SCIENCE CHINA
Information Sciences

• Supplementary File •

Risk-guided reweighting of successful demonstrations for vision-language-action learning

Jiahao XU†, Jianbo YU†, Peiyuan WANG, Hanzhuo ZHANG, Xuanhao XIANG, Hao CHEN, Jiaxing YU, Chuanye LI, Chenchen FU & Wanyuan WANG*

School of Computer Science and Engineering, Southeast University, Nanjing 211189, China

Appendix A Extended Experimental Setup and Reproducibility

This appendix specifies experimental protocols and implementation details. Baseline–RAISE pairs retain the same architecture, interfaces, demonstrations, initialization, optimization budget, and evaluation distribution.

Appendix A.1 Simulation Tasks and Evaluation Protocol

Tasks and demonstrations. The ten tasks in Table 1 in the main text retain their native scenes, robots, action interfaces, and success conditions. Each main experiment uses a fixed *Expert300* set of 300 successful trajectories, with identical paired membership and no imposed clean-to-retry ratio. Successful trajectories containing retries remain included. For PushCube, the four $\pi_{0.5}$-based configurations share one collection of 300 successful demonstrations collected for this study: $\pi_{0.5}$, RAISE ($\pi_{0.5}$), GTP-FA-$\pi_{0.5}$, and RAISE (GTP-FA-$\pi_{0.5}$).

Success criteria. Let $s_t \in \{0,1\}$ be the task-success flag and $c_t \in \{0,1\}$ the completion predicate for stability, with $c_t \leqslant s_t$. They coincide when no additional completion conditions apply. For a rollout of length T,

$$S_{\text{once}} = \max_{1\leqslant t\leqslant T} s_t, \qquad S_{\text{end}} = s_T, \qquad S_{\text{stable@3}} = \max_{3\leqslant t\leqslant T} (c_{t-2}c_{t-1}c_t),$$

where $S_{\text{stable@3}} = 0$ for $T < 3$. The completion window comprises three consecutive evaluation states anywhere in a rollout. Stable Success@3 implies success-once. Its relation to success-at-end depends on subsequent execution: a completed window can precede a later failure, and final success can occur before a three-state window is complete. The physical duration of this window depends on the evaluation sampling rate. For StackCube, c_t requires stacking and release, linear speed $\leqslant 0.01$ m/s, and angular speed $\leqslant 0.5$ rad/s; other tasks retain their own predicates.

Evaluation and aggregation. Models trained for this study and reported in Tables 1 in the main text and 2 in the main text are evaluated on 200 fixed environment seeds for at most 250 steps; evaluation continues after transient success. Training seeds are $\{0,1,2\}$ for BC/DP and $\{42,43,44\}$ for the main VLA results. Within each task, method, and seed, baseline and RAISE start identically and train independently. Paired variants and training seeds share task-specific evaluation lists; BC/DP also share theirs. Main VLA evaluations use `400005010--400005209`, inclusive. For success counts k_s, $\bar{p} = \frac{1}{3}\sum_s 100k_s/200 = 100\sum_s k_s/600$. Thus, the 600 rollouts evaluate three independently trained models on the same 200 initializations. Means and gains are computed before rounding. Initial scenes are task-specific even when numeric seeds coincide; BC/DP and VLA use separately specified evaluation lists. Final-test initial conditions are reserved for evaluation and are excluded from parameter tuning. The StackCube risk-source and demonstration-composition studies in Appendix E.3 and Appendix E.4 use training seed 42.

Simulation backend and control. Simulations use the PhysX CUDA backend. VLA runs on A100/RTX 5090 use ManiSkill3 3.0.0b18 and SAPIEN 3.0.0b1; BC/DP runs use ManiSkill3 3.0.0b21 and SAPIEN 3.0.2. Native control and physics frequencies are 20 and 100 Hz, respectively. Policies retain task-specific joint-position or end-effector-pose increment interfaces, fixed within each baseline–RAISE pair. BC updates its action every native control step, DP replans after eight actions, and $\pi_{0.5}$ after ten. The PegInsertionSide VLA wrapper repeats each action twice. Success rates include all valid task rollouts. Infrastructure exceptions are recorded separately from task outcomes.

* Corresponding author (email: 230248296@seu.edu.cn)
† Jiahao Xu and Jianbo Yu contributed equally to this work.

Appendix A.2 Data Roles and Isolation

Table A1 describes the main simulation RAISE runs: policy demonstrations, progress adaptation, reference, calibration, development, and final evaluation are trajectory-separated. The auxiliary-action-training controls additionally use the auxiliary successes as policy targets. Real-robot runs reuse their action demonstrations for the auxiliary roles specified in Appendix A.6. Development and final evaluation trajectories are excluded from training. Natural failures are valid rollouts without Stable Success@3. Thresholds, offline base weights, and evaluation checkpoints are fixed within each comparison; V3 selection is described in Appendix E.3.

Auxiliary experience. The 300-demonstration budget counts policy action supervision only. Within each task, policy families and training seeds share frozen auxiliary trajectory pools; model-dependent quantities are recomputed when required by an ablation. For StackCube, progress adaptation uses 1000 clean-success trajectories and 1000 retry events from 729 successful trajectories. The success reference bank and calibration set contain 1000 trajectories each. Of 489 natural failures, 352 yield confirmed risk anchors (Appendix E.1). In the main RAISE runs, these auxiliary trajectories support progress adaptation and offline weight construction; Expert300 supplies policy action targets.

Table A1 Data roles and direct gradient use in the main simulation RAISE setting. Auxiliary failure evidence determines weights but supplies no policy action targets.

Data role	Source	Gradient use	Purpose
Expert300	Stable successful demonstrations	Policy	Policy action supervision
Ordinary success	Clean stable-success trajectories	Progress model	Stage and absolute-progress supervision
Retry success	Stable successes containing local retries	Progress model	Local degradation–recovery supervision
Retry-risk anchors	Pre-error states from successful retries	None	Retry-derived risk estimation
Natural failures	Failed rollouts from a frozen base policy	None	Failure localization and risk estimation
Success Bank	Independent successful trajectories	None	Local successful-state reference
Calibration set	Independent successful trajectories	None	Evidence-threshold calibration
Development sets	Disjoint clean/retry/controlled rollouts	None	Method development and ablation
Final evaluation	Frozen seeds or physical trials	None	Final performance reporting

Appendix A.3 Retry-Sensitive Progress Model

Architecture and inputs. Table A2 specifies simulation models. Language-backbone LoRA and two heads are trained; ROBOMETER-4B's pretrained parameters, visual encoder, and pretrained prediction heads remain frozen. Stage-slot semantics and active labels are task-specific; the within-stage head is shared. Main inputs are native render-camera RGB at 512×512/H8. Initial padding repeats the first available observation. Physical-state labels and labeling potentials are used exclusively for offline supervision.

Eligibility and supervision boundaries. Only stable-success trajectories adapt the model. Task-specific physical predicates automatically generate simulation labels from recorded privileged states. Manual annotation validation was limited to 300 StackCube trajectories; the other nine simulation tasks used automatically generated labels without trajectory-level manual annotation (Appendix D). Equation (4) in the main text normalizes each valid contiguous nonterminal segment independently; re-entry starts a new segment. Singletons and retry neighborhoods have no absolute-progress targets. The first stable window anchors unit progress; later frames are excluded from model training, not from fixed policy-demonstration rows. Stage, progress, and terminal masks remain distinct. For StackCube, terminal success without a stable window supplies no training endpoint. The recorded observations and diagnostics are retained.

Retry windows and neighboring events. Annotations determine W_r^{pre}, W_r^{near}, and W_r^{post}; predicted minima do not select them. Windows are clipped at trajectory endpoints, the effective supervision boundary, and neighboring attempts. Under the StackCube labeler, a one-step predicate interruption is transient; an interruption of at least two policy steps can initiate a reliable retry. A connected transient can supply context but cannot independently establish an event. Empty windows or invalid ordering exclude an attempt from $\mathcal{L}_{\mathrm{retry}}$ without relocating its keypoint; the reliable degradation interval remains masked from absolute-progress targets. Clipping resolves conflicts involving near-error windows. Overlap between W_r^{post} and W_{r+1}^{pre} is permitted because they provide positive contexts for different setbacks. Means use actual members without supervision padding. A one-member retry window is valid. A singleton segment is excluded from absolute-progress supervision because temporal normalization requires a nonzero interval. Figure A1 illustrates this distinction. These event-boundary rules are specific to StackCube; other simulation tasks use event labels generated automatically by their task-specific predicates. The labeler targets degradation–recovery and has no dedicated stall-only detector.

Training streams. The clean-success stream supplies stage, absolute-progress, and terminal supervision. An independent auxiliary stream samples eight retry events from eight distinct successful trajectories and weights events equally. Retry trajectories supply only the auxiliary retry loss. Both streams optimize Eq. (7) in the main text. After the prescribed adaptation budget, all parameters are frozen for localization and reweighting; failed trajectories are then analyzed with this fixed model. Appendix B evaluates the resulting model's global progress accuracy and local retry sensitivity. Supervision boundaries and retry-window handling are specified in the preceding paragraphs.

Table A2 Default simulation progress-model configuration. The single-task PushCube model uses LoRA rank 32; the joint ten-task model uses rank 64 and 10,000 updates with task-balanced sampling. Loss coefficients follow the stage/progress/retry/terminal order in Eq. (7) in the main text.

Component	Configuration
Backbone	ROBOMETER-4B
Visual input	render RGB, 512×512, H8
Trainable backbone	language LoRA
LoRA rank / alpha / dropout	64 / 128 / 0.05
LoRA projections	q/k/v/o/gate/up/down
Stage head	$2560 \rightarrow 4$
Within-stage head	$2560 \rightarrow 10$
Main-stream batch	16
Retry auxiliary batch	8 events from 8 trajectories
Optimizer	AdamW
Learning rate	2×10^{-5}
Betas / ϵ	$(0.9, 0.999)$ / 10^{-8}
Weight decay	0.01 for matrix parameters
Gradient clipping	global norm 10
Warmup / scheduler	100 updates / cosine
Training budget	1000 optimizer updates
Loss coefficients	1/1/2/1
Precision	BF16 mixed precision
Model selection	fixed final update 1000

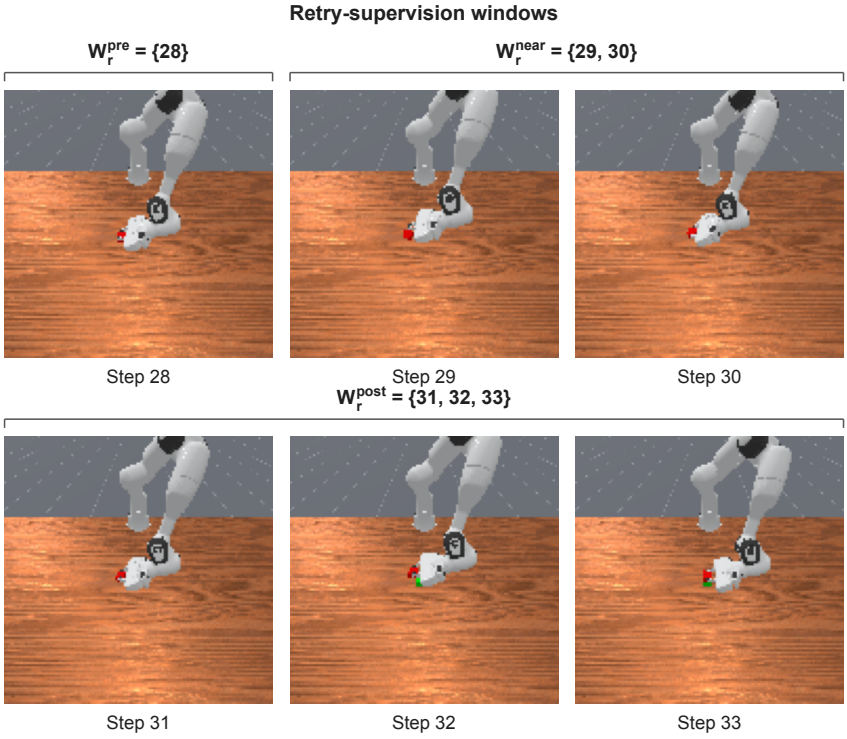

Figure A1 A successful StackCube retry: $W_r^{\text{pre}} = \{28\}$, $W_r^{\text{near}} = \{29, 30\}$, and $W_r^{\text{post}} = \{31, 32, 33\}$. A preceding retry clips the pre-error window from $\{26, 27, 28\}$ to $\{28\}$. Policy-step indices identify the members of each supervision window; window sizes follow the event boundaries.

Appendix A.4 Base Policies and Learner-Specific Training

Table A3 specifies native objectives and training settings. Paired runs share initialization, data, normalization, seed, and budget. BC uses Adam; DP uses AdamW, 500 warmup updates, cosine decay, and EMA with power 3/4. BC/DP normalization statistics are computed from Expert300 and held fixed. The $\pi_{0.5}$ pair starts independently from `pi05_base`, predicts ten-action chunks, and uses fixed final checkpoints. GTP-FA's grasp front-end remains fixed. PushCube BC and BC+RAISE share the execution configuration in Section 4.7 in the main text, including the demonstration-neighbor bank and velocity feedback.

Table A3 Learner-specific configurations, fixed within each baseline–RAISE pair. OpenVLA is used in Table 2 in the main text, not Table 1 in the main text.

Policy	Training data	Optimization	Policy-specific configuration
BC	300/task	Seeds $\{0, 1, 2\}$; batch 1024; LR 3×10^{-4}; 10k iterations	State normalization enabled
Diffusion Policy	300/task	Seeds $\{0, 1, 2\}$; batch 1024; LR 1×10^{-4}; 100k iterations	$T_{\text{obs}} = 2$, $T_{\text{act}} = 8$, $T_{\text{pred}} = 16$; diffusion embedding 64; U-Net $[64, 128, 256]$; 8 groups
$\pi_{0.5}$	300/task	Seeds $\{42, 43, 44\}$; global batch 32; 30k optimizer updates	10-action chunks; fixed final checkpoint
OpenVLA-7B	300/task	Seeds $\{42, 43, 44\}$; AdamW; constant LR 5×10^{-4}; maximum 200k updates/run; batch 16/GPU; accumulation 1	LoRA rank 32, $\alpha = 16$, dropout 0; 224×224 images; deterministic decoding; used only in Table 2 in the main text

OpenVLA-specific preparation. OpenVLA-7B [1] is initialized from pretrained weights and fine-tuned on 300 MPLib trajectories per task. Common preprocessing removes idle actions with negligible end-effector pose change before fixing the training set; RAISE retains every valid sample in this preprocessed set. We follow the official LoRA fine-tuning configuration in `vla-scripts/finetune.py`, using AdamW with a constant learning rate of 5×10^{-4} and up to 200k optimizer updates per run. Batch and adapter settings are listed in Table A3. Training uses `all-linear` LoRA, BF16 without four-bit quantization, and augmented 224×224 images. Evaluation uses each run's final saved checkpoint and deterministic decoding under Appendix A.1. The $\pi_{0.5}$ systems use a 30k-update training budget.

Appendix A.5 RAISE Integration

Equation (16) in the main text gives fixed offline base weights. The effective weights in Eq. (2) in the main text are

$$\widetilde{w}_{i,s} = 1 + \eta_s c(w_i - 1).$$

The following weighting schedules apply to BC+RAISE and DP+RAISE in Table 1 in the main text and Appendix H.1. For BC+RAISE on PushCube and LiftPegUpright, $\eta_s = 0$ during the first half of training and increases linearly to one during the second half. PushCube BC+RAISE uses $c = 1/4$; all other BC+RAISE and DP+RAISE configurations use $c = 1$. The remaining task–learner combinations use $\eta_s = 1$ throughout training. The effective upper bound is 5/4 for PushCube BC+RAISE and 2 for the other configurations. Policy gradients are computed with the weights treated as constants; sample-selection probabilities remain unchanged.

For BC+RAISE in Table 1 in the main text and Appendix H.1, $H_{\text{w}} = 8$ on StackCube, PushCube, and PlaceApple, and 1 on LiftPegUpright and PegInsertionSide. DP+RAISE uses an endpoint at most 15 steps ahead in these experiments. The $\pi_{0.5}$ risk-source and demonstration-composition studies use $H_{\text{w}} = 10$. Endpoints are clipped to valid recorded states. Default thresholds were set empirically from normalized feature, within-stage progress, and positive-advancement scales before initial policy training. The shared PlaceApple setting for BC+RAISE and DP+RAISE in Table 1 in the main text and Appendix H.1 is selected from $\delta_z = \delta_u \in \{0.15, 0.2, 0.3, 0.4\}$ by matching F1 on 200 positive and 200 negative development state pairs; δ_V and β are unchanged.

Setting every weight to one recovers the unweighted sample-mean loss and its gradient. All valid samples remain in training, including successful retries. Unmatched or non-advancing samples retain unit weight. In the normalized loss, their relative contribution decreases as other samples receive larger weights. For $\pi_{0.5}$, the 10×32 flow-matching residual is averaged within each row before reducing the weighted numerator and denominator over the full logical batch. Device- or microbatch-specific ratios are not averaged.

Progress inference, reference construction, failure localization, and risk matching are completed during offline preparation. Risk annotations and later observations are used exclusively for weight construction. Deployment retains the policy's native inputs and inference procedure, including the fixed GTP-FA grasping front-end where applicable. Auxiliary rollout collection and offline scoring contribute the data and computation costs reported in Appendix H.7.

Appendix A.6 Real-Robot Platform and Evaluation

A Franka Research 3 uses a Robotiq parallel-jaw gripper and base/wrist Intel RealSense D435i cameras; computation is listed in Table A4. The five tasks in Table 3 in the main text use 300 demonstrations and 50 physical trials per method/task. Four methods are evaluated in the same round, totaling 1000 trials. Initial scenes are set independently within the same task rules and placement ranges. Independent-group uncertainty estimates are specified in Appendix H.6. PourWater transfers small balls between cups. Real-robot evaluation uses end-of-execution completion following Xu et al. [2]; simulation evaluation reports Stable Success@3.

Recorded RGB-D, robot/gripper states, grasp poses, and execution histories support perception and retrospective review. Policies retain native inputs, without simulator privileged state, tactile/contact ground truth, future observations, human annotation, or outcome-conditioned action generation. All real-robot demonstrations are collected under human control. Local execution errors and

subsequent recoveries remain within the same recorded trajectory. For example, after a cube is dropped during stacking, the demonstrator regrasps it and continues until task completion. The collection contains successful trajectories with and without retries. All 300 successful demonstrations per task supply policy action targets and are partitioned by complete trajectory into 200 for progress adaptation, 50 for successful references, and 50 for calibration, stratified by retry status with task-stage coverage. These auxiliary subsets are mutually disjoint. The progress model uses base-camera RGB; retry-risk anchors come from the 200 adaptation demonstrations. Real-robot runs use $\mathcal{U} = \{R\}$ and no independent natural-failure rollouts. Reweighting preserves cameras, controller, demonstrations, evaluation conditions, and the fixed GTP-FA front-end. Annotation semantics appear in Appendix D.

Execution timing. End-effector pose targets are updated at 10 Hz. Policy inference is synchronous, with a new prediction after each block of ten control commands. These command-level rates are distinct from the camera and annotation time bases.

Appendix A.7 Reproducibility and Reporting Conventions

Records retain data/checkpoint identities, learner settings, seeds or physical trials, and results. Main three-seed evaluations remain separate from the seed-42 study informing V3 selection. Out-of-memory events, unavailable devices, missing dependencies, simulator crashes, corrupted artifacts, and communication failures are recorded as infrastructure exceptions separately from task failures.

Appendix A.8 Computational Resources

Table A4 lists the hardware installed on each host. Main progress models use LoRA (Table A2); full-parameter ROBOMETER training belongs to a separate development setting. The StackCube auxiliary-preparation timing in Table H11 uses one RTX 4090 (24GB) and the host CPU; this allocation is distinct from the installed GPU counts below.

Table A4 Computational platforms for progress-model adaptation, policy experiments, and robot deployment. GPU counts specify the installed devices on each host.

Primary use	CPU	GPUs
ROBOMETER development and adaptation; VLA training and evaluation	Intel Xeon Gold 6330	8× NVIDIA A100
Task-specific ROBOMETER adaptation; VLA training and evaluation	Intel Xeon Silver 4310	2× RTX 5090
StackCube VLA ablations	AMD Ryzen 9 9950X	2× RTX 4090
Behavior Cloning and Diffusion Policy training and evaluation	Intel Xeon Silver 4310	5× RTX 4090
Real-robot perception and VLA deployment	Intel Xeon Gold 6226R	1× RTX 4090
Policy training and evaluation	Intel Xeon Gold 6530	4× Pro6000; 4× A100

Appendix B Additional Analysis of Retry-Sensitive Progress Learning

We examine global calibration and local retry sensitivity through supervision-strength, data-coverage, directional-loss, and input comparisons. The supervision and data-coverage studies use base-camera 128×128/H16 inputs; the input comparison uses paired A/B models.

Appendix B.1 Evaluation Protocol and Experimental Configurations

Models within each comparison share evaluation trajectories, annotations, and validity masks. Clean successes assess global progress; successful retries assess local degradation–recovery. Targets follow Section 3.2 in the main text and Appendix A.3; set sizes and experiment-specific settings accompany each table.

Global accuracy. For progress-valid indices $\mathcal{I}_{\text{prog}}$ with $N_{\text{prog}} = |\mathcal{I}_{\text{prog}}|$, define

$$\text{MAE} = \frac{1}{N_{\text{prog}}} \sum_{i \in \mathcal{I}_{\text{prog}}} |V_i - g_i^*|, \qquad \text{Acc@}\delta = \frac{1}{N_{\text{prog}}} \sum_{i \in \mathcal{I}_{\text{prog}}} \mathbf{1}[|V_i - g_i^*| \leqslant \delta], \tag{B1}$$

where g_i^* is the fixed target. Stage macro-F1 and terminal-anchor error assess stage classification and deviation from unit progress at completion, using their respective masks.

Retry sensitivity. Using the annotated window means from Section 3.2 in the main text, event satisfaction is

$$s_r = \mathbf{1}[\mu_r^{\text{pre}} > \mu_r^{\text{near}}]\mathbf{1}[\mu_r^{\text{post}} > \mu_r^{\text{near}}]. \tag{B2}$$

Both inequalities are strict, with no additional margin. P_{evt} averages events; P_{traj} first averages within trajectories and then weights trajectories equally. We also report $\Delta_{\text{pre}} = \text{mean}_r(\mu_r^{\text{pre}} - \mu_r^{\text{near}})$, $\Delta_{\text{post}} = \text{mean}_r(\mu_r^{\text{post}} - \mu_r^{\text{near}})$, and recovery consistency, the fraction satisfying $\mu_r^{\text{post}} \geqslant \mu_r^{\text{pre}}$. Recovery consistency is an evaluation-only diagnostic. These metrics quantify progress-model predictions; policy task success is evaluated separately.

Appendix B.2 Data Coverage and Optimization Controls

Coverage. E1–E7 vary training data while holding the architecture fixed (Table B1). E1–E4 fix 300 clean trajectories and increase retry coverage; E2/E5/E6 fix the 299-event retry pool and increase clean coverage. E7 combines 1000 clean trajectories and 1000 retry events. All runs use base-camera 128×128/H16 inputs, $\lambda_{\text{retry}} = 1$, equal directional coefficients, seed 20261221, and 1000 updates. Final-checkpoint metrics and development-window near-best summaries remain separate.

Table B1 StackCube training-data coverage. Evaluation uses 200 clean trajectories and 200 retry events from 150 trajectories. Final columns use update 1000. Near-best summarizes checkpoint performance on development data: it is the median of the three highest preference values in the development evaluation window.

	Training data			Final update 1000		Near-best
ID	Clean traj.	Retry traj.	Retry events	P_{evt} (%)	Stage macro-F1	P_{evt} (%)
E1	300	49	78	24.5	0.8531	29.5
E2	300	219	299	36.5	0.8544	37.0
E3	300	367	500	38.0	0.8541	38.5
E4	300	750	1000	40.5	0.8520	40.5
E5	500	219	299	36.5	0.8511	38.0
E6	1000	219	299	37.5	0.8578	39.5
E7	1000	750	1000	40.5	0.8657	40.5

Retry coverage raises satisfaction from 24.5% to 40.5% (E1–E4), whereas extra clean data with fixed retries change it little (E2/E5/E6). Increasing clean coverage improves stage macro-F1 from 0.8520 to 0.8657 (E4/E7), complementing the gain in retry sensitivity. Retry-event and source-trajectory counts increase together, so E1–E4 measure their joint effect.

Directional-loss controls. With E7 membership and $\lambda_{\text{retry}} = 2$ fixed, coefficients c_{pre} and c_{post} multiply the two terms of Eq. (6) in the main text. Balanced $(1, 1)$ and pre-emphasized $(4/3, 2/3)$ settings preserve the coefficient sum and sampling and use the same two training seeds. Compared with balanced $(1, 1)$ weighting, pre-emphasized $(4/3, 2/3)$ weighting increases Δ_{pre} and reduces Δ_{post}, recovery consistency, and joint preference satisfaction for both seeds (Table B2). Joint preference satisfaction evaluates degradation and recovery together; balanced weighting performs better on this joint criterion for both seeds.

Table B2 E7 directional-loss controls at $\lambda_{\text{retry}} = 2$ on the development sets of Table B1. Values are medians over the development evaluation window. Satisfaction and recovery are percentages.

$(c_{\text{pre}}, c_{\text{post}})$	Seed	P_{evt}	P_{traj}	Δ_{pre}	Δ_{post}	Recovery
$(1, 1)$	20261222	46.0	47.1	0.1209	0.2469	77.0
$(1, 1)$	20261223	45.5	45.3	0.1350	0.2753	74.5
$(4/3, 2/3)$	20261222	43.5	43.9	0.1394	0.2178	70.5
$(4/3, 2/3)$	20261223	44.5	44.7	0.1480	0.2440	68.5

Appendix B.3 Task-Wise Progress-Model Evaluation

Configuration B is adapted separately to each task with the shared recipe in Appendix A.3 and task-specific stages, targets, masks, and retries. Table B3 reports progress-prediction accuracy and retry sensitivity after 1000 task-specific adaptation updates on each of the ten tasks.

Global progress accuracy and local retry sensitivity capture different properties. TransportBox has MAE 0.03278, Acc@0.10 of 94.84%, and retry satisfaction of 56.00%. Retry satisfaction ranges from 47.00% on PegInsertionSide to 99.50% on PullCube. Cross-task values reflect both task difficulty and differences in stage semantics and event distributions. The global and local metrics therefore provide complementary task-specific diagnostics.

Appendix B.4 Qualitative Analysis and Learning Dynamics

The following examples use the fixed final checkpoints of the paired A/B models. Learning curves describe their training trajectories separately.

Clean trajectories. Figure B1 aligns two clean-development trajectories with saved predictions and segment-wise targets. In case01, stage-2 segments at steps 6–17 and 19–21 are separated by a stage-1 singleton at 18. The singleton has no absolute-progress target; re-entry restarts $u_t^* = 0$, yielding $g_t^* = b_2 \approx 0.289$. This target reset is a consequence of segment normalization; identifying a physical setback requires separate execution evidence.

Table B3 Task-specific B models on ten tasks at update 1000. Each development set has 200 clean trajectories and 200 retry events; N_{prog} counts valid progress samples and N_{traj} counts retry-source trajectories. Acc@0.10 and preference satisfaction are percentages.

	Clean development				Retry development		
Task	N_{prog}	MAE ↓	Stage macro-F1 ↑	Acc@0.10 ↑	N_{traj}	P_{evt} ↑	P_{traj} ↑
StackCube	3996	0.05449	0.9061	87.81	149	69.50	68.79
PushCube	2567	0.04357	0.8564	90.73	160	89.50	88.63
PullCube	1848	0.03905	0.6707	92.91	200	99.50	99.50
PickCube	1509	0.08370	0.8273	72.63	182	90.50	90.93
LiftPegUpright	1297	0.07392	0.8848	74.79	184	96.00	97.08
PokeCube	4295	0.09163	0.8145	76.44	198	88.50	88.89
PullCubeTool	4509	0.07964	0.8680	76.18	188	94.00	94.33
PegInsertionSide	5381	0.09053	0.8227	67.85	136	47.00	44.22
PlaceApple	9521	0.08667	0.7379	71.69	187	74.00	75.22
TransportBox	14798	0.03278	0.9208	94.84	193	56.00	56.99

Stage macro-F1 uses the four-slot evaluation convention, including the inactive stage slot for PullCube. N_{prog} is not the denominator for stage classification.

Retry preferences and counterexamples. Figures B2 and B3 show all four paired retry-preference outcomes: A and B both satisfy the preference in case03, B alone in case04, A alone for the selected case05 event, and neither in case06. All four executions ultimately succeed. Preference satisfaction evaluates the ordering of predicted progress, and each plotted point averages a complete annotated window. The displayed frames provide visual context. These post-hoc examples illustrate the four outcomes; their frequencies are measured by the full 200-event comparison in Section 4.10 in the main text.

Learning dynamics. Figure B4 separates training losses from development metrics. Both models reduce losses and clean MAE. After early fluctuations, B maintains higher retry satisfaction through update 1000: 69.5% versus 45.5%, with slightly higher clean MAE (0.054492 versus 0.052883). The fixed-final-checkpoint comparison quantifies the trade-off between local retry sensitivity and global progress accuracy. Failure localization combines progress predictions with state-support and action evidence.

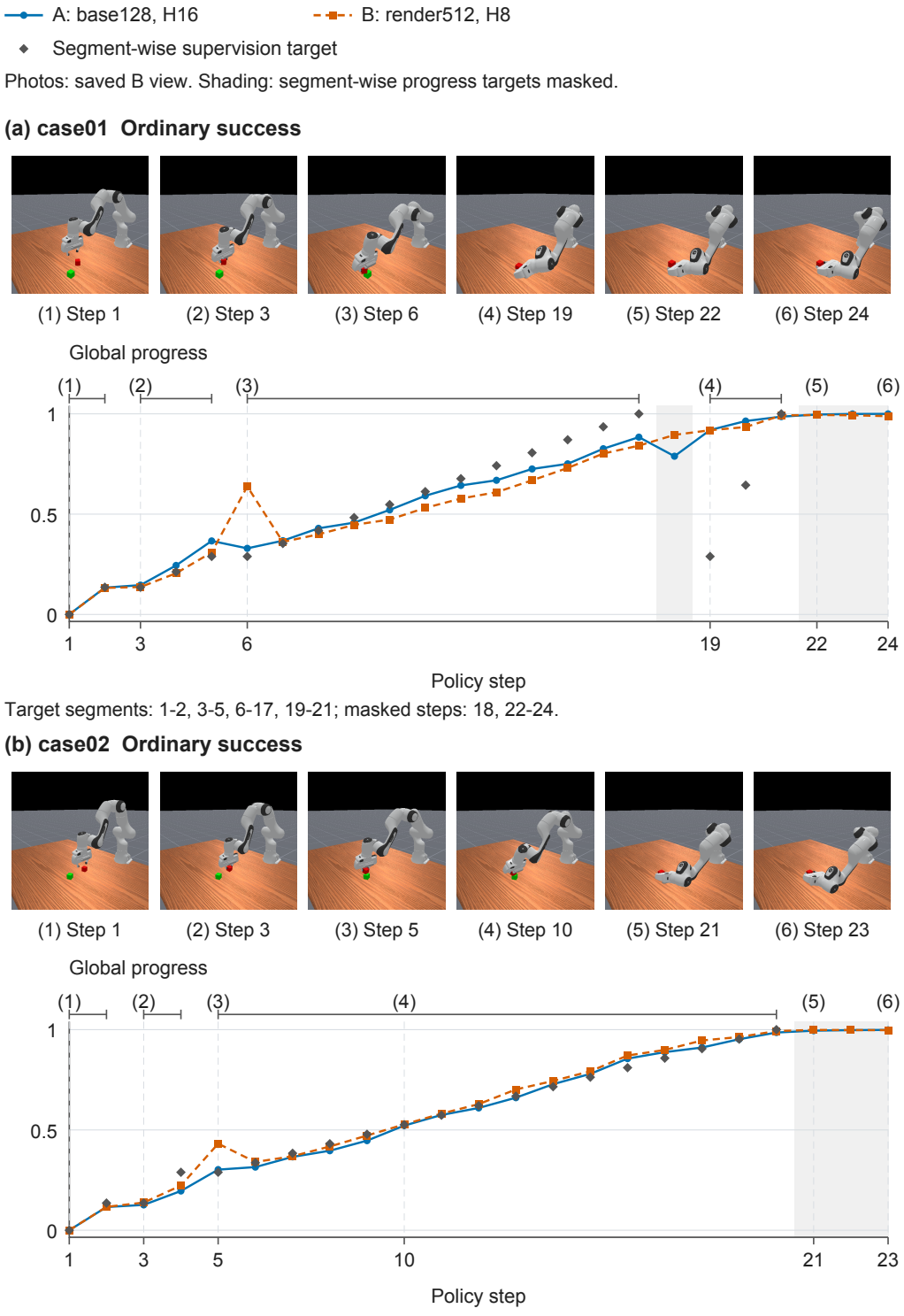

Figure B1 Clean-success predictions and segment-wise targets on StackCube. Numbered B-view images align with policy steps; A/B retain their own inputs. Diamonds mark valid targets and shading excludes absolute-progress supervision only. The case01 reset arises from segment normalization; physical-setback assessment requires separate execution evidence. Saved predictions are unsmoothed and retain their recorded endpoint values.

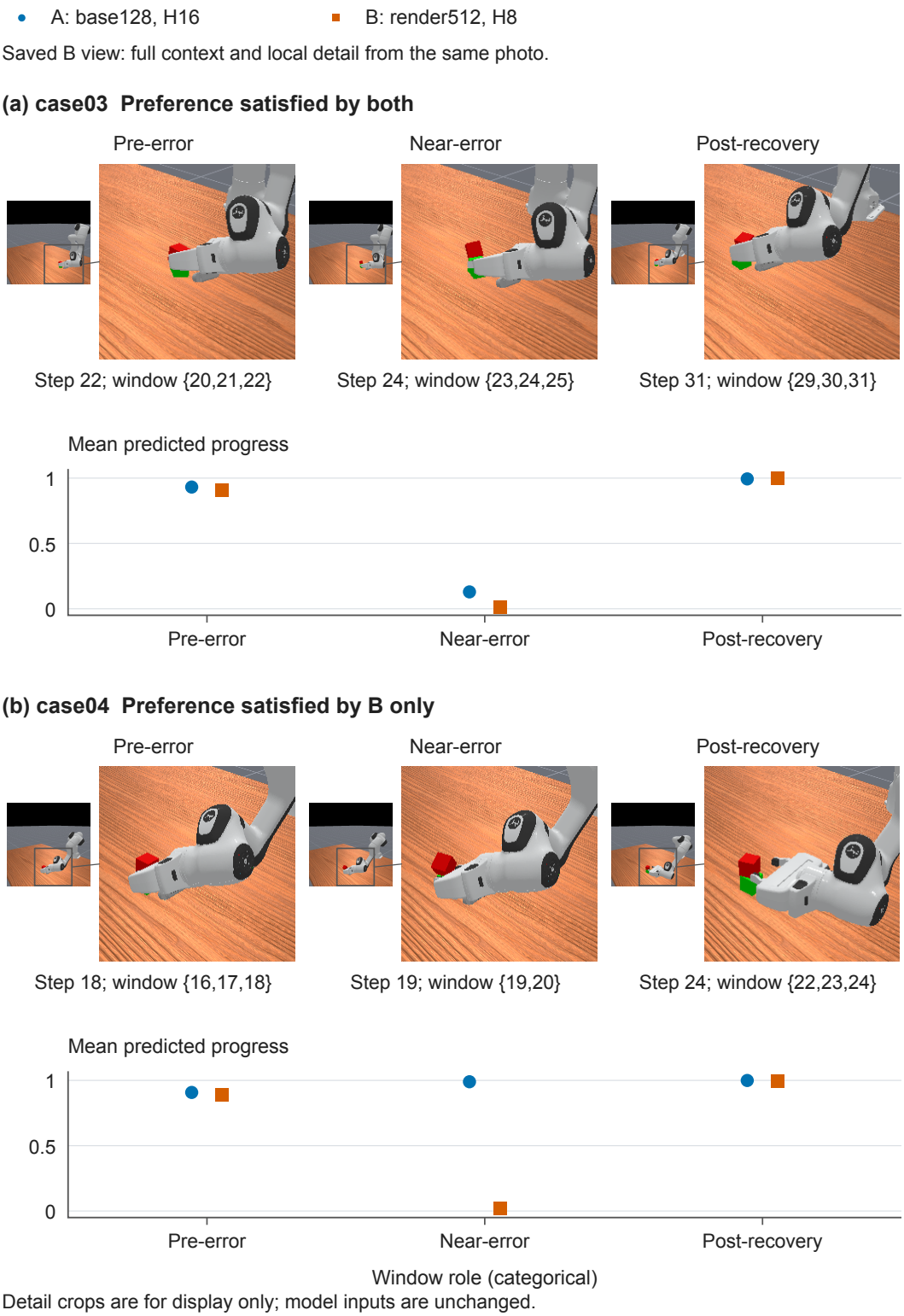

Figure B2 Final A/B retry-window predictions: both satisfy case03; only B satisfies case04. Points average complete annotated windows, with categorical positions identifying window types. Full views and crops share the same B-view image. These post-hoc examples from successful trajectories illustrate event-level prediction ordering.

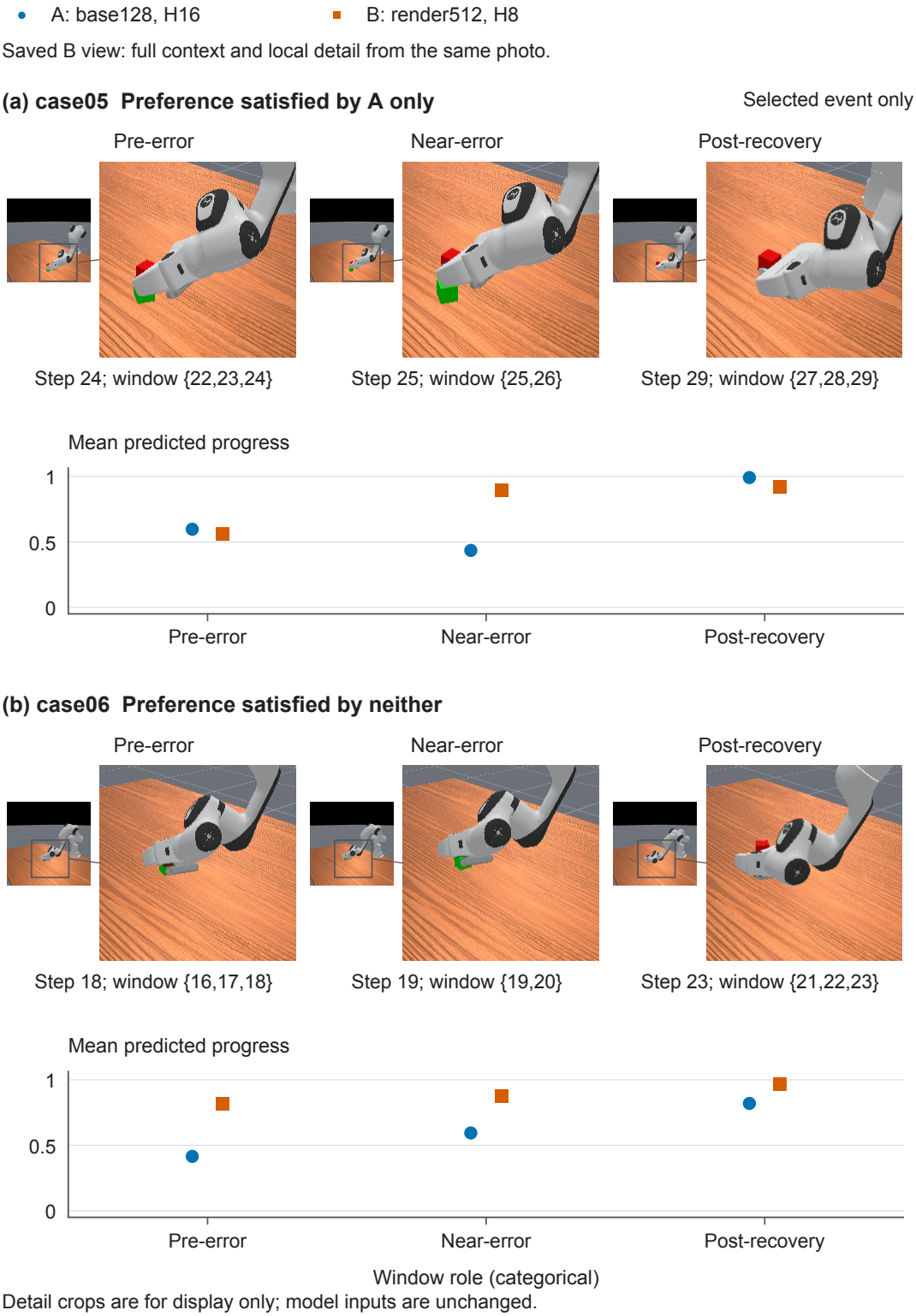

Figure B3 Complementary final-checkpoint cases: only A satisfies the selected case05 event; neither satisfies case06. Both trajectories succeed. Unconnected points are full-window means; paired views are the same saved images. The case05 conclusion concerns its selected event only.

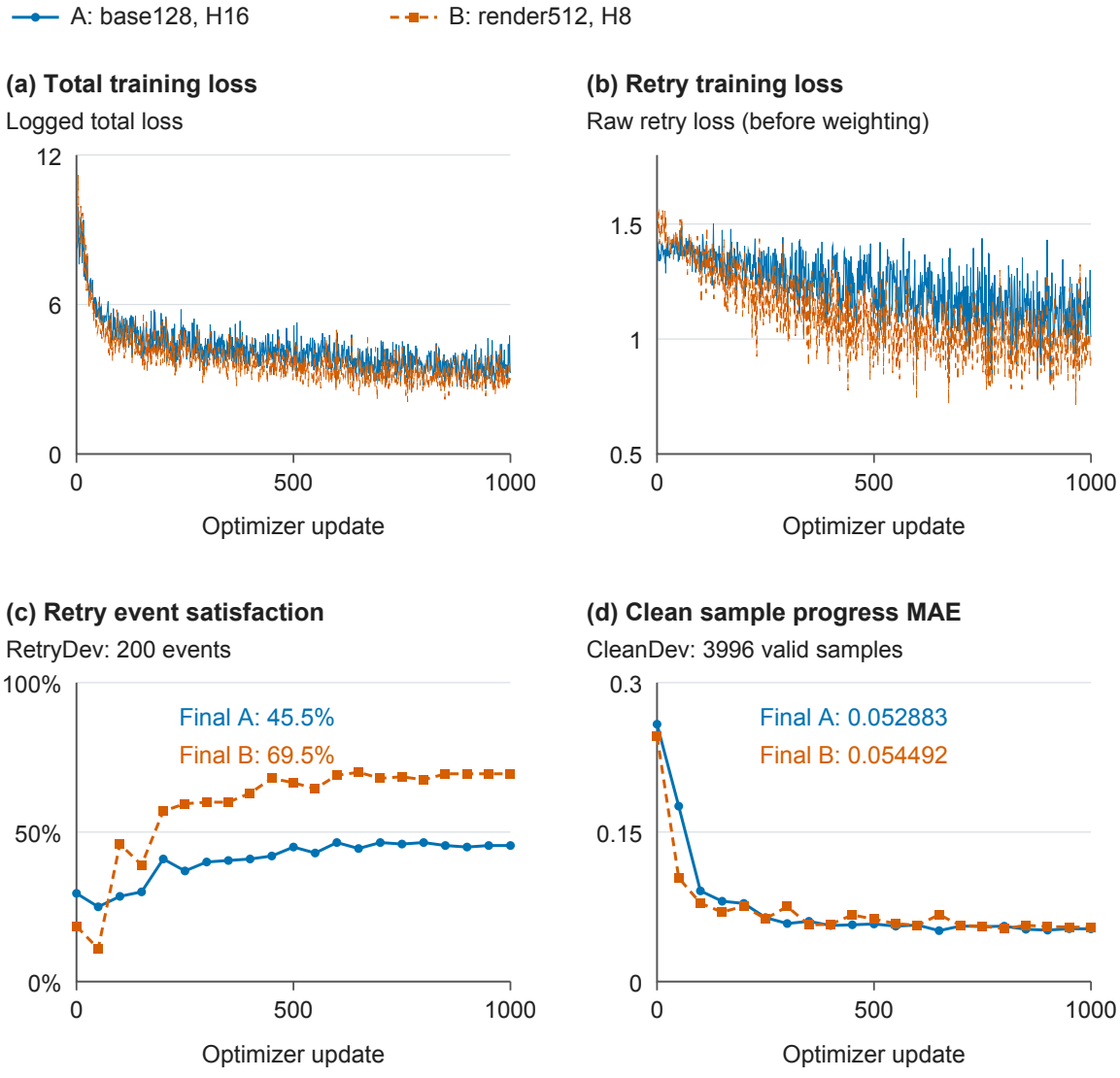

Figure B4 Paired A/B learning dynamics: total loss, raw retry loss before λ_{retry} scaling, retry satisfaction, and clean MAE. Each curve shows unsmoothed values from one training run per configuration. The comparison describes within-run dynamics; across-seed uncertainty remains unmeasured.

Appendix C Success-Referenced Failure Localization

Appendix B examines progress predictions and their remaining errors. We now evaluate how successful references and temporally corroborated evidence support offline failure detection and onset localization. Logged attribution indices are reported alongside onset estimates. Paired A/B input comparisons and the E7L2-01 evidence-rule experiments use separately specified model and evaluation configurations.

Appendix C.1 Evaluation Protocol and Metrics

Data and reference onsets. The paired StackCube study uses four trajectory-disjoint sets: 1000 successful reference trajectories, 1000 successful calibration trajectories, 1000 controlled failures, and 1000 clean evaluation successes. Failures comprise 250 interventions each of wrong direction, action freezing, premature gripper opening, and action perturbation. An example is accepted when the unmodified continuation from the same pre-intervention state reaches stable success but the intervened rollout does not. Its reference $t^*_{a,i}$ is the first actually modified action after bounding and application, not the first altered raw output or later visible consequence. The controlled and clean evaluation sets form the Dev2 development set, used for detector development and visual-input selection.

Model-specific references and calibration. The paired A/B models from Section 4.10 in the main text use update-1000 checkpoints. Trajectory membership and statistical rules are shared, but representations, reference indices, and thresholds are built separately. Retrieval uses five neighbors, ten progress bins, and same-stage expansion. Conditional calibration requires 20 rows, otherwise falling back to stage-level and global nonterminal statistics. Strong state/action thresholds use successful-calibration 99th percentiles; progress uses the stage-conditioned 1st percentile. Confirmation retains five indices, $0.95 \leqslant r_S < 1$ near-boundary support, and Eq. (12) in the main text; refinement uses the weak-action 95th percentile and at most five preceding steps.

Detection and localization metrics. Each trajectory contributes one detection outcome. Confirmation on a controlled failure is a true positive (TP); a trigger on a clean success is a false positive (FP). False negatives (FN) and true negatives (TN) are the respective nontriggered trajectories. Precision and F1 refer to this balanced failure–success mixture. For confirmation indicator $\gamma_i \in \{0, 1\}$, define

$$e_i = \begin{cases} |\hat{t}_i - t^*_{a,i}|, & \gamma_i = 1, \\ 50, & \gamma_i = 0. \end{cases} \tag{C1}$$

For n controlled trajectories,

$$\mathrm{NLE} = \frac{1}{50n} \sum_{i=1}^{n} e_i, \qquad \mathrm{Acc@}k = \frac{1}{n} \sum_{i:\gamma_i=1} \mathbf{1}[e_i \leqslant k], \quad k \in \{3, 5, 10\}. \tag{C2}$$

Misses remain in overall denominators. The fixed value 50 is a miss penalty and normalization scale, not the execution horizon. Conditional Acc@k divides by the number of detected controlled failures. These step-tolerance metrics differ from progress-value tolerances in Appendix B.1.

Uncertainty and timing. Paired 95% confidence intervals use 2000 trajectory-level bootstrap replicates, stratified by four intervention types and clean successes, with seed 20261222. These intervals quantify trajectory-sampling variation conditional on the trained checkpoints; training-seed variation requires separate repeated training runs. Logged $t^{\log}_{\mathrm{dec}}$ is distinct from $\hat{t}_0$ and $\hat{t}$. Because confirmation can use later observations, differences between these indices describe offline attribution timing.

Appendix C.2 Paired Detection and Localization Results

B detects 88 additional failures and adds eight false positives (Table 11 in the main text). Paired intervals exclude zero for the 8.60-point Acc@10 gain and 0.08592 NLE reduction on Dev2. The FPR difference is +0.80 percentage points, with a 95% interval of $[-1.90, +3.30]$. B's 69.0% recall differs from its 49.4% overall Acc@10; conditional Acc@10 is 71.59% on B's 690 detected failures. Overall metrics retain all 1000 failures, whereas conditional metrics describe each model's detected subset.

Table C1 Dev2 results by intervention, with 250 failures per type. Recall and Acc@10 are percentages. NLE and overall localization retain misses according to Eq. (C2).

	Recall ↑		NLE ↓		Acc@10 ↑	
Intervention	A	B	A	B	A	B
Wrong direction	72.4	74.0	0.34376	0.34216	60.4	57.2
Action freezing	51.2	57.2	0.59824	0.57776	32.0	25.6
Premature gripper opening	62.0	81.6	0.49032	0.25592	41.2	71.2
Action perturbation	55.2	63.2	0.56496	0.47776	29.6	43.6

Table C2 Reference-bank and calibration-set sizes with the E7L2-01 checkpoint, evaluated on 500 controlled failures and 500 clean successes. G1–G4 vary bank/calibration trajectory counts. A1–A3 denote the evidence gates defined in Appendix C.4. FP counts trajectories; Acc@10 and FPR are percentages.

Group: bank/calibration	Gate	TP	FP	NLE ↓	Acc@10 ↑	FPR ↓
G1: 500/500	A1	283	143	0.54844	38.2	28.6
	A2	243	139	0.62460	29.8	27.8
	A3	14	14	0.97816	1.2	2.8
G2: 500/1000	A1	277	134	0.56016	36.6	26.8
	A2	243	132	0.62284	30.0	26.4
	A3	12	6	0.98264	0.8	1.2
G3: 1000/1000	A1	277	134	0.56016	36.6	26.8
	A2	236	133	0.63516	28.8	26.6
	A3	12	6	0.98264	0.8	1.2
G4: 1000/500	A1	283	143	0.54844	38.2	28.6
	A2	235	138	0.63944	28.2	27.6
	A3	14	14	0.97916	1.0	2.8

Appendix C.3 Failure-Type Analysis

B has higher recall for each intervention type (Table C1). Acc@10 increases for premature gripper opening (+30.0 points) and action perturbation (+14.0), and decreases for wrong direction (−3.2) and action freezing (−6.4). Recall measures detection coverage, NLE measures penalized onset error, and Acc@10 measures localization within a fixed tolerance. These controlled StackCube comparisons show that detection and onset localization respond differently to intervention type.

Appendix C.4 Development Analyses of Evidence Rules and Reference Sets

The reference-set and evidence-rule studies use the E7L2-01 checkpoint and a shared cohort of 500 controlled failures and 500 clean successes. The Dev2 onset-refinement study reports the E7L2-01 configuration and a paired comparison with the fixed final B checkpoint separately (Table C4).

Reference-set size and evidence gates. A1 uses progress alone. A2 combines progress with successful-state support at the query index. A3 additionally requires a raw-action anomaly at the same index (Table C2). A1–A3 name evidence rules; A/B name visual-input configurations. Under G3, A3 detects 12/500 failures with 1.2% FPR, whereas A1 detects 277/500 with 26.8% FPR. The low recall of A3 persists across all four reference-set sizes.

Temporal and action controls. With applied-action evidence, S0 requires same-index conjunction; R0 permits windowed three-type evidence; R1 uses action/support-initiated two-type confirmation; R2 also permits progress initiation (Table C3). Compared with R0, R1 improves detection and localization while increasing FPR. Relative to R1, R2 adds 0.6 Acc@10 points and 2.2 FPR points. State-conditioned action evidence raises Acc@10 and FPR relative to global action evidence at both $W = 5$ and $W = 7$. S95 adds near-boundary support. Compared with S95, S95 + D2 reduces FPR from 15.8% to 13.6% and changes Acc@10 from 62.2% to 60.8%.

Onset refinement. E7L2-01 variants C0–C5 share identical detection outcomes (Table C4(a)). The adopted five-step C2 rule lowers NLE from 0.30250 to 0.28402 and raises Acc@10 from 59.8% to 62.9% relative to C0; longer look-backs attain higher point estimates on several metrics. For the fixed final B checkpoint (Table C4(b)), C2 lowers NLE from 0.44310 to 0.41340 and increases Acc@3, Acc@5, and Acc@10 by 7.2, 3.9, and 6.8 percentage points, respectively, while recall and FPR remain unchanged at 69.0% and 13.0%. These comparisons hold detection outcomes and the logged decision index fixed while changing the reported onset. The gains measure offline onset-localization accuracy.

Appendix C.5 Temporal Evidence and Qualitative Cases

Evidence coordinates. Figures C1 and C2 show B's saved evidence for three action-perturbation rollouts and one clean success, selected post hoc for accurate attribution, offset attribution, a miss, and a false alarm. Four saved images accompany applied-action deviation, support-distance ratio, and progress regression. Orange curves are evidence, dashed curves are thresholds, the weak support boundary is 0.95, and missing indices remain disconnected. For display, action evidence at index t appears at state $t + 1$, support at t, and g_t at $t + 3$, the last required future state. Ref., Initial, C2, and Log D retain the recorded indices of t_a^*, $\hat{t}_0$, $\hat{t}$, and $t_{\text{dec}}^{\log}$. These display coordinates align the evidence for offline inspection; online decision timing would require tracking when each observation becomes available.

Table C3 Evidence-rule controls with E7L2-01 on a shared cohort of 500 controlled failures and 500 clean successes. W denotes the temporal confirmation window. TP counts trajectories; FPR and Acc@10 are percentages. R1 in panel (a) and Global action at $W = 5$ in panel (b) reuse the same evaluations.

Setting	W	TP	FPR ↓	NLE ↓	Acc@10 ↑
(a) Temporal evidence rules					
S0: same-index conjunction	–	12	1.0	0.98048	1.4
R0: windowed three-type	5	45	0.8	0.92748	5.4
R1: action/support anchor	5	300	9.8	0.52880	36.0
R2: any evidence anchor	5	312	12.0	0.51252	36.6
(b) Global versus state-conditioned action					
Global action	5	300	9.8	0.52880	36.0
State-conditioned action	5	392	12.8	0.33540	57.2
Global action	7	308	10.8	0.50764	38.6
State-conditioned action	7	397	12.8	0.31696	60.2
(c) Weak support with state-conditioned action					
S95	5	412	15.8	0.29220	62.2
S95 + D2	5	398	13.6	0.31336	60.8

Table C4 Onset refinement on StackCube Dev2. (a) E7L2-01 refinement controls: TP/FN/FP/TN are 821/179/116/884. q_A is the weak-action quantile, L the look-back in steps, and C5 permits a support precursor. (b) Matched comparison using the fixed final B checkpoint, with detection outcomes held unchanged. All Acc@k values include all 1000 controlled failures; recall uses these failures and FPR uses 1000 clean successes. Rates are percentages.

(a) E7L2-01 refinement controls

Variant	q_A	L	Support precursor	NLE ↓	Acc@3 ↑	Acc@5 ↑	Acc@10 ↑
C0	–	0	No	0.30250	41.1	48.7	59.8
C1	0.975	5	No	0.29090	43.9	51.3	61.7
C2	0.950	5	No	0.28402	46.4	52.4	62.9
C3	0.975	10	No	0.27694	45.8	54.9	65.7
C4	0.950	10	No	0.26926	47.9	55.4	67.3
C5	0.950	10	Yes	0.26822	47.1	55.5	67.6

(b) Fixed final B: effect of onset refinement

Reported onset	Recall ↑	FPR ↓	NLE ↓	Acc@3 ↑	Acc@5 ↑	Acc@10 ↑
No refinement ($\hat{t}_0$)	69.0	13.0	0.44310	29.2	37.8	42.6
C2 refinement ($\hat{t}$)	69.0	13.0	**0.41340**	**36.4**	**41.7**	**49.4**

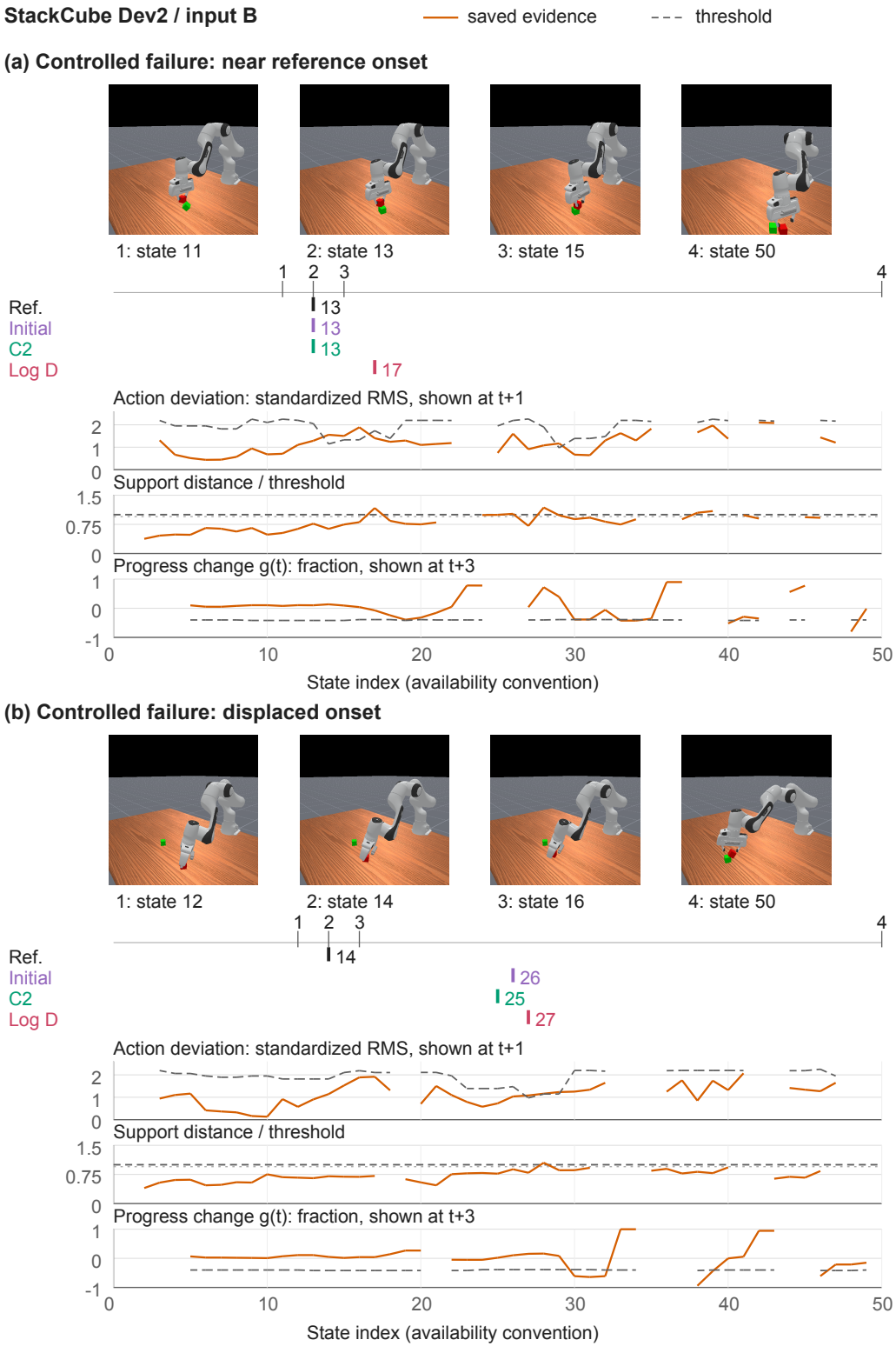

Figure C1 B evidence on action-perturbation rollouts. (a) Initial and refined onsets equal reference 13; Log D is 17. (b) Refinement from 26 to 25 leaves an 11-step error relative to reference 14. Log D is 27; progress confirmation uses observations through state 30. Orange curves are saved evidence, dashed curves are thresholds, and gaps denote unavailable evidence. Log D denotes the offline attribution index. The two selected examples show accurate and offset onset estimates; aggregate localization rates appear in Table 11 in the main text.

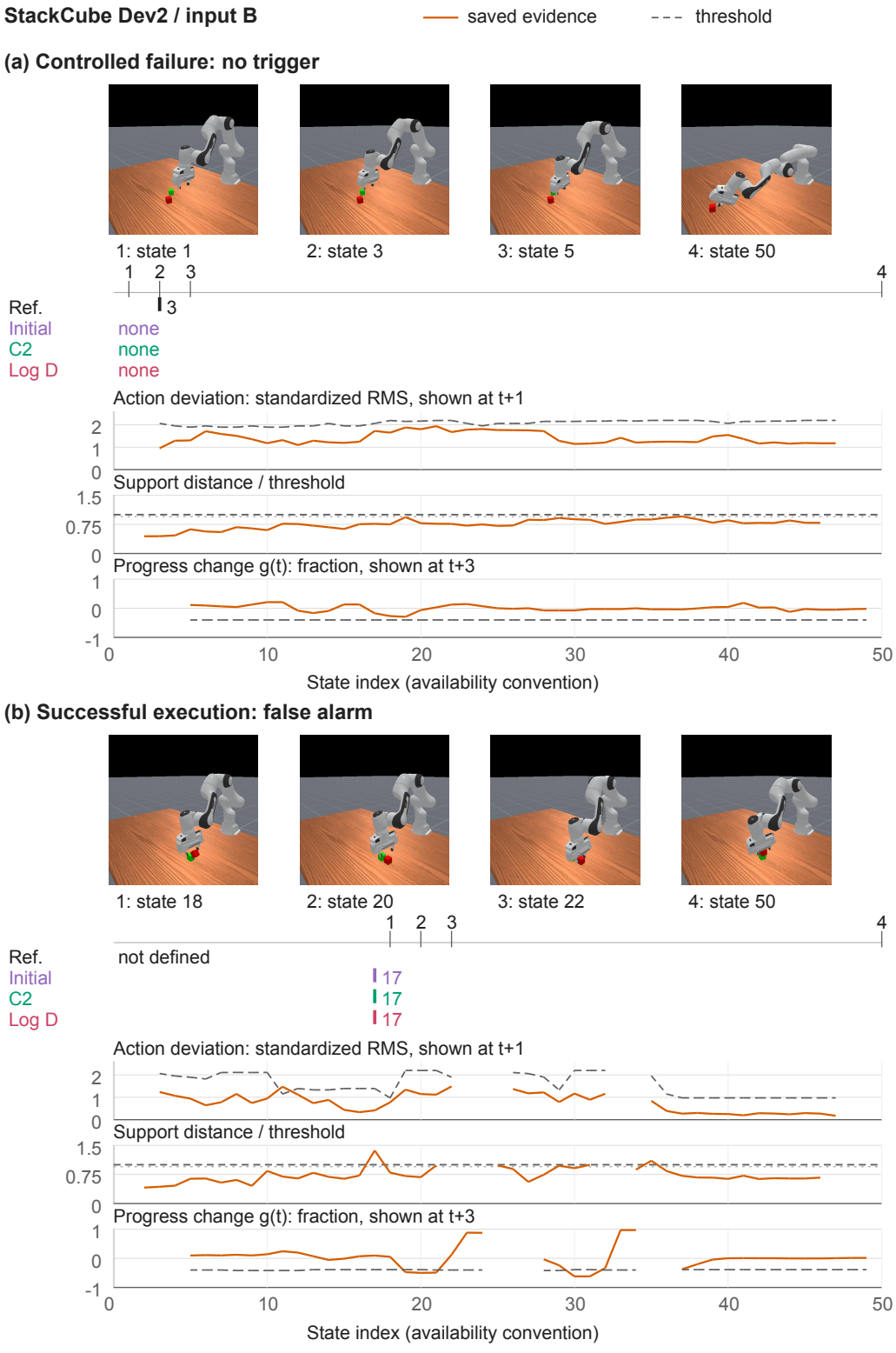

Figure C2 B counterexamples. (a) The reference-3 intervention is missed. (b) A clean success triggers with initial onset, refined onset, and Log D all 17; confirming progress requires state 20, and no failure reference exists. The zero logged gap is computed from offline attribution indices. Mean logged gaps are 2.397 for A and 2.230 for B, each over its own triggered subset. Natural-failure anchors are estimated risk locations; this example shows a false-positive estimate.

Appendix D Event Annotation and Human Review

This appendix defines task events and evaluates human annotation agreement and automatic–human temporal correspondence. Appendix C evaluates controlled failure localization, while policy-success metrics evaluate task completion.

Appendix D.1 Task Stages and Event Definitions

Stages describe execution structure; retries describe local departures. Table D1 lists the real-robot task stages and separate completion criteria. A repeated visit retains the same stage label. A local failure is an observable setback, such as a missed or lost grasp or a disrupted tool–object relation. Correction onset marks remedial action, and recovery re-establishes the relevant relation. Completion follows the task-level criterion: regrasping a dropped cube restores the grasp, while completing StackCube also requires stacking. Routine adjustment alone is not a retry.

Table D1 Real-robot annotation stages and separate completion criteria. PourWater transfers small balls between cups.

Task	Process stages	Completion criterion
StackCube	Approach and grasp; transport and align; stack and release.	The upper cube is stacked, released, and stable.
PokeCube	Acquire the rod; align its tip; push the cube toward the target.	The cube reaches the target region and the robot is stationary.
PullCubeTool	Acquire the tool; position the hook; pull the cube.	The cube reaches the designated retrieval region.
Orange-to-Tray	Approach and grasp; transport and align; place and release.	The released orange remains at the target location in the tray.
PourWater	Acquire the container; position it; pour; finish the transfer.	At least 90% of the initial small balls reach the receiving cup.

Appendix D.2 Annotation Procedure and Visual Interface

Task-specific predicates applied to recorded privileged physical states automatically generate simulation stage, retry, and completion labels. A separate, one-time annotation-validation study reviewed 300 StackCube trajectories. The other nine simulation tasks used automatic labels without trajectory-level manual annotation. The manual study provides a one-time assessment of annotation consistency and correspondence for StackCube.

Failure onset, correction onset, recovery, and the potential-based retry keypoint retain distinct meanings; the labeler stores degradation and recovery boundaries separately. These physical signals are used exclusively to construct supervision (Appendix A.3).

For the five real-robot tasks, two annotators independently label the same 300 demonstrations per task, covering 1500 trajectories in total. Appendix H.6 distinguishes this full annotation collection from the 300-trajectory agreement-study subset and reports the throughput-based effort estimate. The real-robot workspace combines base/wrist views, sparse event labels, temporal navigation, and notes (Fig. D1). Labels are assigned to confirmed annotation-grid frames; all unmarked frames retain an unlabeled status. The displayed 10 Hz grid, source-frame indices, timestamps, and policy-action indices require explicit temporal mapping. For StackCube, the current frame is the third of a visual three-frame stacking, release, and stability check; other tasks use Table D1. This check provides a visual annotation reference. Simulation labels use physical-state predicates, while policy success follows the task-specific evaluation protocol.

Appendix D.3 Agreement on Retry Presence

Two annotators independently review retry presence on 300 simulated StackCube trajectories and an audit subset of 300 physical trajectories from the five-task demonstration collection in Table H9. In the one-time StackCube validation study, each annotator spent approximately five hours, and a third annotator spent approximately three hours adjudicating disagreements. The one-time StackCube validation study therefore required approximately 13 person-hours. The third annotator's adjudication is confined to this study; agreement is computed from the initial annotations as specified below. The unit of analysis is one binary retry-presence judgment per trajectory. Agreement is computed from the two annotators' initial, independent binary judgments before adjudication. Both domains contain 300 complete annotation pairs. All reviewed trajectories, including disagreements, enter the calculation.

Let n_{uv} count responses $u, v \in \{0, 1\}$, where 1 denotes retry presence, and $N_{\text{pair}} = \sum_{u,v} n_{uv}$ among N_{review} reviewed trajectories. We report

$$\begin{aligned} \text{Coverage} = \frac{N_{\text{pair}}}{N_{\text{review}}}, \quad P_o = \frac{n_{11} + n_{00}}{N_{\text{pair}}}, \quad \kappa = \frac{P_o - P_e}{1 - P_e}, \\ P_e = \frac{(n_{11} + n_{10})(n_{11} + n_{01}) + (n_{01} + n_{00})(n_{10} + n_{00})}{N_{\text{pair}}^2}. \end{aligned} \tag{D1}$$

Coverage measures the fraction of reviewed trajectories with a valid annotation pair; agreement measures consistency between the two human judgments, with both annotators treated symmetrically. Table D2 reports these two quantities.

Simulation annotations agree on 193 of 300 trajectories, with 107 disagreements, $P_o = 64.33\%$, and $\kappa = 0.1830$. These disagreements show variation in visual retry judgments. Physical annotations agree on 296 of 300 trajectories, with four disagreements, $P_o = 98.67\%$, and $\kappa = 0.4932$. In the physical subset, each annotator identifies retries in 296 of 300 trajectories. This class distribution yields high chance-expected agreement; the two jointly negative cases provide limited evidence about agreement on retry-absent trajectories. These

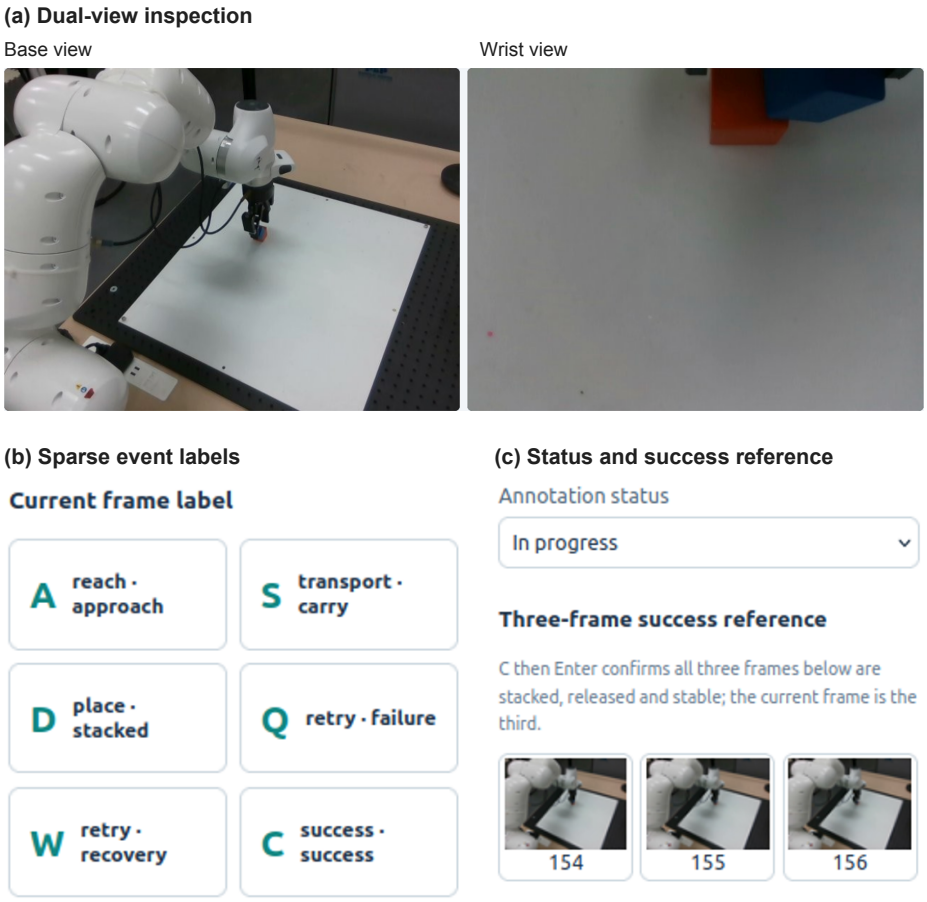

Figure D1 Real-robot annotation workspace: dual-view inspection, sparse event labels, completion reference, and grid/source-time mapping. Panels are display crops from one interface screenshot. Unmarked frames are unlabeled, and the 10 Hz annotation grid is distinct from policy-action indices.

Table D2 Two-annotator judgments of retry presence. Each domain contains 300 reviewed trajectories with complete binary annotation pairs (1: present, 0: absent). All pairs are included. Coverage and observed agreement P_o are percentages.

Domain	Reviewed	Valid pairs	Excluded	Coverage (%)
Simulation	300	300	0	100.00
Real robot	300	300	0	100.00

Domain	n_{11}	n_{10}	n_{01}	n_{00}	P_o (%)	κ
Simulation	150	51	56	43	64.33	0.1830
Real robot	294	2	2	2	98.67	0.4932

statistics characterize annotation agreement on the respective reviewed sets. Task composition, observations, and retry prevalence differ between the two domains.

Appendix D.4 Temporal Correspondence with Human References

As part of the one-time StackCube annotation-validation study, we compare automatic and human stage anchors, local failure onsets, recovery points, and completion confirmations. Each category c uses M_c corresponding annotation pairs on the same simulation-step axis:

$$\mathrm{TA}_c(k) = \frac{1}{M_c} \sum_{j=1}^{M_c} \mathbf{1}[|t_{c,j}^{\mathrm{auto}} - t_{c,j}^{\mathrm{human}}| \leqslant k], \qquad k \in \{2, 10\}. \tag{D2}$$

The sampling unit is a paired event, and the reported percentage measures temporal agreement conditional on matched annotations. Detection recall would additionally require accounting for unmatched events.

Table D3 StackCube automatic–human temporal correspondence on matched event annotations. Each pair contains one human timestamp and one automatic timestamp. M counts these pairs and Q_k counts those within k simulation steps; $\mathrm{TA}(k) = 100Q_k/M$.

Annotation type	M	Q_2	TA(2) (%)	Q_{10}	TA(10) (%)
Stage anchor	400	310	77.50	395	98.75
Local failure onset	126	94	74.60	123	97.62
Recovery point	122	90	73.77	120	98.36
Completion confirmation	146	127	86.99	145	99.32

Table D3 shows agreement rates of 73.77%–86.99% within two steps and 97.62%–99.32% within ten steps. Completion has the highest two-step agreement; failure and recovery boundaries vary more. The two-step and ten-step tolerances distinguish fine-grained and coarse correspondence. Explaining individual discrepancies requires event-level inspection. Human–human presence agreement, automatic–human timing, and controlled failure localization measure label consistency, temporal correspondence, and localization accuracy, respectively. A unique causal onset for natural failures remains unverified by these measurements.

Appendix E Risk-Guided Reweighting and Policy Learning

Following the localization and annotation analyses in Appendix C and Appendix D, we examine how risk evidence shapes learning from successful demonstrations. Appendix E.1–Appendix E.4 analyze offline weights, risk sources, and demonstration composition for StackCube $\pi_{0.5}$ with training seed 42. Section 4.6 in the main text presents BC/DP component and weight-assignment controls in Table 5(a) in the main text and $\pi_{0.5}$ component ablations in Table 5(b) in the main text.

Appendix E.1 Data Sources and Reweighting Protocol

Data sources. The motion-planning Expert300 contains 300 successful trajectories and 31,707 valid action-supervision rows. *Natural mix* denotes this unstratified Expert300 action-demonstration collection. Auxiliary retries provide 1000 pre-error anchors from 729 trajectories; 352 of 489 natural failures yield confirmed anchors, and 137 yield none (Table E1). Only Expert300 supplies policy action targets.

Table E1 StackCube Expert300 and auxiliary risk sources. Trajectories, policy-training rows, and risk anchors are different units; only Expert300 supplies action targets.

Source	Trajectories	Contribution
Expert300	300	31,707 valid action-supervision rows.
Successful retries	729	1000 event-specific pre-error risk anchors.
Natural failures	489	352 confirmed anchors; 137 untriggered rollouts.

Weights and loss. The frozen update-1000 B model uses render-camera 512×512/H8. Each risk source independently returns one nearest same-task, same-stage anchor by cosine distance, then applies $\delta_z = \delta_u = 0.15$; a rejected anchor is not replaced. Advancement uses $t_i^+ = \min(t_i + 10, T_i)$, $\delta_V = 0.12$, and $\beta = 1$, yielding weights in $[1, 2]$ through Eqs. (14) in the main text–(16) in the main text. For $\pi_{0.5}$, V0 uses uniform weights, V1 uses retry risk, V2 uses natural-failure risk, and V3 (RAISE) takes their maximum before the shared advancement gate. All rows remain, including 2700 short terminal chunks; padding adds no physical transitions. Loss reduction follows Appendix A.5: the 10×32 residual is averaged per row and the weighted ratio is computed over the full logical batch, preserving targets, normalization, and sampling probabilities.

Appendix E.2 Distribution of Training Emphasis

Candidate availability, match acceptance, and enhancement ($w_i > 1$) are distinct. A boundary match can have zero risk, and nonpositive advancement suppresses enhancement. Table E2 uses all 31,707 rows. V1/V2/V3 enhance 8634/15,485/16,177 rows; their unit-weight fractions are 72.77%/51.16%/48.98%.

Table E2 Offline $\pi_{0.5}$ weight statistics over all 31,707 unique Expert300 rows. V3 availability/acceptance requires either source to qualify. Top-10 share is the fraction of extra weight $\sum_i(w_i - 1)$ attributed to the ten largest anchors.

Statistic	V1: Retry	V2: Natural	V3: Combined
Candidate coverage (%)	100.00	99.00	100.00
Accepted matches (%)	46.90	76.95	80.69
Enhanced rows, $w > 1$ (%)	27.23	48.84	51.02
Mean raw weight	1.0477	1.1008	1.1116
95th-percentile raw weight	1.3232	1.5136	1.5383
Maximum raw weight	1.8444	1.8669	1.8669
Top-10 anchor extra-weight share (%)	66.06	37.50	31.69

Figure E1 includes the mass at $w = 1$. The shared advancement gate gives $w_i^{\mathrm{V3}} = \max(w_i^{\mathrm{V1}}, w_i^{\mathrm{V2}})$, which determines the ordering of the cumulative weight distributions. The ten largest anchors contribute 66.06%, 37.50%, and 31.69% of extra weight for V1/V2/V3. V3 attributes extra weight to the larger-risk source and splits positive exact ties equally.

Matching and advancement examples. Three post-hoc rows from one expert trajectory illustrate distinct cases. At row 3, accepted natural risk 0.1550 and $q_i = 0.2985$ yield $w_i^{\mathrm{V3}} = 1.0463$. At row 19, an accepted natural match but $\Delta V_i = -0.06757$ gives unit weight. At row 0, advancement is positive but both candidates are rejected, also giving unit weight. For each condition, we select the first matching example in training-row order.

Appendix E.3 Risk-Source Ablation

The four $\pi_{0.5}$ configurations V0–V3 use Expert300 and train independently from `pi05_base`, with seed 42, global batch 32, and 30,000 updates. The risk-source comparison in Table E3(a) evaluates the final checkpoints on the same 200 development scenes, with a 250-step horizon and ten-action chunks. These development scenes are disjoint from the held-out test scenes used for Table 1 in the main text.

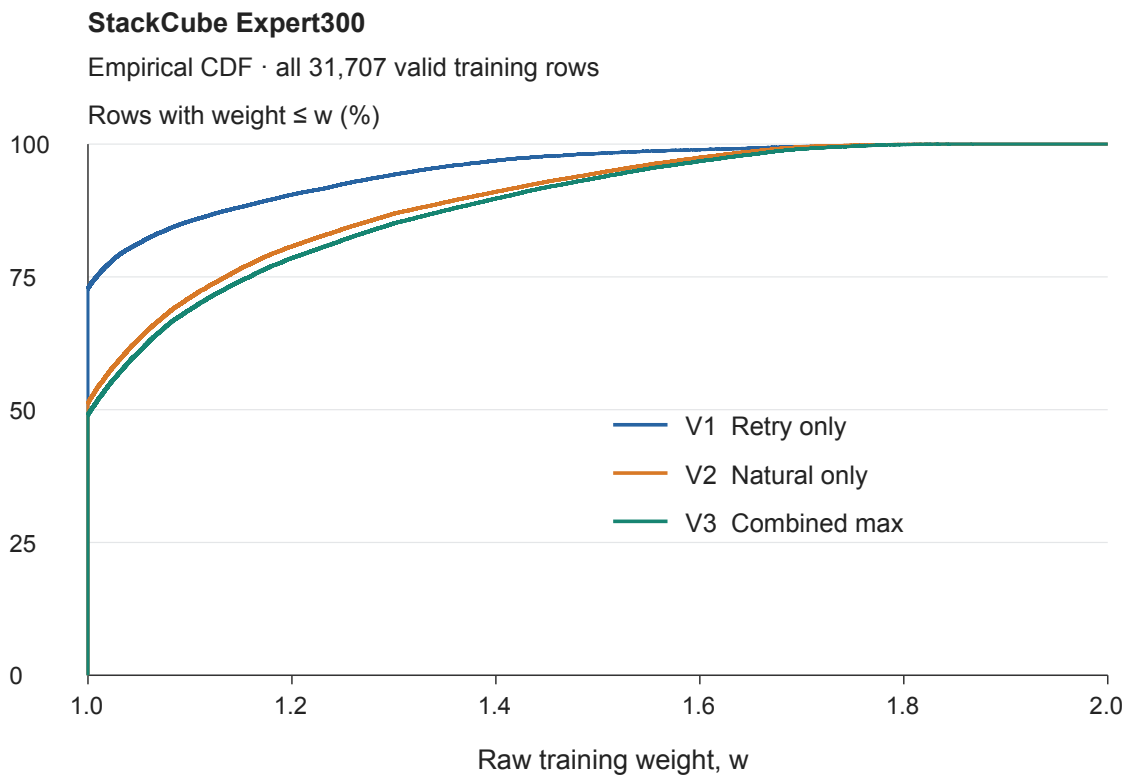

Figure E1 Empirical raw-weight distributions over all 31,707 Expert300 rows, including unit weights. V1 uses retries, V2 natural failures, and V3 their maximum before the shared advancement gate. Saved weights are shown without smoothing or distribution fitting.

Panel (a) varies only risk sources; panel (b) compares V0/V3 on each demonstration composition. All differences are relative to the V0 trained on the same dataset.

On Expert300, RAISE ($\pi_{0.5}$, V3) and uniformly trained $\pi_{0.5}$ (V0) both achieve 96.0% under the two success criteria. The $\pi_{0.5}$ retry-risk-only configuration (V1) is four percentage points lower, and the natural-failure-risk-only configuration (V2) is 0.5 percentage points lower.

Variant selection. On the development scenes in Table E3(a), V3 achieves 96.0% Stable Success@3, matching V0 and exceeding both single-source configurations. We select V3 as the combined-risk configuration and fix its configuration and evaluation checkpoints before held-out testing.

Appendix E.4 Demonstration-Composition Analysis

Composition datasets. G00/G25/G50 contain 0%/25%/50% retry-success trajectories. Each has 300 successes collected with the same frozen V0 policy and retains all 250 actions per trajectory, giving 75,000 rows. Across datasets there are 450 unique trajectories: adjacent mixtures share 225 each, and G00/G50 share 150. Within each dataset, V0/V3 share membership, normalization, initialization protocol, and budget, but train independently. The progress model and external risk pools remain fixed; weights are recomputed for each dataset's rows. G00 contains clean policy demonstrations and retains the shared auxiliary risk pool, including retries. Natural mix comes from a separate collection source and serves as an unstratified reference alongside the ordered G00/G25/G50 comparison.

Results and interpretation. Compared with uniformly trained $\pi_{0.5}$ (V0), RAISE ($\pi_{0.5}$, V3) improves Stable Success@3 by 0.0, 1.5, and 3.0 percentage points on G00/G25/G50 (Table E3(b)); the latter two gains correspond to three and six additional stable episodes. Success-once changes are -0.5, $+0.5$, and $+5.0$ percentage points. Across these three compositions, RAISE's gain increases with the retry fraction, while absolute stable success decreases for both V0 and V3. Reweighting partially offsets this decrease. G50-V3 achieves success-once in 190 episodes and stable success in 185.

Relation to the main evaluation. Table 1 in the main text uses each task's 300 expert demonstrations with their collected retry composition. The StackCube $\pi_{0.5}$ pair uses the 300 Natural mix demonstrations with their collected retry composition and averages independently trained models with seeds $\{42, 43, 44\}$. Each model is evaluated for at most 250 environment steps on the same 200 held-out scenes, using environment seeds `400005010--400005209`. These scenes are disjoint from the development scenes used for V3 selection and are reserved for final performance evaluation. The composition study in this subsection uses training seed 42.

Table E3 StackCube risk sources and demonstration composition. (a) Expert300 risk-source selection on 200 development scenes; (b) controlled retry compositions and the unstratified Expert300 reference (Natural mix). Entries are successes/200 with percentages in parentheses; gains are relative to each dataset's V0. Natural mix uses a separate collection source.

(a) Expert300: risk-source ablation

Variant	Risk source	Once	Stable@3	Δ_{once} (pp)	Δ_{stable} (pp)
V0	None	192 (96.0)	192 (96.0)	0.0	0.0
V1	Retry	184 (92.0)	184 (92.0)	−4.0	−4.0
V2	Natural	191 (95.5)	191 (95.5)	−0.5	−0.5
V3	Combined	192 (96.0)	192 (96.0)	0.0	0.0

(b) Demonstration composition and unstratified Expert300 reference

Dataset	Variant	Clean/retry	Once	Stable@3	Δ_{once} (pp)	Δ_{stable} (pp)
G00	V0	300/0	194 (97.0)	192 (96.0)	0.0	0.0
G00	V3	300/0	193 (96.5)	192 (96.0)	−0.5	0.0
G25	V0	225/75	190 (95.0)	187 (93.5)	0.0	0.0
G25	V3	225/75	191 (95.5)	190 (95.0)	+0.5	+1.5
G50	V0	150/150	180 (90.0)	179 (89.5)	0.0	0.0
G50	V3	150/150	190 (95.0)	185 (92.5)	+5.0	+3.0
Natural mix	V0	Unstratified	192 (96.0)	192 (96.0)	0.0	0.0
Natural mix	V3	Unstratified	192 (96.0)	192 (96.0)	0.0	0.0

Appendix F ManiSkill Benchmark Comparisons

This appendix compares GTP-FA-$\pi_{0.5}$ and RAISE (GTP-FA-$\pi_{0.5}$) with published results on shared ManiSkill tasks, complementing the analyses in Appendix E. The matched GTP-FA-$\pi_{0.5}$–RAISE (GTP-FA-$\pi_{0.5}$) pair measures the effect of reweighting. Cross-study results provide reference performance under each system's resources and evaluation protocol.

Appendix F.1 Comparison Scope and Reporting

GTP-FA-$\pi_{0.5}$ and RAISE (GTP-FA-$\pi_{0.5}$) reuse the results in Table 1 in the main text: three seeds $\{42, 43, 44\}$, 200 episodes per task/model, and environment seeds `400005010--400005209`. The two methods share the grasp front-end, supervision, and adaptation settings, and report Stable Success@3. Published reference rows retain source-defined success rates. Task means and gains use equal task weights and rounding after aggregation of the reported task rates. Both tables use the same evaluated models on their respective shared-task subsets. Published systems retain their reported pretraining, demonstration budgets, controllers, and evaluation protocols. The matched GTP-FA-$\pi_{0.5}$–RAISE (GTP-FA-$\pi_{0.5}$) comparison measures RAISE's incremental gain.

Appendix F.2 Four-Task Comparison with the RDT Release

The RDT release uses 250 trials per method in ten groups of 25 [3]. Table F1 excludes PlugCharger from its five tasks and recomputes the shared-task means. RAISE (GTP-FA-$\pi_{0.5}$) improves on GTP-FA-$\pi_{0.5}$ by 13.78 percentage points, mainly through PegInsertionSide and PushCube. RAISE (GTP-FA-$\pi_{0.5}$) achieves stable-success rates of 37.5% on PickCube and 39.2% on PegInsertionSide. RDT reports a higher PickCube success rate under its own evaluation protocol.

Table F1 Four shared ManiSkill tasks from the RDT release. Peg/Pick/Stack/Push denote PegInsertionSide/PickCube/StackCube/PushCube. Published rows retain source metrics. GTP-FA-$\pi_{0.5}$ and RAISE (GTP-FA-$\pi_{0.5}$) report three-seed Stable Success@3 (200 episodes/task/seed). Avg. covers these four tasks; Δ compares RAISE (GTP-FA-$\pi_{0.5}$) with GTP-FA-$\pi_{0.5}$ in percentage points. Means and gains are rounded after aggregation.

Method	Peg	Pick	Stack	Push	Avg.
Published reference results: official RDT release					
RDT	13.2	77.2	74.0	100.0	66.10
OpenVLA	0.0	8.0	8.0	8.0	6.00
Octo	0.0	0.0	0.0	0.0	0.00
Diffusion Policy	0.0	40.0	80.0	88.0	52.00
Our evaluation: Stable Success@3, 3 seeds × 200 episodes/task					
GTP-FA-$\pi_{0.5}$	16.2	26.2	97.2	54.5	48.53
RAISE (GTP-FA-$\pi_{0.5}$)	39.2	37.5	97.2	75.3	62.30
Δ (pp)	+23.0	+11.3	0.0	+20.8	+13.78

Appendix F.3 Three-Task Comparison with PhysGen

Table F2 retains the rates and rounded averages from Song et al. [4]. PhysGen uses 1000 demonstrations and 125 rollouts per task; its comparison includes published and reproduced baselines. Across these three tasks, mean Stable Success@3 is 59.30% for GTP-FA-$\pi_{0.5}$ and 70.00% for RAISE (GTP-FA-$\pi_{0.5}$), a gain of 10.70 percentage points driven mainly by PushCube. This mean covers PushCube, PickCube, and StackCube; the four-task RDT comparison additionally includes PegInsertionSide.

Table F2 PhysGen's three-task reference rates and rounded means. GTP-FA-$\pi_{0.5}$ and RAISE (GTP-FA-$\pi_{0.5}$) reuse the three corresponding tasks from Table F1 and report three-seed Stable Success@3. Published methods retain their source-defined metrics and protocols. Δ compares RAISE (GTP-FA-$\pi_{0.5}$) with GTP-FA-$\pi_{0.5}$ in percentage points.

Method	PushCube	PickCube	StackCube	Avg.
Published reference results: PhysGen Table 2				
ACT	76	20	30	42
BC-T	98	4	14	39
Diffusion Policy	88	40	80	69
ICRT	77	78	30	62
RDT	100	77	74	84
π_0	100	60	48	69
PhysGen	100	73	48	74
Our evaluation: Stable Success@3, 3 seeds × 200 episodes/task				
GTP-FA-$\pi_{0.5}$	54.5	26.2	97.2	59.30
RAISE (GTP-FA-$\pi_{0.5}$)	75.3	37.5	97.2	70.00
Δ (pp)	+20.8	+11.3	0.0	+10.70

Appendix G Task Illustrations and Execution Examples

This appendix illustrates the task settings and recorded executions analyzed in Appendix E and Appendix F. It presents simulation scenes, real-robot RAISE (GTP-FA-$\pi_{0.5}$) sequences, and uniformly trained $\pi_{0.5}$ (G00 V0) executions, with the source and display convention specified for each group.

Appendix G.1 Simulation Task Overview

Figure G1 illustrates the ten task scenes and goals. Seven panels use task-animation frames; PegInsertionSide, PlaceApple, and TransportBox use expert-rollout base-camera observations.

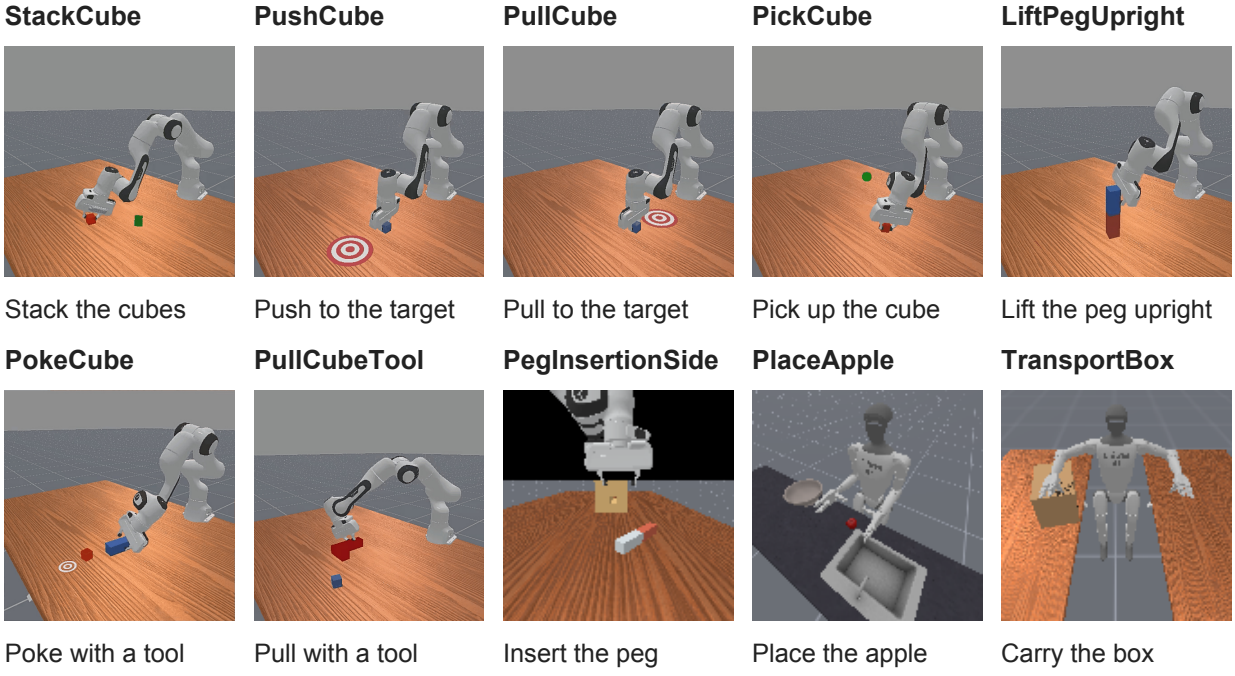

Figure G1 Ten simulation tasks and goals. Task animations supply seven panels; PegInsertionSide, PlaceApple, and TransportBox use expert base-camera observations.

Appendix G.2 Real-Robot Execution Sequences

Figures G2 and G3 show recorded executions of RAISE (GTP-FA-$\pi_{0.5}$), with the grasp front-end fixed and RAISE applied to downstream policy training. Each task presents four chronological moments from one video. Full views show scene geometry; fixed crops from the same frames enlarge object and gripper details for display. Timestamps indicate video-presentation times, and column headings describe the displayed moments. PourWater transfers small balls and uses a 90% completion criterion. The task-success rates are reported in Table 3 in the main text.

Appendix G.3 Recorded $\pi_{0.5}$ Success and Failure

Figure G4 shows two executions of uniformly trained $\pi_{0.5}$ (G00 V0), trained on 300 clean successes with seed 42 and evaluated at the final update 30,000 with a 250-step horizon. The upper example fails both success criteria; the lower satisfies both. Outcome labels are computed from the complete evaluation records. The examples use different initialization seeds and illustrate variation in G00 V0 execution outcomes.

Top: full frame. Bottom: same-frame detail. Time: video presentation.

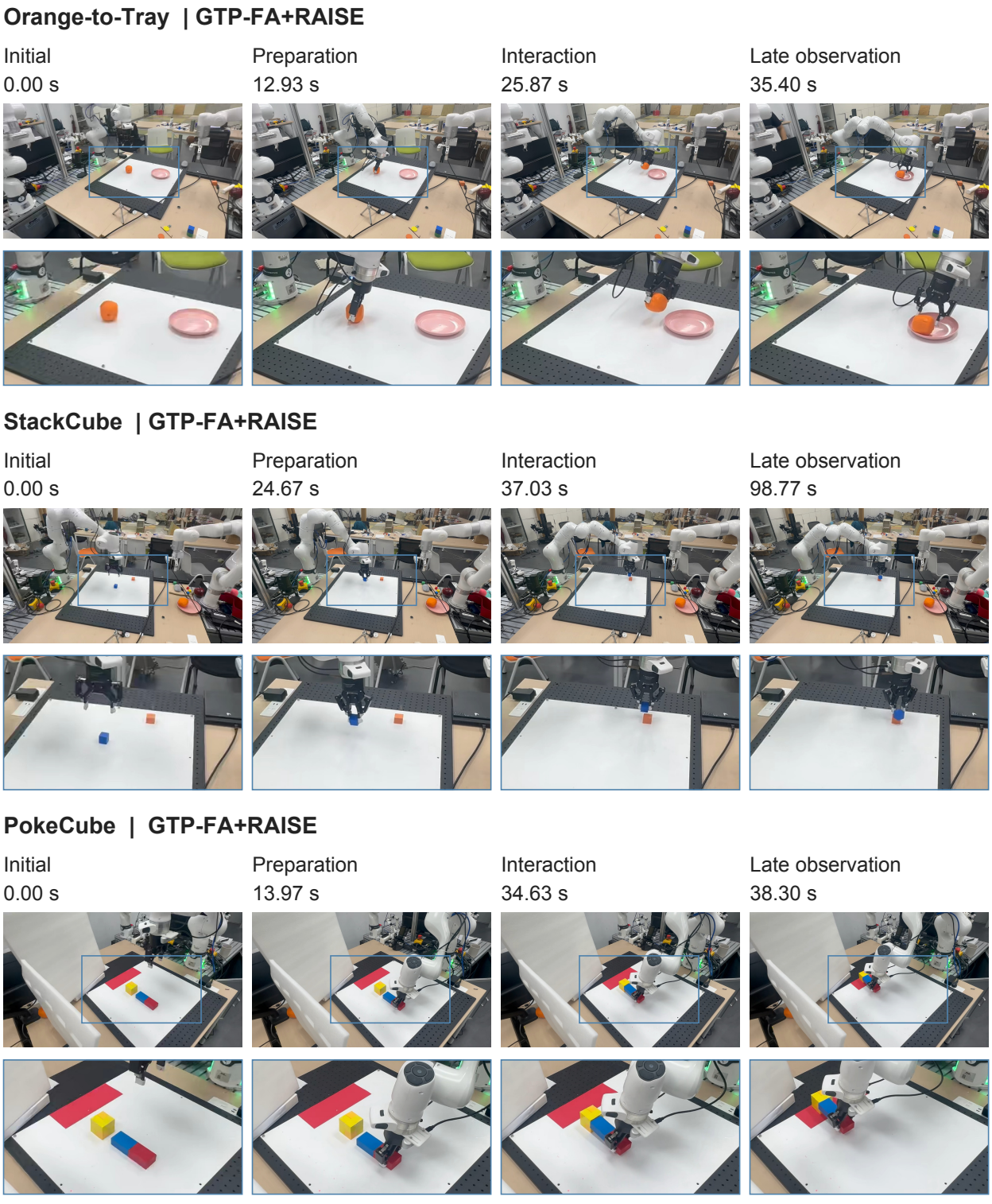

Figure G2 RAISE (GTP-FA-$\pi_{0.5}$) on Orange-to-Tray, StackCube, and PokeCube. Each task shows four chronological video moments: full view above, fixed same-frame detail below. Timestamps and column headings describe the displayed video moments.

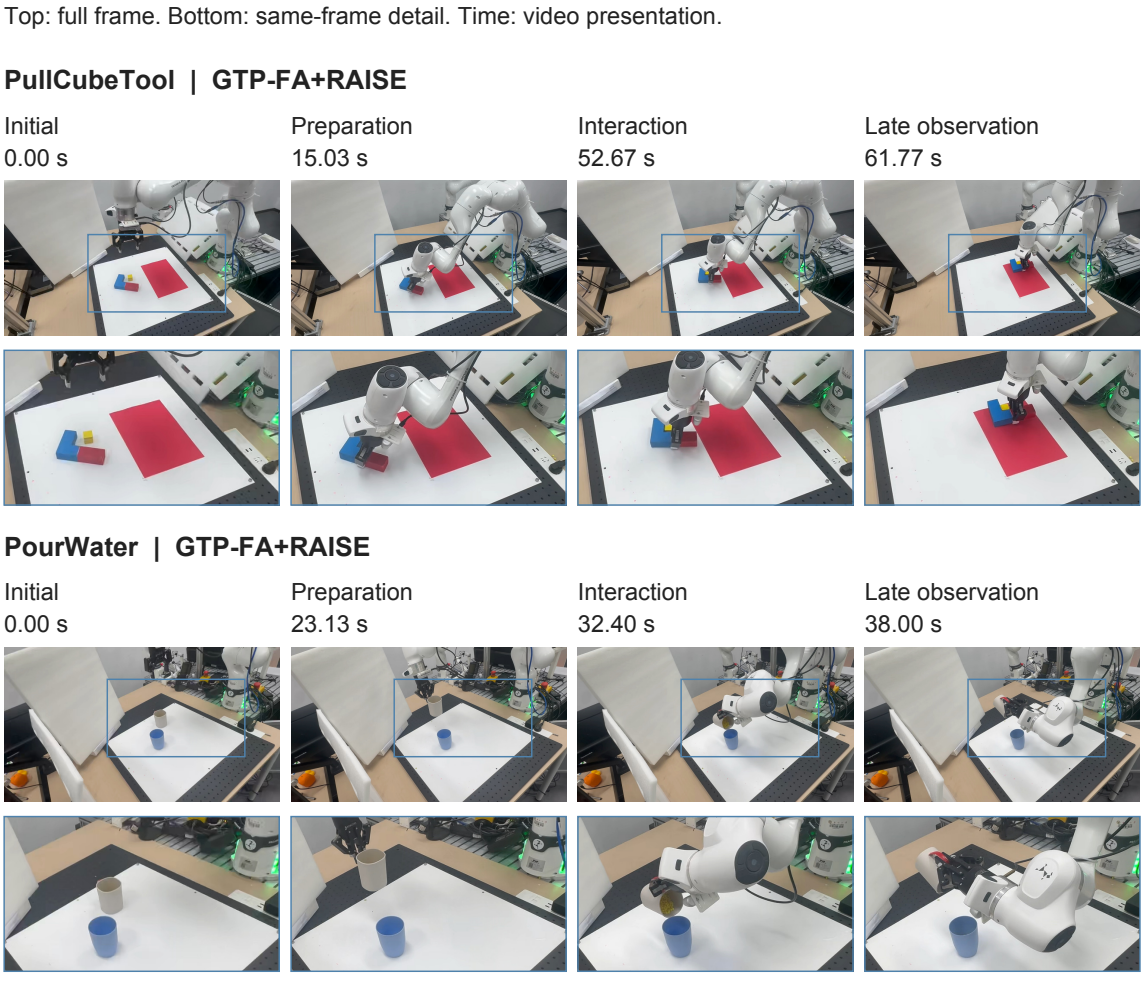

Figure G3 RAISE (GTP-FA-$\pi_{0.5}$) on PullCubeTool and PourWater (small-ball transfer). Views and timestamps follow Fig. G2.

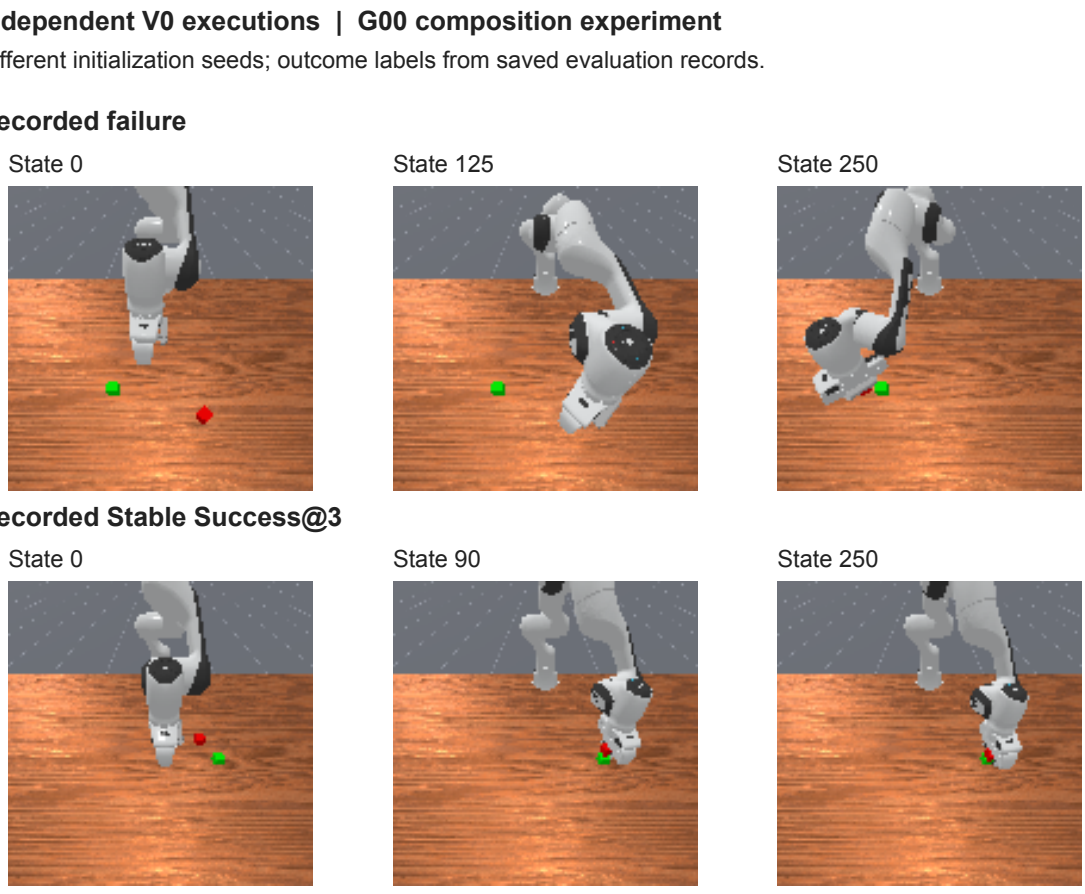

Figure G4 G00 V0 examples: failure (top) and stable success (bottom). Complete records determine outcomes; initializations differ.

Appendix H Controlled Policy Experiments

This appendix evaluates BC/DP data budgets, weighting and curation baselines, risk controls, parameter sensitivity, and resource scaling. It also reports real-robot data usage and evaluation uncertainty.

Appendix H.1 Training Configuration and Evaluation Horizons

The controlled comparisons use BC and DP on StackCube, PushCube, LiftPegUpright, PegInsertionSide, and PlaceApple, abbreviated as Stack, Push, Lift, Peg, and Apple. Their task horizons H are 250, 50, 200, 100, and 100 environment steps. Each configuration is trained independently with seeds $\{0, 1, 2\}$; each model is evaluated on the same 200 initial conditions. Thus, 600 episodes represent three trained models evaluated on 200 initial conditions. Within each task and demonstration scale, BC and BC+RAISE share policy inputs, architecture, action targets, normalization, initialization, update budget, batch order, and execution control; DP and DP+RAISE share these settings as well. BC uses Adam, learning rate 3×10^{-4}, and 10,000 updates. DP uses AdamW, learning rate 10^{-4}, 100,000 updates, 500 warm-up updates followed by cosine decay, and EMA with power 3/4. The batch size is 1,024; normalization statistics are computed from Expert300 and held fixed.

Task-specific configuration. BC+RAISE uses task-specific offline advancement windows and gradually introduces weights on Lift and Push. The PlaceApple matching thresholds for BC+RAISE and DP+RAISE are selected on an independent development set. BC and BC+RAISE share each task's training and execution configuration, as do DP and DP+RAISE. These controls include Push's demonstration-based execution controller. The offline scoring window, policy prediction horizon, and replanning interval have separate roles. Appendix A.5 defines offline windows and weight schedules; Section 4.7 in the main text describes applied weights and the shared Push controller; Appendix H.5 gives threshold selection.

Evaluation horizons. Table H1 evaluates the same BC, BC+RAISE, DP, and DP+RAISE models at task horizon H and at 250 steps. Each model executes for 250 steps, and its H-step results use the prefix of that trajectory with the same policy parameters, initial conditions, control, and success criteria. SuccessOnce (O) counts episodes with at least one task-success flag $s_t = 1$; Stable Success@3 (S) counts episodes in which the completion predicate c_t is true at three consecutive evaluation states, with $c_t \leqslant s_t$ (Appendix A.1). Both metrics accumulate success events up to the evaluation cutoff. BC and BC+RAISE use the same training configuration and evaluation horizon within each comparison, as do DP and DP+RAISE.

Table H1 Evaluation-horizon comparison with 300 action demonstrations. Entries are three-seed mean success-once (O) / Stable Success@3 (S), in percent. The H-step and 250-step results use the same models and trajectory prefixes. H is measured in environment steps.

Task	H	Uniform H: O / S	Uniform 250: O / S	RAISE H: O / S	RAISE 250: O / S
BC					
Stack	250	0.0 / 0.0	0.0 / 0.0	11.2 / 9.8	11.2 / 9.8
Push	50	3.5 / 2.3	4.8 / 3.3	32.8 / 31.8	35.8 / 34.5
Lift	200	61.5 / 61.2	64.0 / 63.8	73.2 / 73.0	74.8 / 74.7
Peg	100	19.2 / 10.3	23.0 / 15.5	46.8 / 43.2	52.0 / 49.3
Apple	100	20.7 / 19.8	24.3 / 23.5	43.2 / 38.0	47.5 / 43.3
DP					
Stack	250	39.7 / 39.2	39.7 / 39.2	44.3 / 43.8	44.3 / 43.8
Push	50	0.0 / 0.0	1.0 / 0.7	0.0 / 0.0	5.2 / 4.0
Lift	200	26.7 / 24.7	30.0 / 28.0	53.7 / 52.8	56.8 / 56.0
Peg	100	0.2 / 0.2	1.2 / 0.7	13.2 / 10.2	17.0 / 14.2
Apple	100	56.2 / 52.2	60.8 / 57.3	68.3 / 63.3	73.5 / 69.7

Appendix H.2 Data-Budget Controls

Additional clean action demonstrations. For each task, we separately collect 1,000 successful, retry-free trajectories using the same demonstrator, state distribution, and action interface as Expert300. They are trajectory-disjoint from Expert300 and all auxiliary pools. Both data scales use the same initialization, update budget, normalization, and evaluation settings. Push's demonstration-based execution library expands from 300 to 1,300 trajectories while its neighbor count and fusion rule remain unchanged; BC and BC+RAISE share the same library, as do DP and DP+RAISE. Auxiliary models, reference and calibration sets, and risk anchors remain frozen across these two action-data scales. Table H2 shows the resulting five-task comparison.

Table H2 Action-demonstration scale and weighting. Entries are three-seed mean Stable Success@3 (%) at task horizons H. Uniform rows report BC / DP; RAISE rows report BC+RAISE / DP+RAISE. The last column averages the five tasks equally.

Action trajectories	Weighting	Stack	Push	Lift	Peg	Apple	Mean
300	Uniform	0.0 / 39.2	2.3 / 0.0	61.2 / 24.7	10.3 / 0.2	19.8 / 52.2	18.7 / 23.2
300	RAISE	9.8 / 43.8	31.8 / 0.0	73.0 / 52.8	43.2 / 10.2	38.0 / 63.3	39.2 / 34.0
1,300	Uniform	2.2 / 41.5	6.7 / 3.0	62.7 / 23.8	14.7 / 2.3	20.3 / 54.3	21.3 / 25.0
1,300	RAISE	10.7 / 44.3	33.5 / 8.7	74.3 / 53.3	46.7 / 12.3	40.2 / 64.5	41.1 / 36.6

The expanded auxiliary-action supervision control is reported in Section 4.8 in the main text and Table 8 in the main text.

Appendix H.3 Matched Weighting and Data-Curation Baselines

We compare six training rules under the common BC or DP configuration, update budget, and task-horizon protocol in Appendix H.1. Each method starts from the same 300 candidate expert demonstrations; curation methods retain or add action trajectories according to their rules, as recorded in Table H3. BC and DP share each task's collection and selection results. Table 4 in the main text reports Stable Success@3. BC+RAISE and DP+RAISE have the highest five-task means, 39.2% and 34.0%, respectively. Among the comparison rules, Demo-SCORE gives the highest means: 34.8% with BC and 30.7% with DP. Policy configurations and evaluation settings are fixed within each policy family; each method uses the auxiliary inputs specified below.

AWR with progress-based advantages. We use the exponential advantage-weighting rule of AWR [5] with endpoint progress differences ΔV as advantage estimates. The AWR controls use the same frozen, task-adapted ROBOMETER-4B as BC+RAISE and DP+RAISE. With $z = (\Delta V - \text{median}(\Delta V))/(\text{IQR}(\Delta V) + 10^{-5})$, the weight is $\min(20, \exp(z))$; invalid scores receive unit weight. Temperature 1 and cap 20 are fixed. On Lift, the AWR-weighted BC policy uses unit weights for the first half of training and linearly introduces the AWR weights afterward.

DemInf. For DemInf [6], state and action β-VAEs are trained on nonterminal state–action pairs from the 300 expert trajectories, using the official state-input configuration. KSG scoring uses $k \in \{5, 6, 7\}$. The fixed 75% retention ratio keeps 225 trajectories; ties are resolved by trajectory identifier. Push's execution-neighbor library is built from the retained trajectories.

Demo-SCORE. For Demo-SCORE [7], each task has an independently trained uniform BC policy using the 300 expert demonstrations. It collects 100 non-test trajectories, each of 250 steps, at each of its 2,500, 5,000, 7,500, and 10,000-update checkpoints. The first three groups each train a candidate classifier; the fourth group selects the classifier and epoch by unweighted validation cross-entropy. Training and validation groups share no trajectories. The threshold γ is the mean training-trajectory score of the selected classifier. Expert trajectories scoring above γ are retained; added online trajectories must also finish successfully. BC and DP reuse the same 400 collected trajectories and selection per task (2,000 collected trajectories across the five tasks).

ReTVL with retry-boundary adaptation. For the retry-boundary adaptation of ReTVL [8], a value model is independently adapted from pretrained Robometer-4B for 500 updates per task. With $r = V(\text{after}) - V(\text{before})$, the weight is $\alpha = \text{clip}((r - \mu + 2\sigma)/(4\sigma + 10^{-8}), 0, 1)$, where μ and σ are the mean and population standard deviation of value increments on that task's expert training samples. Samples whose value increment exceeds the 80th percentile of positive increments receive weight one, and the policy loss is $\sum_i \alpha_i \ell_i / (\sum_i \alpha_i + 10^{-8})$. The ReTVL weighting rule is retained, while physical retry starts or decreases in the geometric criterion replace human correction starts. Terminal success or failure supplies value supervision. Push uses 300 expert trajectories plus 300 online trajectories. The other four tasks use existing retry-training trajectories; complete-trajectory and event counts are listed separately in Table H3. On Lift, ReTVL uses 884 ultimately successful and 36 ultimately failed trajectories, totaling 920 trajectories and 1,000 retry events. RAISE's separately constructed Lift retry set contains 920 ultimately successful trajectories and 1,000 events.

Table H3 Task-wise inputs for the curation and value-weighting baselines. Counts are shared across BC, DP, and policy seeds. Demo-SCORE starts from 300 candidate expert and 400 collected online trajectories per task; The ReTVL column lists value-model training trajectories and retry events.

Task	DemInf retained expert	Demo-SCORE retained expert	Demo-SCORE added online success	$\gamma \times 10^4$	ReTVL value-model trajectories / retry events
Stack	225	150	14	1,315	729 / 1,000
Push	225	242	110	4,144	600 / 18
Lift	225	300	208	2,637	920 / 1,000
Peg	225	299	142	4,079	669 / 1,000
Apple	225	298	198	6,196	922 / 995

Appendix H.4 Risk Controls and Training-Seed Variation

Table 5(a) in the main text reports the BC/DP component controls. For each seed, we average the task-wise gains equally across five tasks, then report the mean and sample standard deviation across the three seed-level averages. DP with progress-only weighting exceeds uniformly trained DP by 3.3 ± 1.7 percentage points. DP+RAISE exceeds the progress-only DP control by 7.5 ± 3.2 points, giving a total 10.8-point gain over DP. BC+RAISE and DP+RAISE each exceed their progress-only control in five-task mean success for every training seed. These seed-level statistics describe panel (a).

On Lift, the training-seed sample standard deviation decreases from 30.17 pp for BC to 5.89 pp for BC+RAISE. On Stack, it increases from 3.06 pp for DP to 9.46 pp for DP+RAISE. The direction of the variability change therefore depends on the task and policy.

Appendix H.5 Parameter Selection and Sensitivity

The default matching limits $\delta_z = \delta_u = 0.15$, advancement saturation $\delta_V = 0.12$, and strength $\beta = 1$ were set empirically from normalized representation-distance, within-stage-progress, and positive-advancement scales before policy training. Table H7 varies one parameter at a time on StackCube. Changing δ_u to 0.10 or 0.20 gives BC+RAISE stable success of 10.17% or 6.50%, compared with 9.83% at the

Table H4 Task-horizon Stable Success@3 (%): mean ± sample standard deviation over three training seeds. Variability is computed across the three trained policies, each evaluated on the same 200 initial conditions. Table 1 in the main text uses the common 250-step horizon.

Task	BC	BC+RAISE	DP	DP+RAISE
Stack	0.00 ± 0.00	9.83 ± 1.44	39.17 ± 3.06	43.83 ± 9.46
Push	2.33 ± 2.31	31.83 ± 1.26	0.00 ± 0.00	0.00 ± 0.00
Lift	61.17 ± 30.17	73.00 ± 5.89	24.67 ± 6.11	52.83 ± 5.80
Peg	10.33 ± 2.36	43.17 ± 5.39	0.17 ± 0.29	10.17 ± 2.47
Apple	19.83 ± 4.19	38.00 ± 4.92	52.17 ± 5.03	63.33 ± 4.73

Table H5 Stable-success counts for the seven key controls. Each entry lists training seeds 0/1/2, with 200 evaluation episodes per seed and task. All rows use the task horizons in Appendix H.1. The 300/1300 suffix denotes action-demonstration count; Demo-SCORE uses its selected expert and online-success trajectories. Repeated results elsewhere reuse these models.

Learner and control	Stack	Push	Lift	Peg	Apple
BC Uniform, 300	0/0/0	2/10/2	53/163/151	19/17/26	44/45/30
BC+RAISE, 300	18/18/23	64/61/66	133/149/156	94/74/91	79/84/65
BC Progress only, 300	8/7/6	29/33/31	133/128/123	37/40/44	54/54/55
BC Global shuffle, 300	8/4/1	25/21/29	128/122/125	31/30/30	42/41/44
BC Uniform, 1300	1/8/4	12/15/13	124/127/125	29/33/26	43/41/38
BC+RAISE, 1300	21/17/26	67/69/65	144/149/153	93/97/90	80/82/79
BC Demo-SCORE	15/13/18	76/85/81	139/140/141	49/51/48	66/62/59
DP Uniform, 300	85/77/73	0/0/0	36/60/52	1/0/0	95/115/103
DP+RAISE, 300	96/101/66	0/0/0	100/119/98	18/26/17	116/134/130
DP Progress only, 300	85/80/82	0/0/0	57/65/61	9/11/8	116/113/110
DP Global shuffle, 300	73/81/77	0/0/0	56/47/51	4/6/3	106/108/105
DP Uniform, 1300	85/83/81	7/6/5	44/48/51	5/8/1	109/109/108
DP+RAISE, 1300	89/89/88	18/17/17	105/107/108	25/25/24	134/124/129
DP Demo-SCORE	89/80/85	4/2/1	95/94/96	16/18/21	110/107/104

Table H6 Paired five-task mean gains of BC+RAISE and DP+RAISE over the listed BC and DP controls in Stable Success@3 (percentage points). For each training seed, we first take within-task differences and then average the five tasks. The last column gives the mean and sample standard deviation of the three seed-level gains. Uniform comparisons match action-data size; progress and shuffle controls use 300 demonstrations. Demo-SCORE has the distinct data usage reported in Appendix H.3.

Base	Control	Gains for seeds 0/1/2	Mean ± SD
BC	Uniform, 300	27.0 / 15.1 / 19.2	20.4 ± 6.0
BC	Progress only, 300	12.7 / 12.4 / 14.2	13.1 ± 1.0
BC	Global shuffle, 300	15.4 / 16.8 / 17.2	16.5 ± 0.9
BC	Demo-SCORE	4.3 / 3.5 / 5.4	4.4 ± 1.0
BC	Uniform, 1300	19.6 / 19.0 / 20.7	19.8 ± 0.9
DP	Uniform, 300	11.3 / 12.8 / 8.3	10.8 ± 2.3
DP	Progress only, 300	6.3 / 11.1 / 5.0	7.5 ± 3.2
DP	Global shuffle, 300	9.1 / 13.8 / 7.5	10.1 ± 3.3
DP	Demo-SCORE	1.6 / 7.9 / 0.4	3.3 ± 4.0
DP	Uniform, 1300	12.1 / 10.8 / 12.0	11.6 ± 0.7

default setting. Increasing δ_V to 0.16 gives DP+RAISE 40.00%, compared with 43.83% at the default setting. The table reports means and sample standard deviations across training seeds.

Table H7 StackCube one-at-a-time parameter sensitivity: Stable Success@3 (%), mean ± sample standard deviation across three training seeds, 200 episodes per seed, and a 250-step horizon. Defaults are $\delta_z = \delta_u = 0.15$, $\delta_V = 0.12$, and $\beta = 1$; each other row changes only the named parameter.

Setting	BC+RAISE	DP+RAISE
Default	9.83 ± 1.44	43.83 ± 9.46
$\delta_z = 0.10$	8.17 ± 1.04	43.50 ± 7.57
$\delta_z = 0.20$	8.17 ± 3.01	43.00 ± 10.69
$\delta_u = 0.10$	10.17 ± 1.04	43.17 ± 8.37
$\delta_u = 0.20$	6.50 ± 1.32	43.00 ± 10.58
$\delta_V = 0.08$	9.50 ± 1.00	43.50 ± 10.90
$\delta_V = 0.16$	9.00 ± 3.28	40.00 ± 8.53
$\beta = 0.5$	7.83 ± 1.04	43.50 ± 11.30
$\beta = 1.5$	7.33 ± 0.58	43.50 ± 9.26

For the PlaceApple BC/DP experiments, a trajectory-disjoint development set selects $\delta_z = \delta_u = 0.30$; δ_V and β remain unchanged. This set contains 200 clean successes and 187 retry successes with 200 retry events, and excludes action training, progress adaptation, reference, calibration, and final-evaluation trajectories. It supplies 200 positive and 200 negative pairs, each drawn from distinct trajectories in the same task and stage. Positive pairs represent the same local error context; negatives represent different error contexts or normal advancement. Labels are judged from the surrounding action clips independently of the matching scores.

A pair is accepted when cosine distance $d \leqslant \delta_z$ and $|u_i - u_j| \leqslant \delta_u$. With FN = 200−TP, F1 is 2TP/(2TP+FP+FN). Development-set matching F1 selects the thresholds, which are fixed before the downstream policy comparisons. Table H8 reports all four candidates.

Table H8 PlaceApple development-set matching. All candidates use the same 200 positive and 200 negative state pairs. TP and FP count accepted positive and negative pairs. Precision, recall, and F1 are percentages. The 0.30/0.30 candidate has the highest F1 among the four candidates.

δ_z/δ_u	TP	FP	Precision	Recall	F1
0.15 / 0.15	122	7	94.57	61.00	74.16
0.20 / 0.20	146	12	92.41	73.00	81.56
0.30 / 0.30	166	16	91.21	83.00	86.91
0.40 / 0.40	176	34	83.81	88.00	85.85

Appendix H.6 Real-Robot Data Reuse and Evaluation Uncertainty

The five tasks use 1500 successful demonstrations in total: 1185 without retries and 315 with retries, containing 462 retry events (Table H9). Every task uses all 300 demonstrations for action supervision and reuses them in three trajectory-disjoint auxiliary roles: 200 for progress adaptation, 50 for successful references, and 50 for calibration. The split is stratified by retry presence and covers task stages. The three auxiliary subsets are disjoint from one another, but all belong to the action-training set. Progress models use base-camera RGB; annotation uses base and wrist views. Retry-risk anchors come from the 200 progress-adaptation demonstrations. No separate terminal-failure trajectories are collected or used in these physical experiments, so the active risk source is $\mathcal{U} = \{R\}$. The 462 retry events are counted across the full 1500-trajectory demonstration collection. Risk anchors use the task-specific 200-trajectory adaptation subsets. Table D2 reports annotation agreement on a 300-trajectory subset of this physical demonstration collection.

Annotation effort estimate. Each of two annotators independently labels all 300 demonstrations per task. At an estimated throughput of three trajectories per minute per annotator, this gives approximately 100 minutes per annotator per task. The two annotation passes therefore require an estimated 200 person-minutes (3.3 person-hours) per task, or 1000 person-minutes (16.7 person-hours) across the five tasks. This estimate is calculated from annotation throughput and covers the two annotation passes. Demonstration collection, model training, robot evaluation, setup, and reconciliation fall outside this estimate; setup and reconciliation effort remain unquantified. The same 1500 action demonstrations provide action targets and auxiliary supervision.

Table H9 Real-robot action demonstrations and retry events. All 300 trajectories per task end successfully. An event is not an additional trajectory; a trajectory can contain multiple retries.

Task	Total	Clean	With retries	Retry events
Orange-to-Tray	300	260	40	56
StackCube	300	236	64	92
PokeCube	300	251	49	68
PullCubeTool	300	228	72	108
PourWater	300	210	90	138

Real-robot evaluation. Table 3 in the main text compares $\pi_{0.5}$, RAISE ($\pi_{0.5}$), GTP-FA-$\pi_{0.5}$, and RAISE (GTP-FA-$\pi_{0.5}$) under a common protocol, totaling 1000 physical executions. Each method independently initializes 50 scenes per task within the same placement ranges and task rules. Each named policy and its RAISE version share demonstrations, cameras, calibration, controller, action interface, and success predicate; the GTP-FA front-end is fixed. Across five tasks, $\pi_{0.5}$ and RAISE ($\pi_{0.5}$) succeed in 30/250 and 67/250 executions, respectively. GTP-FA-$\pi_{0.5}$ and RAISE (GTP-FA-$\pi_{0.5}$) succeed in 188/250 and 212/250 executions. Task success is judged at the end of execution. Table H10 reports Newcombe 95% intervals for independent binomial differences [9], computed from Wilson score intervals without continuity correction [10].

Table H10 Real-robot success-rate gains (percentage points): RAISE ($\pi_{0.5}$) over $\pi_{0.5}$, and RAISE (GTP-FA-$\pi_{0.5}$) over GTP-FA-$\pi_{0.5}$. Each method/task has 50 independent trials. Pointwise 95% confidence intervals use the Newcombe–Wilson procedure described above, without multiplicity correction.

Task	$\pi_{0.5}$ gain	95% interval	GTP-FA gain	95% interval
Orange-to-Tray	+14.0	[-1.6, 29.0]	+4.0	[-7.7, 16.0]
StackCube	+16.0	[2.2, 29.8]	+12.0	[-4.4, 27.7]
PokeCube	+12.0	[-5.6, 28.6]	+6.0	[-7.7, 19.7]
PullCubeTool	+18.0	[1.0, 33.8]	+14.0	[-1.6, 29.0]
PourWater	+14.0	[1.5, 27.1]	+12.0	[-6.9, 29.7]

The five-task mean gains are 14.8 pp for RAISE ($\pi_{0.5}$) over $\pi_{0.5}$ and 9.6 pp for RAISE (GTP-FA-$\pi_{0.5}$) over GTP-FA-$\pi_{0.5}$. All ten task-level point estimates are positive. For RAISE ($\pi_{0.5}$), the intervals on StackCube, PullCubeTool, and PourWater lie entirely above zero; the other seven intervals include zero. These task-wise intervals use 50 trials per method and task.

Appendix H.7 Auxiliary Cost and Resource Scaling

Table H11 reports machine wall-clock time for the six StackCube auxiliary-preparation stages on one RTX 4090 (24GB) and the host CPU. The successful auxiliary pools contain 1000 clean trajectories, 729 retry trajectories with 1000 events, and 1000 reference and 1000 calibration trajectories. The failure pool contains 489 trajectories, of which 352 produce anchors. Progress adaptation uses 1000 updates. BC+RAISE and DP+RAISE share the auxiliary data, adapted model, features, references, calibration, and anchors across their three training seeds. Each method exports weights for its own advancement window. The six stages total 1410 minutes (23.5 hours) for StackCube, from auxiliary-data collection through weight export. Downstream policy training and evaluation are outside this timing window. BC+RAISE and DP+RAISE reuse the shared auxiliary assets across training seeds. Progress inference and risk retrieval run offline, leaving policy deployment unchanged.

Reuse and cost accounting. For a fixed task with unchanged observations, stage definitions, and auxiliary-data protocol, BC+RAISE, DP+RAISE, RAISE ($\pi_{0.5}$), and RAISE (GTP-FA-$\pi_{0.5}$) can reuse the adapted progress model and task data while preserving the prescribed data-role separations. Each policy requires weights computed for its own advancement window, followed by its own training and evaluation. Reuse spreads the task-level preparation cost across these policies and training seeds. Training each policy is accounted for separately from RAISE-specific preparation. Table H11 quantifies one StackCube auxiliary-preparation workflow.

Separate one-time annotation validation. The one-time, 300-trajectory StackCube validation study in Appendix D.3 required approximately 13 person-hours: approximately five hours for each of two independent annotators and three hours for a third annotator's adjudication. This human-effort total covers the annotation-validation study and is reported separately from the machine-preparation profile. The other nine simulation tasks use trajectory labels generated automatically from recorded physical states. Routine setup, task-predicate implementation, and execution supervision remain unquantified. Appendix H.6 reports the real-robot annotation estimate.

Table H11 StackCube auxiliary machine-preparation time on one RTX 4090 (24GB) and the host CPU. The pools contain 3729 successful and 489 failed trajectories, or 4218 unique trajectories. The shared 300 action demonstrations, development data, separately added 1000 demonstrations, and downstream policy training and evaluation are outside the measured workflow. Machine minutes report sequential wall-clock time. The one-time manual validation of 300 StackCube trajectories used approximately 13 person-hours, accounted for separately.

Stage	Machine minutes
Collection and quality filtering	180
Automatic simulation-label generation	30
Progress-model adaptation	1,140
Features, references, calibration	30
Failure localization and anchors	15
Weight generation and export	15
Total	1,410

Smaller auxiliary pools. For each configuration in Table H12, we readapt the progress model, rebuild references and calibration, extract anchors, and train BC+RAISE and DP+RAISE with three seeds each. Action demonstrations remain fixed. From A to C, Stable Success@3 increases from 2.8% to 6.2% for BC+RAISE and from 40.2% to 42.7% for DP+RAISE; the full pools give 9.8% and 43.8%, respectively. All auxiliary roles change jointly across these resource settings.

Table H12 StackCube auxiliary-data scale with 300 fixed action demonstrations. Retry counts are trajectories/events; reference/calibration counts are trajectories. Each role is trajectory-disjoint from the others and nested across scales. Success is three-seed mean Stable Success@3 (%), 200 episodes per seed and 250 steps. All auxiliary roles change jointly.

Set	Clean	Retry traj./events	Reference/ calibration	Failures	Unique	BC+RAISE / DP+RAISE
A	100	84/114	100/100	12	396	2.8 / 40.2
B	250	194/265	250/250	31	975	4.7 / 41.3
C	500	376/516	500/500	63	1,939	6.2 / 42.7
Full	1,000	729/1,000	1,000/1,000	489	4,218	9.8 / 43.8

Benchmark workload. The seven key controls use 105 independent runs per policy family; AWR with progress-based advantages, DemInf, and ReTVL add 45; the two constrained-shuffle controls add 30; five component ablations add 75; eight Stack parameter variants add 24; three reduced-resource settings add 9; and the joint-model downstream comparison adds 15. This totals 303 BC-based and 303 DP-based runs, or 121,200 task-horizon evaluation episodes. Using RTX 4090 timing references of 0.5 hours per complete BC run and 4 hours per DP run gives an estimated 1363.5 equivalent task-hours for the BC/DP control suite. The $\pi_{0.5}$ component ablations in Table 5(b) in the main text are accounted for separately.

The measured StackCube auxiliary machine time is reported separately in Table H11; the one-time annotation-validation effort is a separate research cost.

The 12 all-auxiliary training paths and 36 checkpoint evaluations in Appendix H.2 are counted separately. The common 250-step evaluation reuses 51 trained models for 10,200 episodes; Stack reuses its 250-step records.

Appendix H.8 Joint Progress Adaptation Across Ten Tasks

We compare ten individual progress models, each adapted for 1000 updates, with one joint model trained for 10,000 task-balanced updates. Each task retains its own successful references and risk calibration. Downstream evaluation trains BC+RAISE and DP+RAISE on the five policy-evaluation tasks with three training seeds each. Both settings use 10,000 adaptation updates in total. The individual PushCube model uses LoRA rank 32; the joint model uses rank 64.

Table H13 Downstream Stable Success@3 (%) with individual or joint progress adaptation. Each cell is BC+RAISE / DP+RAISE, averaged over three policy seeds at the task horizons of Appendix H.1.

Progress adaptation	Stack	Push	Lift	Peg	Apple	Avg.
Individual	9.8 / 43.8	31.8 / 0.0	73.0 / 52.8	43.2 / 10.2	38.0 / 63.3	39.2 / 34.0
Joint, ten tasks	5.5 / 40.5	20.3 / 0.0	65.2 / 34.5	23.3 / 4.2	24.7 / 56.2	27.8 / 27.1

Individual progress adaptation gives five-task means of 39.2% for BC+RAISE and 34.0% for DP+RAISE. Joint adaptation gives 27.8% and 27.1%, respectively. Individual adaptation gives higher success on all five BC+RAISE tasks and four DP+RAISE tasks. Both DP+RAISE configurations have zero success on PushCube at the 50-step horizon. Joint adaptation stores one progress model for the ten training tasks; individual adaptation stores ten. Downstream evaluation covers five tasks included in the adaptation corpus, with the total number of adaptation updates matched between settings. The main method uses task-specific progress adaptation.

References

1 Kim M J, Pertsch K, Karamcheti S, et al. Openvla: An open-source vision-language-action model. arXiv preprint arXiv:2406.09246, 2024
2 Xu J, Wang P, Zhang H, et al. Grasp-then-plan with failure attribution: A closed two-stage framework for precise and generalizable robotic manipulation. arXiv preprint arXiv:2606.03385, 2026
3 Liu S, Wu L, Li B, et al. Rdt-1b: a diffusion foundation model for bimanual manipulation. arXiv preprint arXiv:2410.07864, 2024
4 Song Z, Li Q, Qin S, et al. Learning physics from pretrained video models: A multimodal continuous and sequential world interaction models for robotic manipulation. In: Proceedings of the 2026 International Conference on Multimedia Retrieval, 2026. 758–767
5 Peng X B, Kumar A, Zhang G, et al. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. arXiv preprint arXiv:1910.00177, 2019. URL https://arxiv.org/abs/1910.00177
6 Hejna J, Mirchandani S, Balakrishna A, et al. Robot data curation with mutual information estimators. In: Proceedings of Robotics: Science and Systems, Los Angeles, CA, USA, 2025
7 Chen A S, Lessing A M, Liu Y, et al. Curating demonstrations using online experience. In: Proceedings of Robotics: Science and Systems, Los Angeles, CA, USA, 2025
8 Qin X, Lu J, Wang K, et al. Beyond monotonic progress: Retry-supervised value learning for robot imitation. arXiv preprint arXiv:2606.24633, 2026. URL https://arxiv.org/abs/2606.24633
9 Newcombe R G. Interval estimation for the difference between independent proportions: Comparison of eleven methods. Statistics in Medicine, 1998, 17: 873–890
10 Wilson E B. Probable inference, the law of succession, and statistical inference. Journal of the American Statistical Association, 1927, 22: 209–212