COMMITFLOW: EXECUTION-ALIGNED SAMPLING FOR RECEDING-HORIZON ROBOT POLICIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Flow-based robot policies jointly generate an action chunk, but receding-horizon controllers execute only a prefix before replanning and discard the remaining actions. With bidirectional attention across action tokens, noise assigned to this discarded tail can nevertheless influence the actions sent to the robot. We call this dependence tail-noise leakage and introduce tail-resampling sensitivity (TRS) to measure how the executed prefix changes when only tail-local noise is resampled. To remove this dependence, we propose **CommitFlow**, which aligns the sampler's action-token dependencies with the controller's execution boundary. Its strict formulation isolates every prefix from subsequent position-local noise, while CommitFlow-K preserves bidirectional coordination within a known execution block and prevents influence from the discarded tail. A shared stochastic context supports chunk-level coordination without reopening tail-to-prefix paths. We prove counterfactual prefix invariance under both dependency designs while retaining parallel action sampling. Controlled interventions demonstrate that tail-noise leakage can affect closed-loop outcomes: resampling the full discarded tail produces both success and failure for the baseline in 6 of 40 fixed states, whereas CommitFlow remains invariant. Beyond this mechanism study, the full model improves aggregate success on LIBERO (+1.18 pp), RoboTwin (+3.92 pp), and AgiBot G1 tasks (+2.57 pp), while reducing across-replan inconsistency. These results show that execution-aligned sampling can eliminate a behaviorally consequential dependence on discarded-tail randomness while maintaining or improving aggregate task performance.

1 INTRODUCTION

Action chunking allows modern robot policies to generate a coordinated sequence of controls from the current observation (Zhao et al., 2023; Chi et al., 2025; Black et al., 2025a; Bjorck et al., 2025; Shukor et al., 2025; Zheng et al., 2026; Wu et al., 2026). However, during receding-horizon execution, many policies use only part of that sequence: they execute the first K actions of an H-step chunk, receive a new observation, then replan (Chi et al., 2025; Black et al., 2025b;a). The remaining $H - K$ actions (i.e., the discarded tail) are replaced before execution. This creates a distinction between what the model generates and what the robot actually uses. Crucially, discarding an output does not necessarily remove its influence on the executed prefix. In those flow-based policies with unrestricted attention, action positions interact bidirectionally throughout iterative sampling. Noise assigned to a discarded position can therefore propagate into the controls that the robot will finally execute. Consider two sampling inputs with the same observation and identical noise at the first K positions, but differing only in the tail. Although neither sampled tail reaches the robot, these two inputs can produce different executed prefixes. We call this dependence *tail-noise leakage*. It arises within a single sampling call: the policy's execution boundary determines which outputs are used, but does not by itself restrict which stochastic inputs can affect them.

To study this dependence, we ask: **How much can the executed prefix change when only the local noise assigned to the discarded tail is resampled?** We introduce tail-resampling sensitivity (TRS), which measures the resulting prefix change while holding the observation, prefix-local noise, and any shared stochastic context fixed. This intervention isolates discarded-tail influence from the randomness used to generate the committed actions. We also test its consequences in closed-loop execution. On 40 prespecified LIBERO (Liu et al., 2023) states, changing only the first call's tail-local

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

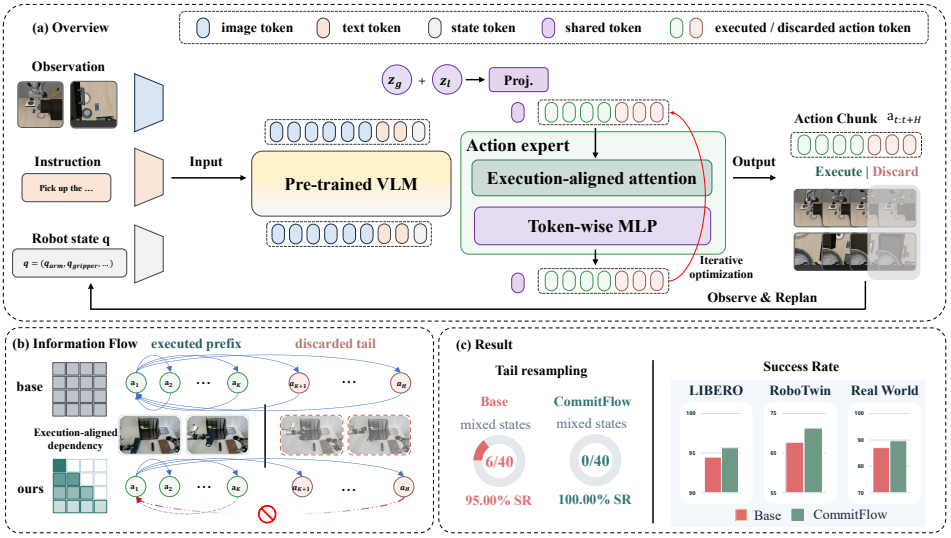

Figure 1: **Overview of CommitFlow.** (a) A VLM-conditioned action expert generates action chunks, with only the prefix executed before replanning. (b) Prefix-closed attention blocks tail-to-prefix influence, while shared stochastic context supports chunk-level coordination. (c) CommitFlow preserves outcomes under controlled tail resampling and achieves higher aggregate success in separate LIBERO, RoboTwin, and real-world evaluations.

noise, with all later-call randomness fixed, produces both successful and failed rollouts in six states for the unrestricted baseline. Thus, discarded-tail dependence can affect task outcomes, rather than merely changing an action-space diagnostic.

These observations motivate **CommitFlow**, an execution-aligned sampling framework that constrains action-token dependencies at the controller's commitment boundary. Its strict, K-agnostic form allows each action token to read only current and earlier action states, making every prefix independent of subsequent position-local noise. When K is known, **CommitFlow-K** instead isolates the committed block as a whole: its actions retain bidirectional interaction, while the discarded tail cannot influence them. Conditioning tokens remain one-way sources in both designs, preventing indirect tail-to-prefix paths. Under the corresponding closure and uniqueness assumptions, we prove counterfactual prefix invariance: changing only tail-local noise leaves the protected prefix unchanged. However, strict closure removes later-to-earlier communication even among actions that will be executed together. To support chunk-level coordination under this restriction, we combine position-local noise with a shared stochastic context. The shared latent contributes to every position's initial noise through a shared/local construction related to correlated noise priors for video diffusion (Ge et al., 2023). It is also provided as a conditioning token that all action positions can read, but that cannot read action states. This supplies a common source of variation without reopening tail-specific dependency paths.

We evaluate on both simulation and real-world tasks. The full model improves aggregate success by 1.18 pp on LIBERO (Liu et al., 2023), 2.78–3.92 pp across the tested execution lengths on RoboTwin (Chen et al., 2025), and 2.57 pp on AgiBot G1, while reducing across-replan inconsistency. Factorial and known-boundary studies further distinguish the roles of isolation and coordination: prefix closure removes tail sensitivity, while shared context or within-block interaction helps recover task performance under that constraint. Together, these results support designing the sampler's dependencies around the portion of its output that the controller commits to execute.

Our contributions could be summarized in three parts:

- We identify how randomness assigned to discarded action positions can alter the executed prefix of receding-horizon flow policies. We introduce TRS to quantify this dependence and demonstrate through controlled interventions that it can change closed-loop task outcomes.
- We propose CommitFlow to protect arbitrary prefixes or a known execution block from tail-local randomness. We prove counterfactual prefix invariance and incorporate shared stochastic context to support coordination without reopening tail-to-prefix paths.

- Our model improves aggregate success on LIBERO, RoboTwin, and AgiBot G1, while reducing across-replan inconsistency. Factorial and known-boundary ablations clarify the complementary roles of tail isolation and action coordination.

2 Related Work

Information flow in generative sequences. The transformer variant of Diffusion Policy (Chi et al., 2025) uses causal attention over action positions. CDP (Ma et al., 2025) conditions action generation on historical actions while retaining bidirectional interaction among the current denoising targets. Diffusion Forcing (Chen et al., 2024) trains causal sequence models with independently sampled noise levels across positions, enabling flexible sequence generation and planning. CommitFlow likewise uses a causal ordering over action positions. Our formulation derives the required invariance from the controller's execution boundary and measures its violation by resampling discarded-tail local noise. For a known boundary, the contract permits bidirectional interaction within the entire committed block while preventing influence from the discarded tail. This distinction separates the execution requirement from the stronger restriction of causal ordering at every action position.

Continuity and asynchronous execution. BID (Liu et al., 2025) selects action chunks using backward coherence with previous predictions and forward contrast over future plans, balancing temporal consistency and reactivity. RTC (Black et al., 2025c) guides new action chunks to remain consistent with actions committed during asynchronous inference. Training-time action conditioning (Black et al., 2025d) instead simulates inference delays during training and directly conditions on committed action prefixes, avoiding inference-time inpainting. SEAM (Zhan et al., 2026) uses the previous chunk's unexecuted tail as a reference for smoother transitions. PAINT (Ho et al., 2026) selects initial noise to produce a continuation consistent with the executed prefix. REMAC (Wang et al., 2026b) combines masked action-chunk training with prefix-preserved sampling to mitigate intra-chunk perception–action mismatch and improve inter-chunk continuity during asynchronous inference. Streaming Flow Policy (Jiang et al., 2025) treats the action trajectory itself as a flow trajectory and streams actions during its integration, while FlashVLA (Li et al., 2026) uses chunk-wise causal attention to decode a streaming buffer of action chunks. These methods relate generation to ongoing execution through sample selection, trajectory reuse, conditioning, or streaming computation. Our focus is dependence inside a single sampled chunk: local noise assigned only to discarded positions can influence the prefix exported for execution. We define and test invariance to that noise with the conditioning and all other stochastic inputs fixed.

Adaptive execution horizons. AutoHorizon (Wang et al., 2026a) dynamically estimates each chunk's execution horizon from action self-attention patterns. A3 (Chen et al., 2026) uses trajectory consensus across sampled rollouts and dual verification to accept the longest verified action prefix. DVAC (Feng et al., 2026) uses the variance of clean-action estimates across the final denoising steps to identify a stable executable prefix. DEHP (Zhao et al., 2026) learns an execution-horizon predictor through online reinforcement learning while keeping the underlying chunk policy frozen. These methods determine how much of a prediction to execute. We take the boundary K as given when evaluating the non-interference contract and constrain which local random inputs can influence the accepted prefix. CommitFlow satisfies this requirement at every boundary, while CommitFlow-K specializes the dependency structure to a known boundary.

3 Method

3.1 Preliminaries

Let c denote the conditioning inputs: the current visual observation, proprioceptive state, and language instruction. For a demonstrated action chunk $A \in \mathbb{R}^{H\times d}$, conditional flow matching (Lipman et al., 2023; Black et al., 2025b) interpolates between A and a noise sample ϵ:

$$X_t = (1-t)A + t\epsilon, \qquad U_t = \epsilon - A, \qquad t \in [0,1]. \tag{1}$$

The velocity field $v_\theta(c, t, X_t)$ is trained by minimizing $\mathbb{E}\|v_\theta(c,t,X_t) - U_t\|_2^2$. At inference, the corresponding ordinary differential equation (ODE) is integrated from $t = 1$ to $t = 0$ to generate an

action chunk $\hat{A}$. A receding-horizon controller executes only the first K actions, $\mathrm{Proj}_K(\hat{A}) = \hat{a}_{1:K}$, before acquiring a new observation and replanning.

With unrestricted attention across action tokens, the velocity at each position can depend on the entire action state X_t. Including the optional shared context z_g introduced in Section 3.3, this dependence can be written as

$$v_{\theta,h} = f_{\theta,h}(c, t, z_g, X_{t,1:H}). \tag{2}$$

The resulting computation graph allows tail states $X_{t,j}$,$j > K$, to influence prefix velocities $v_{\theta,h}$, $h \leq K$. Because this coupling is present at every ODE evaluation, noise assigned to tail positions can affect the generated prefix, even though the resulting tail actions are discarded by the controller. We refer to this dependence of the executed prefix on randomness in the discarded tail as tail-noise leakage.

3.2 Tail-Resampling Sensitivity

Let $\Phi_\theta(c; z_g, z_\ell)$ denote the action chunk produced by numerically integrating the learned velocity field. Here, z_ℓ collects independent local noise draws $z_{\ell,h} \sim \mathcal{N}(0, I)$ at each action position, and z_g is an optional shared latent. Under an independent and identically distributed (IID) noise prior, $\epsilon_h = z_{\ell,h}$ and z_g is omitted; Section 3.3 introduces the shared/local alternative.

To isolate the influence of discarded-tail noise, let R_K replace $z_{\ell,K+1:H}$ with independent draws from the same prior while leaving $z_{\ell,1:K}$ unchanged. Holding c and, when present, z_g fixed within each pair, we define tail-resampling sensitivity (TRS) as

$$\mathrm{TRS}_K = \mathbb{E}\left[\frac{\|\mathrm{Proj}_K\,\Phi_\theta(c; z_g, z_\ell) - \mathrm{Proj}_K\,\Phi_\theta(c; z_g, R_K z_\ell)\|_F}{\sqrt{Kd}}\right]. \tag{3}$$

The expectation is taken over evaluation contexts, initial noise draws, and independent tail resamples. TRS thus quantifies tail-noise leakage within a single sampling call, rather than task success or consistency across replans. A nonzero value beyond numerical error indicates that tail-local noise can affect the executed prefix in the implemented sampler. Its magnitude depends on the scale of the action-indexed latent representation and should not be interpreted as a distance between action distributions.

3.3 CommitFlow: Execution-Aligned Factorization

CommitFlow separates two roles. A prefix-closed action dependency graph removes tail-to-prefix paths but also removes future-to-past communication between action tokens. An optional shared/local noise construction supplies chunk-level context without restoring those action-token paths. Only the dependency constraint is required for prefix invariance.

Optional shared stochastic context. Under IID noise, each action position starts from its own local draw, $\epsilon_h = z_{\ell,h}$. The optional shared/local construction adds a chunk-level vector z_g while preserving the marginal noise distribution at every position:

$$z_g \sim \mathcal{N}(0, I), \quad z_{\ell,h} \sim \mathcal{N}(0, I), \quad \epsilon_h = \sqrt{\rho}\, z_g + \sqrt{1-\rho}\, z_{\ell,h}, \tag{4}$$

where z_g and the $z_{\ell,h}$ are mutually independent and $0 \leq \rho < 1$. This construction preserves the standard Gaussian marginal at each position, $\epsilon_h \sim \mathcal{N}(0, I)$, while introducing cross-position covariance $\mathrm{Cov}(\epsilon_h, \epsilon_j) = \rho I$ for $h \neq j$. We also provide a learned projection of z_g as a shared conditioning token. Every action token can attend to this token, but the shared token cannot read or aggregate action states. The local innovations $z_{\ell,h}$ remain position-specific. Accordingly, TRS holds z_g fixed and resamples only the tail-local innovations, isolating the effect of randomness assigned exclusively to discarded positions.

The resulting prior also satisfies

$$\mathbb{E}\|\epsilon_{h+1} - \epsilon_h\|_2^2 = 2d(1-\rho). \tag{5}$$

With $\rho = 0.5$, the expected squared difference between adjacent noise vectors is half its IID value of $2d$, while each marginal remains standard Gaussian. This identity concerns the noise prior only; the corresponding action-space diagnostics are reported in Appendix Table E.2 and Sections C.3–C.4.

Prefix-closed action dependencies. To block tail-to-prefix influence for every possible execution boundary, strict CommitFlow imposes a lower-triangular dependency relation over local action tokens:

$$v_{\theta,h} = f_{\theta,h}(c, t, z_g, X_{t,1:h}), \tag{6}$$

Unlike the base model in Equation (2), which allows each position to access all H action states, strict CommitFlow restricts position h to its own state and those at earlier positions. The z_g argument is omitted under IID noise.

At each ODE step, action token h may attend to the conditioning tokens, the optional shared token, and action tokens $1{:}h$. Neither the conditioning tokens nor the shared token may attend to action states. This mask is applied at every transformer layer during both training and sampling, eliminating direct and indirect paths from any later position $j > h$ to h. All action-token velocities are still computed in parallel in a single masked transformer call per ODE evaluation.

Proposition 1 (tail-local prefix invariance). Suppose that the conditioning and shared tokens do not depend on action states, the action dependencies are lower triangular, and the induced prefix initial-value problem has a unique solution. For fixed c, z_g, and K, any two local-noise draws z_ℓ and z'_ℓ satisfying $z_{\ell,1:K} = z'_{\ell,1:K}$ produce identical prefixes:

$$\mathrm{Proj}_K\, \Phi_\theta(c; z_g, z_\ell) = \mathrm{Proj}_K\, \Phi_\theta(c; z_g, z'_\ell). \tag{7}$$

Consequently, resampling $z_{\ell,K+1:H}$ while holding z_g fixed gives $\mathrm{TRS}_K = 0$ up to numerical precision. Appendix A provides the proof. This is a counterfactual guarantee within a single sampling call; it does not imply that the masked and unrestricted models induce the same joint action distribution.

Known execution boundary. When the execution boundary K is known, enforcing closure at every prefix is stronger than necessary. The controller commits to the first K actions as a block, so these actions may interact bidirectionally as long as they remain isolated from the tail. We refer to this variant as CommitFlow-K:

$$v_{\theta,1:K} = F_{\theta,K}(c, t, z_g, X_{t,1:K}), \qquad v_{\theta,K+1:H} = G_{\theta,K}(c, t, z_g, X_{t,1:H}), \tag{8}$$

where z_g is omitted under IID noise. The committed block cannot attend to the tail, while tail positions may attend to both blocks. The same closure and uniqueness argument establishes $\mathrm{TRS}_K = 0$ at the specified boundary, but does not guarantee invariance at an arbitrary boundary $K' \neq K$.

Figure 2 compares the three dependency patterns. Rows and columns index target and source action positions, respectively; filled cells indicate allowed dependencies, and black lines mark the illustrated execution boundary. Base permits unrestricted interaction across action tokens. Strict CommitFlow isolates every prefix through a lower-triangular dependency structure, whereas CommitFlow-K isolates the specified execution block while retaining bidirectional interaction within it. Shared stochastic context can be used with any of these patterns, but does not itself enforce prefix closure.

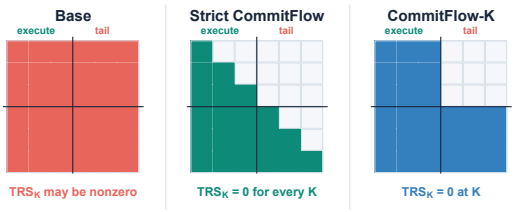

Figure 2: **Execution-aligned dependency designs.**

4 Experiments and Results

The evaluation begins with a controlled tail-resampling probe. In this probe, the complete conditioning input and all non-tail randomness remain fixed while only tail-local noise is resampled, allowing the resulting prefix deviation to be measured directly. The same tail-only intervention is then applied in LIBERO (Liu et al., 2023) rollouts to test whether the resulting change in the executed prefix can alter the final task outcome. Further LIBERO experiments evaluate overall task success, separate the effects of dependency structure and stochastic context, and measure consistency across replans. A separate study evaluates Block-5, a known-boundary design that retains bidirectional interaction

within the executed block while isolating it from the discarded tail. Beyond LIBERO, RoboTwin 2.0 (Chen et al., 2025) extends the evaluation to a longer action-prediction horizon across all 50 clean-domain tasks, while online AgiBot G1 trials assess the method on physical hardware.

All evaluated policies share a common $\pi_{0.5}$ backbone (Black et al., 2025a). For each comparison, training data and budget, initial states, and evaluation conditions are matched across methods. *Base* uses unrestricted action-token attention with IID noise, whereas CommitFlow uses strict prefix closure with shared/local stochastic context ($\rho = 0.5$). Appendices B and D.1, together with the benchmark-specific ledgers, provide full protocols, denominators, pairing rules, and statistical procedures.

4.1 Tail-to-Prefix Sensitivity

At $H = 30$, we fix c, prefix innovations, the shared context when present, and all other non-tail randomness for each held-out AgiBot G1 observation. We resample only the tail-local noise assigned to discarded actions. Figure 3 reports the resulting normalized prefix RMSE and maximum deviation across diagnostic prefix lengths. Panel (a) also shows a 95% observation-cluster bootstrap interval. Appendix C.1 provides the complete protocol and numerical values.

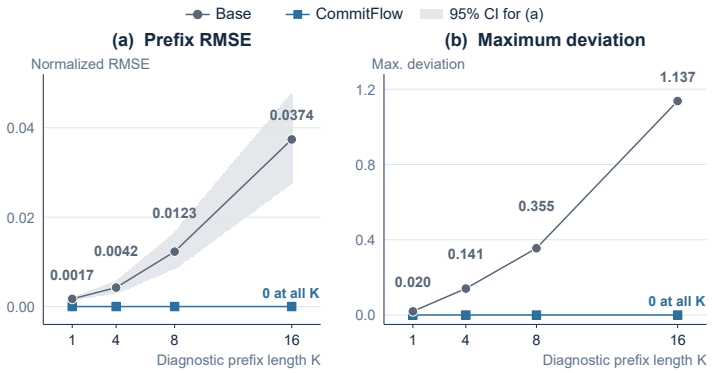

Figure 3: **Tail-resampling sensitivity.** (a) Mean normalized prefix RMSE with 95% bootstrap intervals. (b) Maximum observed prefix deviation.

Both panels show that Base is sensitive to tail noise at every diagnostic prefix length. The mean and maximum deviations increase with K_{diag}, and the much larger maximum shows that individual prefixes can move far more sharply than the average suggests. In contrast, strict CommitFlow remains invariant throughout, as guaranteed by Proposition 1. Because tail noise is the only changed input, this controlled contrast isolates a tail-to-prefix dependence in Base and provides an empirical implementation check for strict CommitFlow.

4.2 Closed-Loop Effects of Tail Resampling

To determine whether prefix changes caused by discarded-tail noise can ultimately affect task success, we conduct a paired intervention on 40 fixed LIBERO states with $H = 10$ and $K = 5$. For each state and each intervention range $m \in \{0, 1, 3, 5\}$, we perform four rollouts with the same environment initialization, prefix-local noise, and randomness in all subsequent policy calls. Across these four rollouts, only the first-call local noise assigned to the last m positions of the discarded tail is resampled. The $m = 0$ condition serves as a no-resampling control, whereas $m = 5$ resamples the entire five-action tail.

Table 1: **Closed-loop effect of discarded-tail resampling on LIBERO ($H = 10$, $K = 5$).** Forty fixed states receive four matched rollouts for every method and intervention range, yielding 160 rollouts for each method and value of m. Prefix RMSE averages the six pairwise deviations among the four executed prefixes. "Mixed" counts states whose four rollouts contain both success and failure.

m	Base			CommitFlow		
	Success	Prefix RMSE	Mixed	Success	Prefix RMSE	Mixed
0	100.00%	0.000000	0/40	100.00%	**0.000000**	**0/40**
1	98.75%	0.007527	1/40	100.00%	**0.000000**	**0/40**
3	96.88%	0.009361	4/40	100.00%	**0.000000**	**0/40**
5	95.00%	0.010334	6/40	100.00%	**0.000000**	**0/40**

Under Base, widening the intervention increases both prefix deviation and outcome variation. From $m = 0$ to $m = 5$, mixed states increase from 0/40 to 6/40, while successes decrease from 160/160 to 152/160. Because first-call discarded-tail noise is the only manipulated variable, these within-state changes show that discarded-tail noise alone can propagate through execution and alter task success. CommitFlow remains invariant, with zero prefix RMSE, no mixed states, and 160/160 successes at every m. Appendix C.2 details the protocol, nested noise pairing, and aggregate results.

4.3 Task Success and Design Analysis on LIBERO

4.3.1 Overall Task Success and Component Ablation

Having shown that tail resampling can alter a closed-loop outcome, we evaluate whether removing dependence on tail-local noise preserves task success and how the two components of CommitFlow contribute. The overall LIBERO comparison spans three execution lengths, while a matched 2×2 ablation separates action-token dependencies from stochastic context (Fig. 4).

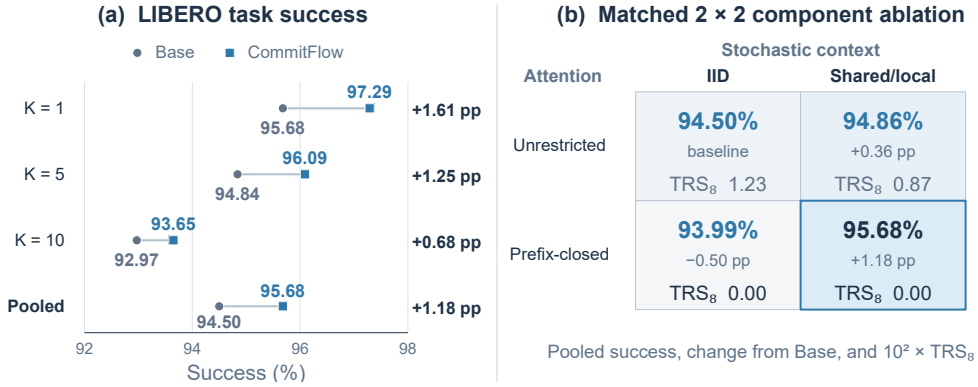

Figure 4: **LIBERO task success and matched component ablation.** (a) Success across execution lengths. (b) Pooled 2×2 ablation.

Across the three execution lengths, CommitFlow consistently improves success, raising the pooled rate from 94.50% to 95.68%, a gain of 1.18 pp. The 95% paired task-bootstrap interval for this gain is $[0.49, 1.84]$ pp. The gain is largest at $K = 1$ and narrows toward $K = 10$, consistent with tail isolation mattering more when a larger fraction of the chunk is discarded. Because K also changes replanning frequency, this trend reflects the broader commitment-length trade-off rather than tail length alone. The positive difference at $K = H = 10$, where no tail is discarded, indicates that the observed success-rate difference cannot be attributed to leakage removal alone.

The matched 2×2 ablation clarifies the role of each component. Prefix closure is sufficient for zero observed TRS_8, while its success point estimate without shared context is 0.50 pp below Base+IID. Under unrestricted attention, shared/local context raises the success point estimate by 0.36 pp but leaves a nonzero tail path. Under prefix closure, the corresponding point-estimate difference is 1.68 pp, yielding the only zero-TRS configuration that also exceeds Base. The positive interaction ($+1.32$ pp) supports a complementary division of labor in which closure provides structural isolation and shared context improves measured task success under strict prefix closure.

4.3.2 Block-5 as a Known-Boundary Control

The preceding factorial study provides the primary evidence for task success and component effects. As an exploratory known-boundary control, Block-5 tests whether the lower success point estimate under strict closure may reflect lost interaction among actions executed together. Actions 1–5 interact bidirectionally, while positions 6–10 cannot influence them.

Table 2: **LIBERO known-boundary control.** $H = 10$, $K = 5$, and $\mathrm{TRS}_5 = 0$ for both variants.

Configuration	Success	95% CI	$10^2 \times \mathrm{TRS}_5$
Block-5+IID	**96.82%**	$[94.27, 98.59]$	**0.00**
Block-5+shared/local	96.67%	$[94.79, 98.18]$	**0.00**

Under IID noise, Block-5 reaches 96.82%, compared with 94.32% for strict prefix closure at $K = 5$. This unpaired 2.50 pp difference is descriptive and does not establish a boundary effect, but its direction motivates the hypothesis that retaining within-block interaction can benefit task success.

Within Block-5, replacing IID noise with shared/local context changes success from 96.82% to 96.67%, a paired difference of -0.16 pp with a 95% interval of $[-1.30, 1.04]$ pp. This provides no evidence of an additional task-success benefit from shared context, while both variants retain zero observed TRS_5. Together, these results motivate a boundary-specific design that isolates the committed block from discarded positions while preserving bidirectional interaction within it (Appendix Table E.4).

4.4 Replanning Consistency

Prefix invariance concerns a single planning call, but closed-loop control also depends on revisions after new observations. We therefore measure overlap revision between actions assigned to the same future steps in consecutive plans and boundary jump between successive executed controls across a replanning boundary. Table 3 reports both metrics. Lower values indicate greater consistency, and Appendix B.4 provides detailed definitions and paired statistics.

Table 3: **Across-replan consistency on paired LIBERO episodes.** Values and paired-difference intervals are scaled by 10^2, with lower values indicating greater consistency.

Metric (K)	Base	CommitFlow	Relative Δ	Paired 95% CI
Overlap revision (1)	33.49	**31.21**	-6.79%	$[-2.90, -1.66]$
Overlap revision (5)	48.78	**46.00**	-5.71%	$[-4.05, -1.52]$
Boundary jump (5)	52.81	**51.37**	-2.74%	$[-2.38, -0.50]$
Boundary jump (10)	63.44	**61.42**	-3.19%	$[-3.94, -0.22]$

CommitFlow has lower values on all four metrics, with every paired-difference interval excluding zero. Relative reductions are 5.71%–6.79% for overlap revision and 2.74%–3.19% for boundary jump. The overlap-revision reduction indicates that the difference extends across future steps shared by consecutive plans rather than only the immediate boundary. Together with the within-call TRS results, these findings show that CommitFlow eliminates within-call tail-to-prefix dependence and exhibits more consistent updates across consecutive plans on the evaluated trajectories. This does not imply universally smoother or more responsive control, and CommitFlow has higher within-chunk second differences, as quantified in Appendix C.4.

4.5 Longer-Horizon Evaluation on RoboTwin

We next extend the evaluation to a longer action-prediction horizon on RoboTwin 2.0, setting $H = 50$ and testing all 50 clean-domain tasks at $K \in \{10, 20, 50\}$. Figure 5 summarizes aggregate success and task-level win/tie/loss counts.

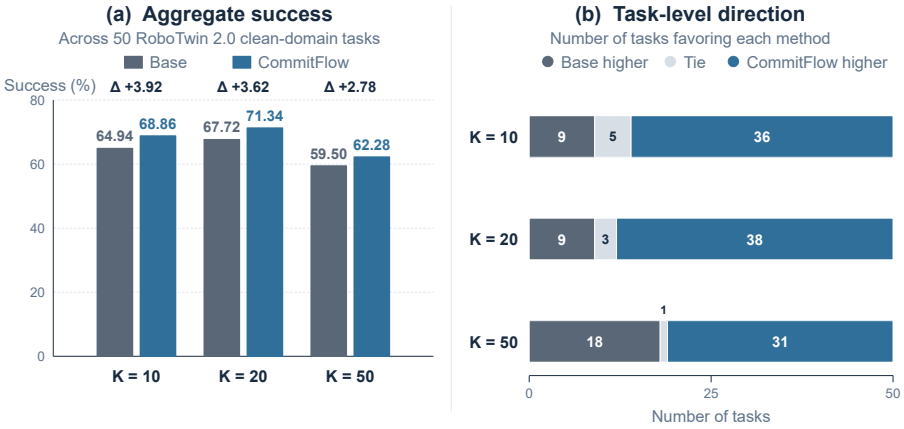

Figure 5: **RoboTwin across 50 clean-domain tasks.** (a) Aggregate success. Δ labels report CommitFlow minus Base. (b) Task-level win/tie/loss counts.

CommitFlow achieves higher observed success at all three execution lengths, with gains of 3.92 pp at $K = 10$, 3.62 pp at $K = 20$, and 2.78 pp at $K = 50$. The gain is largest at $K = 10$, when 40 of the 50 predicted actions are discarded, and narrows as K increases. Because K also changes replanning frequency, this trend cannot be attributed solely to tail isolation. The remaining 2.78 pp gain at $K = H = 50$, where no action is discarded, indicates that dependency structure and stochastic context also contribute to the aggregate difference.

Task-level counts show that the aggregate gain is broad but not uniform. CommitFlow records higher observed success on 36 of 50 tasks at $K = 10$ and 38 at $K = 20$. At $K = 50$, the corresponding counts are 31 for CommitFlow and 18 for Base, yielding a more mixed result under full-chunk execution. Relative performance therefore varies by task and execution length. Appendix Tables F.1 and F.2 provide complete aggregate and task-level results, and Appendix F.3 summarizes positive and negative task-level changes.

4.6 Online Hardware Evaluation

The final evaluation compares Base and CommitFlow on seven fixed-start AgiBot G1 tasks, testing whether the aggregate simulation trend extends to physical execution.

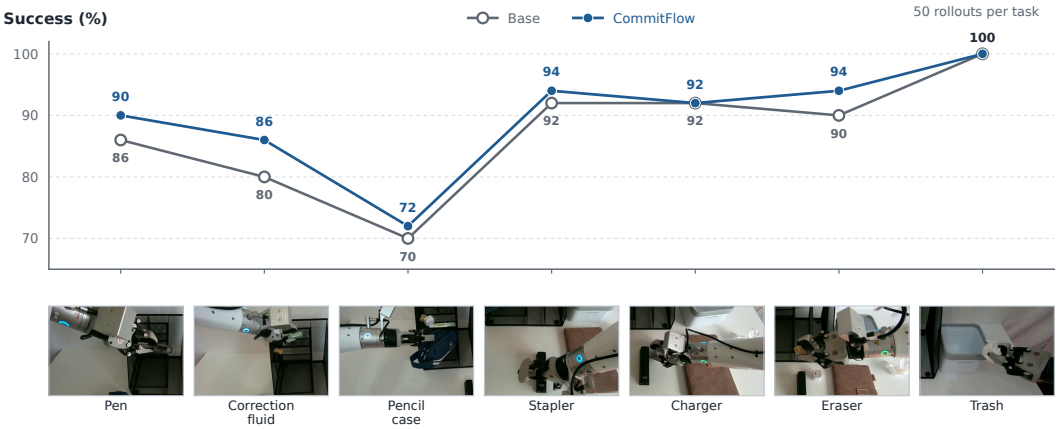

Figure 6: **Online success across seven fixed-start AgiBot G1 tasks.** The lower row shows representative frames from successful physical rollouts to identify each task.

CommitFlow records higher observed success on five of seven tasks and matches Base on two. The largest task-level gain is $+6$ pp on correction fluid, while aggregate success rises from 87.14% for Base to 89.71% for CommitFlow ($+2.57$ pp). The gains span several manipulations rather than a single task. These fixed-start trials provide hardware evidence consistent with the aggregate simulation trend. Appendix G reports the protocol, complete per-task results, and a separate analysis of the book task.

5 Conclusion, Limitations, and Future Work

Conclusion. Taken together, the results establish execution-boundary non-interference as a useful design principle for stochastic action-chunk policies. Randomness assigned only to discarded actions can otherwise influence the controls that are actually executed, whereas CommitFlow removes this dependence and achieves higher aggregate task success across the evaluated simulation benchmarks and hardware trials. More broadly, the controller's commitment boundary should be reflected in the sampler's dependency structure rather than imposed only after a fully coupled action chunk has been generated.

Limitations and future directions. The current guarantee concerns within-call dependence on tail-local noise rather than general closed-loop robustness, and the empirical study is limited to a $\pi_{0.5}$-family backbone. Extending execution-aligned sampling to other stochastic action generators and adaptive commitment boundaries remains an important direction for future work.

Reproducibility statement. Appendices A–G document the invariance proof, mask and implementation parity, matched evaluation grids, aggregation and retry rules, bootstrap procedures, diagnostic definitions, and simulation and real-robot result ledgers.

AI use statement. Generative AI tools were used to assist in drafting and polishing portions of the manuscript, including improvements to wording, organization, and presentation. The authors determined the scientific content, developed the method, designed and conducted the experiments, verified the analyses and citations, and take full responsibility for the final manuscript.

REFERENCES

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

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. In *Proceedings of the 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40, 2025a. URL https://proceedings.mlr.press/v305/black25a.html.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. In *Proceedings of Robotics: Science and Systems*, 2025b. doi: 10.15607/RSS.2025.XXI.010. URL https://www.roboticsproceedings.org/rss21/p010.html.

Kevin Black, Manuel Galliker, and Sergey Levine. Real-time execution of action chunking flow policies. In *Advances in Neural Information Processing Systems 38*, pp. 37596–37620, 2025c. doi: 10.52202/085713-1122.

Kevin Black, Allen Z Ren, Michael Equi, and Sergey Levine. Training-time action conditioning for efficient real-time chunking. *arXiv preprint arXiv:2512.05964*, 2025d.

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *Advances in Neural Information Processing Systems*, 37:24081–24125, 2024.

Feng Chen, Xianghui Wang, Yuxuan Chen, Boying Li, Yefei He, Zeyu Zhang, and Yicheng Wu. Dynamic execution commitment of vision-language-action models. *arXiv preprint arXiv:2605.11567*, 2026.

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

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

Xiangdong Feng, Yuxuan Cheng, Chen Shi, Boyao Han, Yuxuan Yan, Yitong Hong, Zhuotao Tian, and Li Jiang. Denoising tells when to replan: Denoising-variance adaptive chunking for flow-based robot policies, 2026. URL https://arxiv.org/abs/2606.03847.

Songwei Ge, Seungjun Nah, Guilin Liu, Tyler Poon, Andrew Tao, Bryan Catanzaro, David Jacobs, Jia-Bin Huang, Ming-Yu Liu, and Yogesh Balaji. Preserve your own correlation: A noise prior for video diffusion models. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22873–22884. IEEE, 2023.

Trong-Bao Ho, Quang-Tan Nguyen, Thien-Loc Ha, Gia-Binh Nguyen, Viet-Thanh Nguyen, Long Dinh, Minh N. Vu, Duy M. H. Nguyen, An Thai Le, and Ngo Anh Vien. Start right, arrive right: Asynchronous execution via initial noise selection. *arXiv preprint arXiv:2606.19774*, 2026. URL https://arxiv.org/abs/2606.19774.

Sunshine Jiang, Xiaolin Fang, Nicholas Roy, Tomás Lozano-Pérez, Leslie Pack Kaelbling, and Siddharth Ancha. Streaming flow policy: Simplifying diffusion/flow-matching policies by treating action trajectories as flow trajectories. *arXiv preprint arXiv:2505.21851*, 2025.

Zekai Li, Jiaming Tang, and Zhijian Liu. Flashvla: Streaming action decoding for fast and asynchronous vla inference. *arXiv preprint arXiv:2608.27384*, 2026.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

Yuejiang Liu, Jubayer Hamid, Annie Xie, Yoonho Lee, Max Du, and Chelsea Finn. Bidirectional decoding: Improving action chunking via guided test-time sampling. In *International Conference on Learning Representations*, volume 2025, pp. 4594–4627, 2025.

Jiahua Ma, Yiran Qin, Yixiong Li, Xuanqi Liao, Yulan Guo, and Ruimao Zhang. Cdp: Towards robust autoregressive visuomotor policy learning via causal diffusion. *arXiv preprint arXiv:2506.14769*, 2025.

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, et al. Smolvla: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025.

Haoxuan Wang, Gengyu Zhang, Yan Yan, Ramana Rao Kompella, and Gaowen Liu. Vla knows its limits: Adaptive execution horizons for robot policies. In *European Conference on Computer Vision*, pp. 607–623. Springer, 2026a.

Haoxuan Wang, Gengyu Zhang, Yan Yan, Yuzhang Shang, Ramana Kompella, and Gaowen Liu. Real-time robot execution with masked action chunking. In *International Conference on Learning Representations*, 2026b. URL https://arxiv.org/abs/2601.20130.

Wei Wu, Fangjing Wang, Fan Lu, He Sun, Shi Liu, Yunnan Wang, Yibin Yan, Yong Wang, Shuailei Ma, Xinyang Wang, et al. From foundation to application: Improving vla models in practice. *arXiv preprint arXiv:2607.06403*, 2026.

Dijia Zhan, Xuemiao Xu, Jinyi Li, and Jie Tang. Seam: Smooth execution of action-chunked motion for vision-language-action policies. *arXiv preprint arXiv:2607.04609*, 2026.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Proceedings of Robotics: Science and Systems*, 2023. doi: 10.15607/RSS.2023.XIX.016. URL https://www.roboticsproceedings.org/rss19/p016.html.

Yuchi Zhao, Miroslav Bogdanovic, Arjun Sohal, Liyu Tao, Kourosh Darvish, Alán Aspuru-Guzik, Florian Shkurti, and Animesh Garg. Dynamic execution horizon prediction for chunk-based robot policies. *arXiv preprint arXiv:2606.11408*, 2026.

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, et al. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. In *International Conference on Learning Representations*, volume 2026, pp. 60580–60606, 2026.

A Theory and Proofs

A.1 Prefix-Invariance Proof

Fix c, z_g, and K, and let $x(t)$ and $x'(t)$ be two flow trajectories with $z_{\ell,1:K} = z'_{\ell,1:K}$. Because conditioning and shared-token states never read action-token states, they are identical functions of (c, t, z_g) along the two trajectories; in particular, the tail cannot influence the prefix through an indirect global-token path. Together with the lower-triangular action mask, this implies

$$v_{\theta,1:K}(c, t, z_g, x) = F_{\theta,K}(c, t, z_g, x_{1:K}),$$

so the first K coordinates form a closed subsystem. Moreover, $y(1) = y'(1)$ for $y = \mathrm{Proj}_K\, x$ and $y' = \mathrm{Proj}_K\, x'$, because the two draws share z_g and their first K local innovations. Thus y and y' solve the same initial-value problem

$$\frac{dy}{dt} = F_{\theta,K}(c, t, z_g, y), \qquad y(1) = y'(1). \tag{A.1}$$

By the assumed uniqueness of this prefix initial-value problem, $y(t) = y'(t)$ over the integration interval and, in particular, at $t = 0$. This proves equation 7. For the known-boundary construction in equation 8, closure of the committed block $1{:}K$ replaces the lower-triangular premise; the same initial-value argument proves invariance at that fixed K. In finite precision, the diagnostic should be zero up to kernel and reduction error.

A.2 Dependency and Covariance Geometry

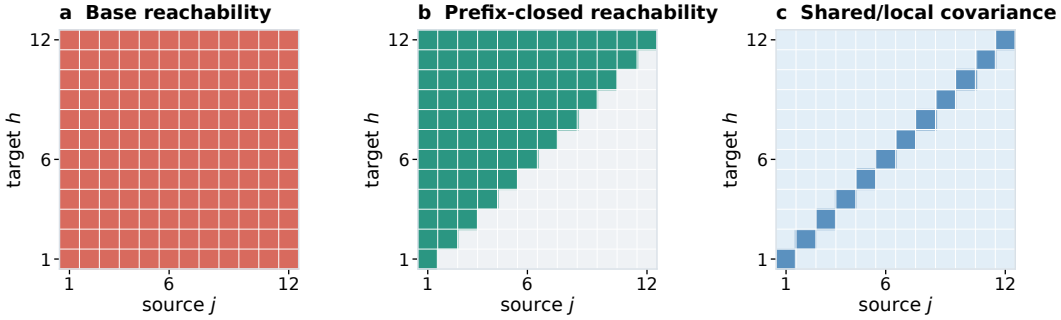

Figure A.1: **Dependency and covariance geometry.** Unrestricted Base attention permits every local position to affect every target. Strict CommitFlow permits only source positions $j \le h$; panel (b) plots target index h from 1 at the bottom to 12 at the top. Action-independent conditioning and shared-token sources remain globally visible but do not read action tokens, closing indirect tail-to-prefix paths. The shared/local construction preserves unit diagonal covariance and gives off-diagonal covariance ρI. Cell color denotes allowed dependence or prior correlation, not learned influence magnitude.

Figure A.1 separates permitted local-token reachability from the covariance supplied by the chunk-shared factor; cell color encodes structural allowance or prior correlation rather than measured effect size.

B Evaluation Protocols and Statistical Details

B.1 Completeness, Pairing, and Retry Rules

Only complete result manifests enter an aggregate. For each suite cell in the three-evaluation-seed LIBERO grid (Table E.3), we use the predesignated primary run only when both Base and CommitFlow pass the episode-count and zero-exception checks; otherwise we use the earliest complete paired retry. Failed single arms and duplicate directories never enter the denominator. The factorial comparisons in Fig. 4(b) and Table E.1 are admitted only as a complete four-configuration

block with matched seeds, tasks, and episode counts; a missing or failed single configuration is not substituted into that block. The trajectory study likewise requires paired Base/CommitFlow records for every included episode. All completeness and retry decisions are resolved before comparing success outcomes. The Block-5 study is admitted only after all 48 manifests—two configurations, two training seeds, three evaluation seeds, and four LIBERO suites—pass the prescribed episode-count and zero-exception checks. The resulting block contains 3,840 episodes and records zero exceptions.

B.2 Pooled Factorial Success-Rate Uncertainty

For the pooled LIBERO factorial analysis, the statistical unit is the task. Each of 20,000 bootstrap draws resamples 10 tasks with replacement within each of the four suites, preserving equal suite weights; the same sampled task indices are applied to all four configurations, and each task retains its 144 aggregated episodes. We recompute the four success rates, the three differences relative to Base+IID, and the pooled interaction $I_{\text{succ}} = [S(\text{CommitFlow}) - S(\text{Prefix-closed+IID})] - [S(\text{Base+shared/local}) - S(\text{Base+IID})]$ for every draw, then report the 2.5th and 97.5th percentiles using RNG seed 20260904. The configuration-rate intervals for Base+IID, Base+shared/local, Prefix-closed+IID, and the full model are $[93.00, 95.89]$, $[93.23, 96.34]$, $[92.48, 95.43]$, and $[94.08, 97.10]\%$, respectively. The corresponding paired difference intervals are $[-0.14, 0.87]$, $[-1.29, 0.30]$, and $[0.49, 1.84]$ pp; the interaction interval is $[0.21, 2.38]$ pp. These intervals quantify variation across tasks at fixed suite weights; they are not episode-level binomial or seed-stratified intervals.

B.3 Block-5 Success-Rate Uncertainty

For the Block-5 analysis, the statistical unit is again the task. Each of 20,000 bootstrap draws resamples 10 tasks with replacement within each LIBERO suite, using the same sampled task indices for the IID and shared/local configurations; every task retains its 48 episodes. Percentiles use RNG seed 20260904. The marginal 95% intervals are $[94.27, 98.59]\%$ for Block-5+IID and $[94.79, 98.18]\%$ for Block-5+shared/local. The paired shared/local-minus-IID contrast is -0.16 pp with interval $[-1.30, 1.04]$ pp. Block-versus-Prefix-closed and Block-versus-Base differences are treated as descriptive because the corresponding earlier $K = 5$ task vectors are unavailable.

B.4 Trajectory Metrics and Statistical Inference

Continuous action dimensions 0–5 are included and the binary gripper dimension is excluded. Each reported K contains 40 tasks and 400 matched episode pairs (400 episodes per method). The statistical unit is the task: confidence intervals use 20,000 task-cluster bootstrap draws. Benjamini–Hochberg correction is applied jointly across seven designated primary tests: overlap revision at $K \in \{1, 5\}$, boundary jump at $K \in \{5, 10\}$, and within-chunk second difference at $K \in \{1, 5, 10\}$. For the four rows shown in Table B.1, the adjusted q values in table order are 8.75×10^{-5}, 2.10×10^{-4}, 0.00589, and 0.0414; for within-chunk second difference at $K = 1, 5, 10$, the adjusted q value is 8.75×10^{-5} in each case. Overlap revision compares the common physical-time support of consecutive plans; it is undefined at $K = H = 10$ because successive plans have no overlapping support. Boundary jump compares successive executed controls across a replan and is reported for $K \in \{5, 10\}$; no $K = 1$ value is reported or inferred.

Table B.1: **Replanning consistency on paired LIBERO episodes.** Values and paired-difference intervals are scaled by 10^2; lower is more consistent.

Metric (K)	Base	CommitFlow	Relative Δ	$10^2\times$95% CI
Overlap revision (1)	33.49	**31.21**	-6.79%	$[-2.90, -1.66]$
Overlap revision (5)	48.78	**46.00**	-5.71%	$[-4.05, -1.52]$
Boundary jump (5)	52.81	**51.37**	-2.74%	$[-2.38, -0.50]$
Boundary jump (10)	63.44	**61.42**	-3.19%	$[-3.94, -0.22]$

B.5 Illustrative Trace Protocol

For visualization, Figure C.2 uses the episode closest to the median joint improvement among 138 matched episode pairs in which both rollouts succeed and contain at least 100 steps, and for which CommitFlow has lower boundary jump and overlap revision. This conditioning makes the trace illustrative rather than representative of all matched episodes; quantitative claims rely on Table B.1.

B.6 Illustrative Benchmark Context

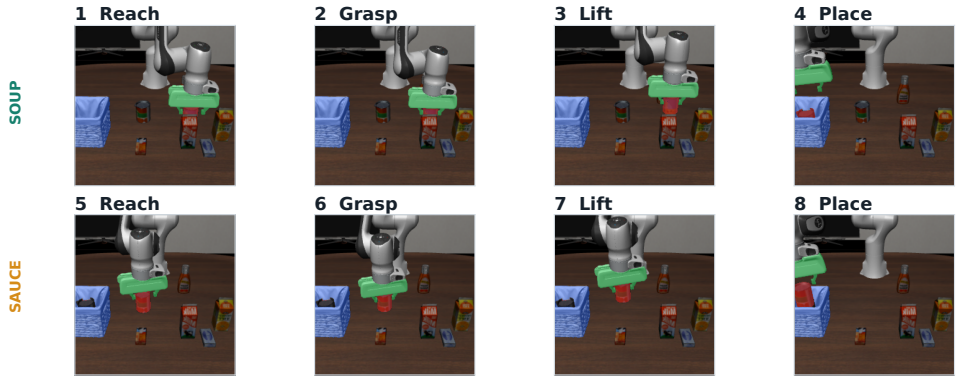

Figure B.1: **Illustrative LIBERO-10 interaction sequence, not a method comparison.** The two rows follow the two target objects through reach–grasp–lift–place for the instruction "put both the alphabet soup and the tomato sauce in the basket." This figure provides task context only; all method comparisons use the prespecified evaluations in the main text.

C Mechanism Diagnostics

C.1 Empirical Tail-Resampling Sensitivity

We evaluate configuration-matched Base and full CommitFlow final 30k-step checkpoints trained on the same in-house real-robot desk-manipulation data ($H = 30$). At each diagnostic prefix length $K_{\text{diag}} \in \{1, 4, 8, 16\}$, the offline probe uses 16 held-out observations, eight tail-local resamples per observation, 128 records, and a 10-step ODE solver. This mechanism-only setting is independent of the LIBERO ablation and does not provide online task-success evidence. Mean normalized prefix RMSE directly estimates the quantity in equation 3; 95% intervals resample observations after averaging the eight repeats within each observation. We additionally retain the complementary maximum statistic

$$\widehat{S}_{\max,\infty}(K_{\text{diag}}) = \max_{\substack{i=1,\dots,16 \\ r=1,\dots,8}} \left\| \text{Proj}_{K_{\text{diag}}} \Phi_\theta(c_i; z_{g,i}, z_{\ell,i}) - \text{Proj}_{K_{\text{diag}}} \Phi_\theta\big(c_i; z_{g,i}, R^{(i,r)}_{K_{\text{diag}}} z_{\ell,i}\big) \right\|_\infty , \tag{C.1}$$

which asks whether any recorded tail-local resample changes the prefix under analysis.

Table C.1: **Offline diagnostic values on real-robot observations ($H = 30$).** Base RMSE values and intervals are scaled by 10^3; L_∞ values use the original scale. Each prefix length contains 128 resampling records.

K_{diag}	$10^3\times$ Base RMSE mean [95% CI]	$10^3\times$ CommitFlow RMSE mean	Base L_∞ p95	Base L_∞ maximum
1	1.72 [1.52, 1.94]	**0.00**	0.01	0.02
4	4.24 [3.00, 5.71]	**0.00**	0.08	0.14
8	12.29 [8.61, 16.49]	**0.00**	0.24	0.35
16	37.40 [27.78, 47.58]	**0.00**	0.62	1.14

The zero values are the empirical signature predicted by Proposition 1. The nonzero Base values show that tail-local perturbations can reach the diagnostic prefix when action-token dependencies are unrestricted. This controlled comparison does not by itself attribute task-success differences or separate the two architectural components; that factorial component comparison is reported independently in Fig. 4(b).

C.2 Paired Closed-Loop Tail-Range Intervention

We conduct a descriptive closed-loop intervention with the matched final 30k-step, training-seed-3047 Base and strict-CommitFlow checkpoints at $H = 10$ and $K = 5$. Task selection is outcome-blind: for each suite, we hash the frozen identifier `LIBERO-TAIL-INTERVENTION-V1` together with the suite name using SHA-256, interpret the first eight digest bytes as an unsigned big-endian integer, and reduce modulo ten. The resulting zero-based task IDs are Spatial 6 (black bowl next to cookie box $\rightarrow$ plate), Object 3 (BBQ sauce $\rightarrow$ basket), Goal 7 (turn on the stove), and LIBERO-10 2 (turn on the stove and put the moka pot on it). Initial-state indices 0–9 are retained for every task, giving 40 fixed states. Base and CommitFlow use the identical state array, environment seed 20260906, and policy-noise seed 2026090601.

Each state receives four rollouts per method and intervention range. For $m > 0$, only the last m positions of the discarded first-call tail, $z_{\ell,11-m:10}$, are resampled; the environment initialization, prefix-local innovations $z_{\ell,1:5}$, the remaining tail positions, shared latent when present, and all counter-keyed random variables in later policy calls are held fixed. Thus, $m = 1$, 3, and 5 resample positions 10, 8–10, and 6–10, respectively. Across values of m, the same sampled noise is reused whenever a $(\text{state}, \text{draw}, \text{position})$ tuple appears in more than one intervention range, so the noise realizations are nested rather than regenerated independently. The $m = 5$ condition is the original full-tail intervention. At $m = 0$, the intervention set is empty and the identical realization is repeated four times as a negative control, rather than providing four independent samples. For $m > 0$, draw 0 has no special statistical role. For the seven-dimensional executed-action prefix, all six unordered draw pairs enter the same statistic used in the original intervention:

$$r_{s,ij} = \sqrt{\frac{1}{5 \times 7} \sum_{t=1}^{5} \sum_{d=1}^{7} \left(a_{s,i,t,d} - a_{s,j,t,d}\right)^2}, \qquad \overline{r} = \frac{1}{40 \times 6} \sum_{s=1}^{40} \sum_{i<j} r_{s,ij}. \tag{C.2}$$

A state is counted as outcome-varying if its four binary success labels are not all identical. Because the states form a fixed mechanism-diagnostic set, the rollout success fractions and outcome-variation counts below are descriptive and are not treated as independent benchmark estimates or hypothesis tests. In particular, the rows compare intervention scopes on the same states at the same K; they do not estimate a general success-rate curve as a function of m.

Table 1 summarizes the range intervention. On this fixed set, broader Base interventions produce larger executed-prefix deviations and more states with mixed outcomes. At $m = 5$, the six outcome-varying states contain all eight failed Base rollouts; the other 34 states succeed under every draw, and no state fails under every draw. The zero CommitFlow entries are the closed-loop consequence expected from its prefix-invariance property under the fixed subsequent-randomness protocol. The experiment establishes sensitivity to discarded-tail choice but does not identify how much of the aggregate LIBERO success difference is caused by tail isolation.

C.3 Complementary Diagnostics

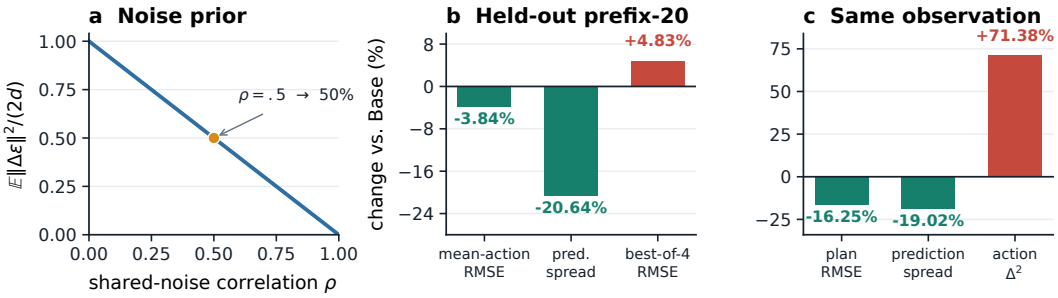

Figure C.1: **Complementary mechanism diagnostics from separate protocols.** Panel (a) shows analytic prior geometry. In panels (b) and (c), green and red denote decreases and increases relative to Base; direction is interpreted per metric. Panel (b) uses held-out real-robot prefix-20 prediction ($H = 30$), whereas panel (c) uses LIBERO same-observation resampling ($H = 10$). These panels use protocols distinct from the tail-local resampling probe in Figure 3.

Figure C.1 places three deliberately separate protocols side by side for qualitative comparison; no pooled statistical claim is made across panels.

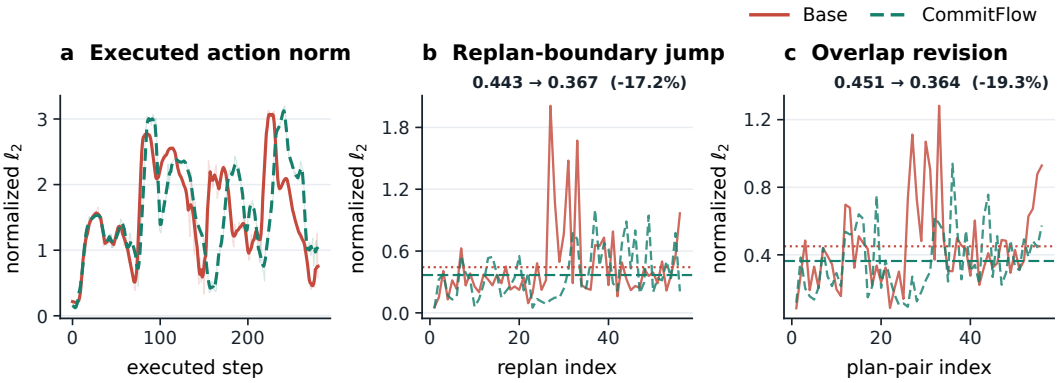

Figure C.2: **Illustrative paired successful LIBERO-10 rollout.** Both methods complete the task (initialization 4, $K = 5$). Panel (a) overlays faint raw traces with five-step moving averages; the horizontal dotted and dashed lines in panels (b) and (c) mark method means. The trace is nearest the median joint improvement among 138 jointly successful pairs with lower CommitFlow boundary jump and overlap revision; Table B.1 gives aggregate results.

A separate held-out real-robot replay protocol evaluates 512 observations per method at prefix length 20, with four stochastic predictions per observation under each of three sampling seeds and a 10-step ODE solver. For target prefix y and predictions $\{\hat{a}^{(m)}\}_{m=1}^{4}$, mean-action RMSE compares $\frac{1}{4}\sum_m \hat{a}^{(m)}$ with y; prediction-spread RMS is the square root of the mean coordinate-wise sample variance over the prefix; and best-of-four RMSE is $\min_m \mathrm{RMSE}(\hat{a}^{(m)}, y)$. Metrics are averaged across the three sampling seeds within each observation before pooling observations. They measure offline action fit and sampling dispersion, not closed-loop task success.

Table C.2: **Held-out real-robot prefix-20 diagnostics ($H = 30$).** RMSE values are scaled by 10^2. Relative change is CommitFlow versus Base; the three metrics do not share a universal preferred direction.

Metric	Base	CommitFlow	Relative Δ
Mean-action RMSE	6.04	**5.81**	-3.84%
Prediction-spread RMS	4.01	**3.18**	-20.64%
Best-of-four action RMSE	**4.78**	5.01	$+4.83\%$

Under the separate LIBERO same-observation protocol ($H = 10$), Base+IID and the full Prefix-closed+shared/local model generate eight plans from each of 400 fixed observations. Plan-pair RMSE is the mean RMS distance over the 28 unordered sample pairs; prediction-spread RMS is the square root of the mean coordinate-wise sample variance over the eight plans; and action differences are averaged over samples and time on the six continuous arm dimensions. This probe gives a 16.25% decrease in pairwise plan RMSE, a 19.02% decrease in prediction spread, and a 71.38% increase in within-plan second difference. These observed changes are consistent with greater cross-sample concentration without generic action flattening. The $+71.38\%$ same-observation second-difference result is distinct from the $+78$–82% closed-loop range in Table C.3; likewise, the held-out real-robot best-of-four regression shows that lower dispersion need not improve stochastic coverage.

C.4 Action-Variation Trade-offs

For a normalized continuous-action sequence $\tilde{a}_{1:H}$, first difference averages $\|\tilde{a}_{h+1} - \tilde{a}_h\|_2$ over valid h, while second difference averages $\|\tilde{a}_{h+2} - 2\tilde{a}_{h+1} + \tilde{a}_h\|_2$; per-plan values are then averaged over the applicable samples or episodes. The executed first difference applies the first definition to consecutive realized controls across the closed-loop trajectory.

Table C.3: **Complementary action-variation diagnostics from separate protocols.** The first three rows use the same Base/full-CommitFlow checkpoint pair as Table B.1 and summarize 400 matched LIBERO episode pairs over 40 tasks at each applicable $K \in \{1, 5, 10\}$ (400 episodes per method). They are closed-loop ranges and are distinct from the 1.09 same-observation first-difference ratio in Table E.2. The final row is the held-out real-robot prefix-20 diagnostic from Table C.2. Positive changes indicate greater observed variation or error under CommitFlow, as named by each metric.

Metric	Relative change, CommitFlow vs. Base
Within-chunk first difference ℓ_2	$+7.70\%$ to $+9.40\%$
Executed first difference ℓ_2	$+1.10\%$ to $+5.60\%$
Within-chunk second difference ℓ_2	$+78.00\%$ to $+82.00\%$
Held-out best-of-four action RMSE	$+4.83\%$

These countermetrics qualify the consistency result: within-chunk second difference is higher by 81.67%, 78.71%, and 78.46% at $K = 1, 5, 10$, respectively, with BH-adjusted $q = 8.75 \times 10^{-5}$ for each designated primary comparison. The full CommitFlow configuration does not make every action-space notion of smoothness smaller; it combines lower observed cross-plan revision with greater observed within-plan curvature in these measurements, motivating adaptive replanning and explicit responsiveness measurements.

D Implementation, Efficiency, and Further Limitations

D.1 Implementation and Evaluation Parity

Backbone and hyperparameters. All policies use the $\pi_{0.5}$ base architecture (Black et al., 2025a): a 2B-parameter PaliGemma-initialized vision–language module and a 300M-parameter action expert. CommitFlow changes only the stochastic action pathway and token dependencies; matched pairs share the same pretrained backbone initialization, data, training, solver, and evaluation. We pre-specified $\rho = 0.5$ before downstream task-success evaluation as the symmetric design point assigning

equal prior variance to the shared and local components, and held it fixed across benchmarks, seeds, and execution lengths. No validation or test sweep was used to select it; separately registered $\rho \in \{0.25, 0.75\}$ mechanism scans were not used for model selection, and we do not claim $\rho = 0.5$ is optimal. The value is unrelated to the "0.5" in $\pi_{0.5}$.

One-way dependency implementation. The implementation realizes the one-way dependency in Section 3.3 in both modes. ODE sampling prefills the conditioning prefix into a read-only key–value cache and recomputes only the shared-token/action suffix; the layerwise suffix mask prevents the shared-token query from reading action keys and values. The joint training pass applies the corresponding block mask to the full prefix–suffix sequence, preventing either conditioning-prefix or shared-token states from reading action states.

For Block-5, the action suffix uses a two-block lower-triangular mask in both training and sampling. Rows 1–5 may attend to action columns 1–5, while rows 6–10 may attend to action columns 1–10. Conditioning and, when present, the shared token remain one-way sources that do not read action states. Consequently, no direct or layer-mediated path runs from local variables at positions 6–10 to actions 1–5.

Block-5 tail-resampling probe. We evaluate the final 30k-step checkpoint for each context choice and training seed. For every checkpoint, 16 fixed LIBERO observations receive eight tail-local resamples with a 10-step solver and diagnostic seed 3058, yielding 128 records. The observation, positions 1–5, and any shared latent remain fixed while positions 6–10 are resampled. Across all four checkpoints and 512 records, both normalized prefix RMSE and prefix L_∞ deviation are zero at six-decimal reporting precision. This is an offline structural probe, not a task-success measurement.

Evaluation grids. LIBERO evaluation covers all four suites with $H = 10$, $K \in \{1, 5, 10\}$, two training seeds, and paired environment initializations. Each of the three evaluation seeds in Table E.3 contains 1,920 episodes per method over 40 tasks. The RoboTwin clean-domain evaluation fixes $H = 50$, evaluates $K \in \{10, 20, 50\}$ with unseen instructions, and uses 100 shared initializations per task over 50 tasks, yielding 5,000 episodes per method in each configuration.

Factorial ablation protocol. The factorial LIBERO ablation uses the same three LIBERO evaluation seeds as Table E.3, the same 40 tasks, and 5,760 episodes in every success cell. It crosses Base (unrestricted) versus prefix-closed action attention with IID versus shared/local stochastic context while retaining the remaining protocol. The shared/local level combines the correlated prior in equation 4 with the learned shared-latent projection; it is therefore a context component rather than only a change of marginal noise. TRS_8 fixes the observation and local innovations at positions 1–8 and, when present, the shared latent, while resampling only positions 9–10. Thus, the subscript 8 denotes an offline diagnostic prefix length and is distinct from the closed-loop execution settings $K \in \{1, 5, 10\}$. A reported value of 0 means that no deviation was observed at six-decimal reporting precision. All four LIBERO ablation checkpoints use the same observation/resampling protocol and equal record counts; this probe is independent of the $H = 30$ offline probe on real-robot observations in Figure 3. Plan-pair RMSE and within-plan first difference use matched same-observation probes and are normalized to Base+IID; these diagnostic ratios are separate from the 5,760-trial success denominator and from the closed-loop quantities in Table C.3. The success rates disaggregated by closed-loop execution length appear in Table E.1.

Architecture-level latency protocol. The timing experiment isolates compiled device computation rather than checkpoint quality: parameter values are randomly initialized because they do not change tensor shapes or the compiled operator graph. Both methods use the same fixed observation and local-noise tensors; CommitFlow additionally receives a fixed 32-dimensional shared-noise tensor. We measure the full visual/language prefix pass and ten ODE vector-field evaluations at batch size 1, $H = 10$, and BF16 on NVIDIA H200 GPUs (driver 570.195.03; JAX/JAXlib 0.5.3). Each of four independent processes performs 30 warm-up pairs followed by 500 timed pairs in alternating AB/BA order, with `jax.block_until_ready()` after every call. Pooled over 2,000 pairs, Base/CommitFlow median latencies are 35.69/35.84 ms and p95 latencies are 36.03/36.15 ms. A circular moving-block bootstrap within run (block length 10; 20,000 resamples) gives a 95% interval of $[+0.39, +0.42]\%$ for the median ratio. A secondary, exploratory 400-pair run whose timing boundary also includes sampler random-number generation gives a -0.68%

median difference; the direction reversal within 1% motivates only a latency-parity claim. These measurements exclude preprocessing, serialization, network transport, and closed-loop environment time and are not checkpoint-specific end-to-end measurements.

D.2 Further Limitations and Future Evaluation

Non-uniform task-level effects and coupled controls. The higher observed RoboTwin clean-domain aggregate rates do not imply uniform dominance: depending on K, Base has higher observed success on 9–18 of 50 tasks, and 1–5 tasks tie. Individual task deltas are descriptive estimates from 100 shared initializations and should not be over-interpreted without task-level uncertainty and multiplicity control. Moreover, varying K jointly changes the discarded-tail fraction and the frequency of observation-conditioned replanning, so this sweep cannot by itself attribute the observed difference to either factor. A factorial evaluation that changes horizon and execution length independently would separate these effects.

Scope of success-rate uncertainty. The pooled LIBERO factorial analysis accompanying Fig. 4(b) uses suite-stratified paired task-bootstrap intervals, and Table 2 reports the separately matched Block-5 comparison at $K = 5$. The individual-seed LIBERO rows, original per-K factorial decomposition, cross-architecture Block-5 point differences, 40-state tail intervention, RoboTwin comparisons, and online real-robot success counts remain descriptive; no hypothesis tests are claimed for them.

Potential commitment–flexibility trade-off. Appendix Table E.2 shows that the full model has 16.25% lower plan-pair RMSE while its measured within-plan first difference is 8.6% higher; the related, non-independent prediction-spread statistic is retained only as a secondary diagnostic. The lower plan-pair RMSE is therefore not accompanied by a lower measured first difference, arguing against simple flattening under this metric. The full model may nevertheless trade flexibility for commitment: direct measurements of disturbance recovery, adaptation latency, and event-conditioned correction remain future work.

Continuity across replans. The prefix-invariance result is an intra-call counterfactual guarantee: it fixes the complete conditioning input, prefix-local innovations, the shared latent when present, and any other non-tail stochastic inputs, while resampling only discarded tail-local innovations. Consecutive calls may condition on different observations and ordinarily draw new shared and local variables; the architecture therefore imposes no equality or continuity constraint on overlapping actions across calls. The observed reductions in overlap revision and boundary jump (Table B.1) are empirical average differences, not a pointwise or all-horizon guarantee of chunk-to-chunk continuity. Future work may combine prefix closure with adaptive execution lengths, uncertainty- or event-triggered replanning, explicit overlap-consistency objectives, or transport of shared latent state with a learned reset mechanism.

Backbone scope. The invariance result applies to parallel flow policies satisfying equation 6. Empirical evidence is presently limited to a $\pi_{0.5}$-family backbone; other diffusion and vision-language-action architectures remain untested.

Real-robot evidence. The offline $H = 30$ diagnostic and online $H = 50$, $K = 20$ success study are separate protocols. The online study reports one fixed initial configuration per task and descriptive success counts; it does not establish generalization to randomized object placement, disturbance recovery, or safety-critical operation. The camera-limited book task is disclosed separately in Appendix G and is not included in the primary seven-task aggregate.

E LIBERO SUPPLEMENTARY RESULTS

E.1 ABLATION RESULTS DISAGGREGATED BY EXECUTION LENGTH

Table E.1: **Component ablation on LIBERO, disaggregated by execution length K.** Bold marks the highest observed success rate within the original 2×2 factorial configurations in each execution-length column and in the pooled column. Per-K entries are descriptive point estimates; pooled paired intervals are reported in Appendix B.2.

Configuration	$K = 1$	$K = 5$	$K = 10$	Pooled over K
Base+IID	95.68%	94.84%	92.97%	94.50%
Base+shared/local	96.09% (+0.42 pp)	95.31% (+0.47 pp)	93.18% (+0.21 pp)	94.86% (+0.36 pp)
Prefix-closed+IID	95.36% (−0.31 pp)	94.32% (−0.52 pp)	92.29% (−0.68 pp)	93.99% (−0.50 pp)
Prefix-closed+ shared/local (CommitFlow)	**97.29%** (+1.61 pp)	**96.09%** (+1.25 pp)	**93.65%** (+0.68 pp)	**95.68%** (+1.18 **pp**)
Interaction I_K (pp)	+1.51	+1.30	+1.15	+1.32

Note. Each K column contains 1,920 episodes per configuration; the pooled column contains 5,760. Parenthesized values are percentage-point changes relative to Base+IID. With $S_K(\cdot)$ denoting success rate, $I_K = [S_K(CommitFlow) - S_K(\text{Prefix-closed+IID})] - [S_K(\text{Base+shared/local}) - S_K(\text{Base+IID})]$.

Table E.2: **Same-observation LIBERO diagnostics for the matched component ablation.** Ratios are normalized to Base+IID. Lower plan-pair RMSE indicates less cross-sample plan variation, while the first-difference ratio measures adjacent-action variation within a plan.

Configuration	Plan-pair RMSE ratio	First-diff. ratio
Base+IID	1.00	1.00
Base+shared/local	0.91	1.02
Prefix-closed+IID	1.07	0.93
Prefix-closed+shared/local (CommitFlow)	0.84	1.09

E.2 LIBERO EVALUATION-SEED SUCCESS LEDGER

Table E.3: **LIBERO result ledger across three evaluation seeds.** Each evaluation seed pools two training seeds, all four 10-task suites, and $K \in \{1, 5, 10\}$, totaling 1,920 episodes per method over the same 40 tasks. The pooled row contains 5,760 episodes per method.

Evaluation seed	Base success	CommitFlow success	Δ
20260821	94.43%	**95.63%**	+1.20 pp
20260824	94.38%	**95.94%**	+1.56 pp
20260825	94.69%	**95.47%**	+0.78 pp
Pooled	94.50%	**95.68%**	+1.18 pp

The pooled difference corresponds to 68 additional successful episodes for CommitFlow; its paired task-bootstrap interval is reported in the discussion of Fig. 4(a). All 144 Base/CommitFlow manifests represented in Table E.3 are complete and record zero exceptions.

E.3 KNOWN-BOUNDARY BLOCK-5 CHECK

Table E.4: **Seed-level ledger for the LIBERO Block-5 execution-boundary check at $H = 10$ and $K = 5$.** Each configuration pools two training seeds, three evaluation seeds, 40 tasks, and 1,920 episodes (48 episodes per task). Rate intervals are 95% suite-stratified task-bootstrap intervals. Each TRS_5 entry pools two trained checkpoints and 256 tail-resampling records; 0 denotes no observed deviation at six-decimal precision.

Configuration	Seed 6113 rate	Seed 9187 rate	Pooled success	Rate [95% CI]	$\mathrm{TRS}_5 \downarrow$
Block-5+IID	96.88%	96.77%	96.82%	[94.27, 98.59]	0.00
Block-5+shared/local	96.88%	96.46%	96.67%	[94.79, 98.18]	0.00

The paired task-level contrast between Block-5+shared/local and Block-5+IID is -0.16 pp, with a 95% task-bootstrap interval of $[-1.30, 1.04]$ pp. Thus, this experiment detects no additional success benefit from shared/local context under Block-5. Compared with the descriptive $K = 5$ entries in Table E.1, Block-5+IID is $+2.50$ pp above Prefix-closed+IID and $+1.98$ pp above Base+IID, while Block-5+shared/local is $+0.57$ pp above Prefix-closed+shared/local. These cross-architecture comparisons are reported as point estimates only because the earlier $K = 5$ task-level vectors required for paired bootstrap contrasts are unavailable.

F ROBOTWIN CLEAN-DOMAIN SUPPLEMENTARY RESULTS

F.1 CLEAN-DOMAIN PROTOCOL LEDGER

Table F.1: **RoboTwin clean-domain configuration ledger with exact evaluation RNG seeds.** Evaluation uses all 50 tasks in `demo_clean` with `instruction_type=unseen`; `demo_randomized` is not evaluated. The Primary, Replicate A, and Replicate B evaluations correspond to seeds 0, 1, and 2, respectively. Each row contains 5,000 episodes per method, with generated horizon $H = 50$.

Eval seed	K	Discarded tail	Base	CommitFlow	Δ
0	10	80.00%	64.94%	**68.86%**	+3.92 pp
0	20	60.00%	67.72%	**71.34%**	+3.62 pp
0	50	0.00%	59.50%	**62.28%**	+2.78 pp
1	10	80.00%	65.38%	**68.94%**	+3.56 pp
2	10	80.00%	65.28%	**69.36%**	+4.08 pp

F.2 COMPLETE TASK-LEVEL RESULTS

Table F.2: **Complete task-level RoboTwin clean-domain success (Primary evaluation; seed 0).** Each task uses 100 shared initializations under unseen instructions. Cells report Base / CommitFlow / Δ success counts at fixed $H = 50$; because every task has 100 trials, each count also equals its success percentage. Bold marks the higher value.

Task	$K = 10$ B/CF/Δ	$K = 20$ B/CF/Δ	$K = 50$ B/CF/Δ
`adjust_bottle`	38/**40**/+2	76/**85**/+9	90/**99**/+9
`beat_block_hammer`	**91**/86/−5	69/**76**/+7	86/**88**/+2
`blocks_ranking_rgb`	67/**69**/+2	62/**63**/+1	**55**/53/−2
`blocks_ranking_size`	**40**/28/−12	**38**/26/−12	**16**/10/−6
`click_alarmclock`	37/**42**/+5	78/**79**/+1	**72**/55/−17
`click_bell`	**39**/33/−6	**80**/75/−5	**78**/58/−20
`dump_bin_bigbin`	60/**70**/+10	53/**69**/+16	**59**/53/−6

Continued on next page

Table F.2 continued

Task	$K = 10$ B/CF/Δ	$K = 20$ B/CF/Δ	$K = 50$ B/CF/Δ
grab_roller	88/**98**/+10	93/**100**/+7	95/**99**/+4
handover_block	36/**54**/+18	9/**18**/+9	**11**/3/−8
handover_mic	96/96/0	96/**99**/+3	88/**93**/+5
hanging_mug	5/**19**/+14	4/**8**/+4	4/**8**/+4
lift_pot	35/**49**/+14	26/**50**/+24	**15**/10/−5
move_can_pot	**85**/81/−4	76/**78**/+2	**61**/58/−3
move_pillbottle_pad	65/**84**/+19	76/**89**/+13	33/**62**/+29
move_playingcard_away	74/**78**/+4	77/**79**/+2	**83**/77/−6
move_stapler_pad	56/56/0	28/**36**/+8	**19**/17/−2
open_laptop	89/**90**/+1	92/**93**/+1	88/**93**/+5
open_microwave	**71**/14/−57	**87**/44/−43	**89**/83/−6
pick_diverse_bottles	67/**77**/+10	69/**76**/+7	43/**61**/+18
pick_dual_bottles	66/**73**/+7	61/**70**/+9	29/**47**/+18
place_a2b_left	43/**71**/+28	53/**72**/+19	54/**59**/+5
place_a2b_right	51/**66**/+15	54/**66**/+12	47/**59**/+12
place_bread_basket	87/87/0	79/**89**/+10	67/**72**/+5
place_bread_skillet	90/**91**/+1	86/**94**/+8	75/**84**/+9
place_burger_fries	89/**98**/+9	88/**93**/+5	43/43/0
place_can_basket	**62**/47/−15	**67**/62/−5	**44**/39/−5
place_cans_plasticbox	67/**69**/+2	**85**/60/−25	65/**76**/+11
place_container_plate	88/**90**/+2	92/92/0	**94**/93/−1
place_dual_shoes	63/**69**/+6	59/**74**/+15	**21**/19/−2
place_empty_cup	98/**100**/+2	96/**99**/+3	94/**97**/+3
place_fan	16/**40**/+24	26/**45**/+19	38/**41**/+3
place_mouse_pad	54/54/0	48/**52**/+4	23/**31**/+8
place_object_basket	57/**69**/+12	**72**/65/−7	54/**63**/+9
place_object_scale	91/**92**/+1	87/87/0	74/**80**/+6
place_object_stand	94/**98**/+4	94/**96**/+2	84/**86**/+2
place_phone_stand	52/**58**/+6	54/**61**/+7	**62**/56/−6
place_shoe	84/**91**/+7	86/**91**/+5	75/**83**/+8
press_stapler	**70**/62/−8	**91**/87/−4	**90**/87/−3
put_bottles_dustbin	10/**37**/+27	49/**50**/+1	26/**52**/+26
put_object_cabinet	66/**76**/+10	**80**/77/−3	76/**84**/+8
rotate_qrcode	34/**68**/+34	68/**83**/+15	81/**83**/+2
scan_object	67/**70**/+3	65/**68**/+3	45/**49**/+4
shake_bottle	96/**99**/+3	97/**100**/+3	98/**99**/+1
shake_bottle_horizontally	96/**98**/+2	98/**100**/+2	98/**99**/+1
stack_blocks_three	**54**/41/−13	40/40/0	18/**36**/+18
stack_blocks_two	93/**98**/+5	84/**94**/+10	76/**86**/+10
stack_bowls_three	76/76/0	62/**72**/+10	**70**/68/−2
stack_bowls_two	95/**96**/+1	88/**95**/+7	95/**97**/+2
stamp_seal	53/**57**/+4	**44**/41/−3	24/**34**/+10
turn_switch	**46**/38/−8	44/**49**/+5	**50**/32/−18
All tasks	64.94/**68.86**/+3.92	67.72/**71.34**/+3.62	59.50/**62.28**/+2.78
W/T/L	36 / 5 / 9	38 / 3 / 9	31 / 1 / 18
Median Δ [Q1,Q3]	+3.00 [0.00, +10.00]	+4.00 [+1.00, +9.00]	+2.50 [−2.75, +8.00]

F.3 TASK-EFFECT DECOMPOSITION

Table F.3: **RoboTwin clean-domain task-effect decomposition at evaluation seed 0.** W/T/L denotes wins, ties, and losses. Consistent signs occur for 21 positive and 5 negative tasks; 15 flip sign and 9 include a tie. Delta correlations for $(K = 10, 20)/(10, 50)/(20, 50)$ are 0.72/0.33/0.11.

K	W/T/L	Positive mass	Negative mass	Net / Δ
10	36/5/9	$+324$	-128	$+196$ / $+3.92$ pp
20	38/3/9	$+288$	-107	$+181$ / $+3.62$ pp
50	31/1/18	$+257$	-118	$+139$ / $+2.78$ pp

G ONLINE REAL-ROBOT EVALUATION

G.1 PROTOCOL AND COMPLETE TASK COUNTS

The online evaluation uses an AgiBot G1 and a matched Base/strict-CommitFlow checkpoint pair that is separate from the $H = 30$ offline-diagnostic pair. Each policy call predicts $H = 50$ actions, executes the first $K = 20$, and then replans. For every task and method, the robot performs 50 rollouts and is reset to the same task-specific initial configuration after each rollout. The seven-task primary aggregate therefore contains 350 rollouts per method. Table G.1 reports every attempted task.

Table G.1: **Complete online AgiBot G1 task results.** "Full success" requires completion of the entire task. The book task is excluded from the primary aggregate because the collected head-camera observations lose the right hand during the grasp phase; its stage-completion count is reported separately below.

Task	Arm	Base	CommitFlow	Primary aggregate
Pen	Left	86.00%	**90.00%**	Yes
Correction fluid	Left	80.00%	**86.00%**	Yes
Pencil case	Left	70.00%	**72.00%**	Yes
Stapler	Right	92.00%	**94.00%**	Yes
Charger	Right	92.00%	92.00%	Yes
Eraser	Right	90.00%	**94.00%**	Yes
Trash	Right	100.00%	100.00%	Yes
Book: full task	Right	0.00%	0.00%	No
Seven-task aggregate	–	87.14%	**89.71%**	–

The book task requires a push, grasp, and placement sequence. During data collection, the head camera loses visibility of the right hand at the grasp phase. Neither policy completes the full sequence; both complete the initial push stage in 50.00% of trials. For completeness, counting the book rollouts as full-task failures gives 76.25% for Base and 78.50% for CommitFlow, while the primary comparison excludes this camera-limited task for both methods.

Video supplement. All clips are outcome-stratified qualitative selections and do not estimate success rates. For six of the seven fully evaluable real-robot tasks, the supplement contains one success and one failure per method. Trash contains one success per method because no failure was observed. Book contains one partial rollout per method. The partial label denotes intermediate-stage completion rather than full-task success. For simulation, the supplement contains one success and one failure per method for task 01 from LIBERO-10 and LIBERO-Goal. These clips provide illustrative subset coverage rather than benchmark-wide samples. All quantitative tables use the complete evaluation protocols and are not computed from the displayed videos.