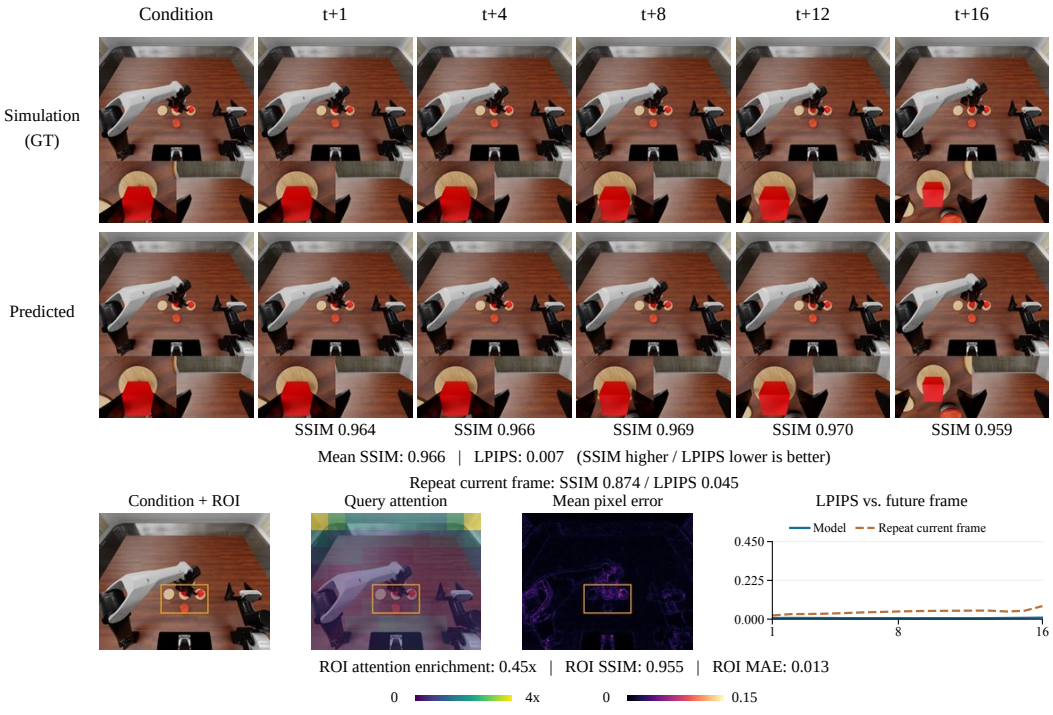


Figure 62: **RoboDojo task 33: Swap blocks via an empty mat.**

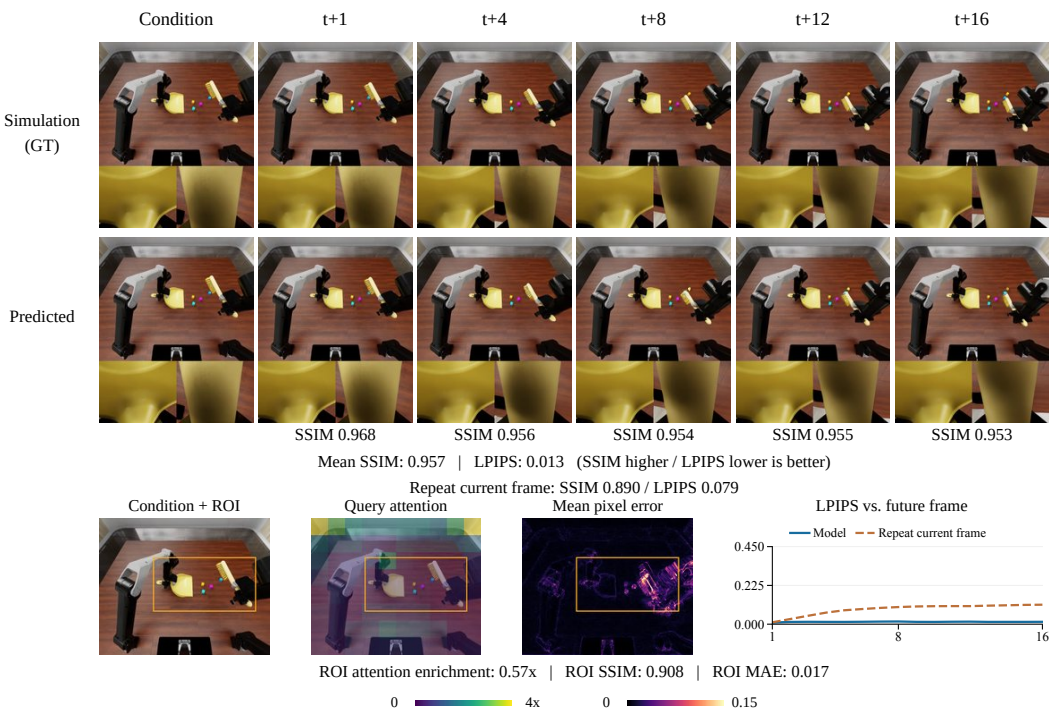


Figure 63: **RoboDojo task 34: Sweep blocks.**