SEAM: SMOOTH EXECUTION OF ACTION-CHUNKED MOTION FOR VISION-LANGUAGE-ACTION POLICIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-Language-Action (VLA) policies generate locally coherent action chunks, yet consecutive predictions can disagree in their overlapping future actions, introducing abrupt changes at execution boundaries. We propose **SEAM** (**S**mooth **E**xecution of **A**ction-chunked **M**otion), a training-free inference method for flow matching VLAs. SEAM uses the previous chunk's unexecuted tail as a temporally aligned consistency reference. Its core mechanism, *Velocity-guided Loss Steering* (VLS), alternates learned velocity-field steps with closed-form corrections toward a time-dependent mean reference. The corrected state feeds subsequent velocity evaluations, coupling cross-chunk consistency with observation-conditioned generation without policy-network backpropagation. In a 1,300-episode comparison on LIBERO-10 with $\pi_{0.5}$, SEAM reduces boundary action second differences by 28% and chunk transition discontinuity by 27% at near-baseline denoising-loop cost. Matched-state controls show that the aligned reference improves the continuity–success trade-off over zero or shuffled references. With SmolVLA on LIBERO-Spatial, the same guidance setting reaches 72.0% success versus 65.6% for baseline and 60.0% for RTC, with respective realized end-effector jerk reductions of 58.8% and 40.0%, at approximately 1% additional full-policy execution cost.

1 INTRODUCTION

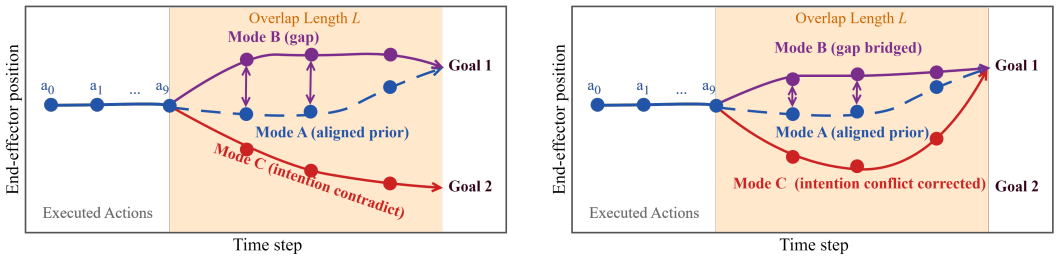
Vision-Language-Action (VLA) policies have become central to language-conditioned robotic manipulation, from RT-1 and RT-2 to OpenVLA, π_0, and $\pi_{0.5}$ (Brohan et al., 2022; Zitkovich et al., 2023; Kim et al., 2024; Black et al., 2024; Physical Intelligence et al., 2025). Given visual observations and a natural-language instruction, these policies generate robot action trajectories that must remain task-directed and physically smooth. Recent flow matching VLAs such as $\pi_{0.5}$ (Physical Intelligence et al., 2025) model this generation process as an ODE that transports Gaussian noise into an action trajectory.

A key design choice is *action chunking* (Zhao et al., 2023; Chi et al., 2025): the policy predicts H future actions, executes the first K, and then replans. Each chunk forms a locally coherent motion plan, while consecutive predictions overlap over $H-K$ steps. The new prediction should connect to the previous plan's unexecuted tail. Independent sampling and updated observations can nevertheless produce disagreement in this overlap, introducing abrupt changes at execution boundaries.

Chunk disagreement reflects both independent sampling and changes in the observation between queries. One possible source is a change between feasible local strategies: contact-rich manipulation often admits several approaches (Shafiullah et al., 2022), and consecutive samples can choose different continuations even under nearby observations (Wang, 2026; Black et al., 2025a). Figure 1a illustrates a direction reversal and a position gap at a chunk transition. We target the resulting mismatch in overlapping action predictions, using the previous plan as a reference while generating from the current observation.

Existing approaches introduce continuity at different stages. Temporal ensembling (Zhao et al., 2023) averages overlapping outputs; RTC (Black et al., 2025a) applies inference-time inpainting gradients through the policy; and BID (Liu et al., 2025) selects consistent samples. Training-time RTC and Legato learn continuation through action conditioning (Black et al., 2025b; Liu et al., 2026), while A2C2 adds a lightweight per-step correction head to a frozen base policy (Sendai et al.,

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.



(a) Chunk Boundary Artifacts (b) SEAM: VLS-Guided Overlap Steering

Figure 1: SEAM overview. (a) Schematic examples of chunk-boundary mismatch: a direction reversal (*Mode C*) and a position gap (*Mode B*), where the next chunk preserves direction but starts from an offset. Goals 1 and 2 denote two feasible local continuations. (b) SEAM steers the overlap window toward the aligned prior, the unexecuted tail of the previous chunk, during denoising without backpropagating through the policy.

2025). These approaches motivate studying where a consistency constraint can enter the generation process and what computation it requires.

We focus on a closed-form correction inside an existing flow solver. In synchronous execution, the robot consumes K actions before requesting the next chunk, so the unexecuted tail directly specifies a temporally aligned reference. Using this reference during integration introduces continuity while retaining the pretrained policy and its observation-conditioned velocity field.

We propose **SEAM** (**S**mooth **E**xecution of **A**ction-chunked **M**otion), a training-free inference-time method whose core mechanism is **Velocity-guided Loss Steering (VLS)**. SEAM uses the previous prediction's $L{=}H{-}K$ tail as an analytic cross-chunk consistency reference. VLS constructs a time-interpolated consistency target from this reference and applies a closed-form correction after each Euler step of the flow matching ODE without backpropagating through the policy network. On LIBERO-10 with $\pi_{0.5}$, the baseline boundary jerk is more than twice its interior jerk (0.194 vs. 0.094), indicating that command variation is concentrated at chunk boundaries. SEAM reduces boundary jerk by 28% and chunk transition discontinuity by 27% at $1.01\times$ denoising-loop cost, while preserving baseline-level task success (95.7% vs. 94.8%). RTC reaches lower boundary jerk and chunk discontinuity with 95.1% success, but its policy-network gradients raise denoising-loop cost to $1.22\times$ the baseline; ACT-TE reaches lower motion metrics at substantially lower task success. These comparisons highlight SEAM's intended operating regime: improving cross-chunk action continuity while keeping denoising-loop overhead small and preserving task-directed policy predictions.

In summary, our key contributions are as follows:

- We cast cross-chunk consistency in synchronous chunked execution as an analytic overlap-alignment objective and propose SEAM, a training-free inference-time method to mitigate this artifact.
- We design VLS, the core mechanism of SEAM, as a closed-form correction that steers the denoising ODE toward an aligned prior without backpropagating through the policy network, reducing cross-chunk inconsistency at execution boundaries.
- We evaluate two flow matching backbones across four LIBERO suites, using matched-state controls and varied execution lengths and delivery delays to characterize task success, command continuity, and realized end-effector motion at low inference overhead.

2 Related Work

2.1 Generalist, Chunked, and Generative Robot Policies

Generalist robot policies connect visual observations, language instructions, and actions, including RT-1, RT-2, OpenVLA, Octo, π_0, $\pi_{0.5}$, and LingBot-VLA (Brohan et al., 2022; Zitkovich et al., 2023; Kim et al., 2024; Octo Model Team et al., 2024; Black et al., 2024; Physical Intelligence et al.,

2025; Wu et al., 2026). Action chunking provides locally coherent predictions (Zhao et al., 2023; Chi et al., 2025), while diffusion and flow matching represent multiple feasible continuations (Wen et al., 2025b;a; Jia et al., 2026; Yashima et al., 2026). SEAM addresses consistency between successive predictions by conditioning inference on the unexecuted previous plan, keeping the pretrained backbone and training data unchanged.

2.2 Cross-Chunk Consistency and Inference-Time Steering

Cross-chunk methods differ in how they constrain the next prediction. Temporal ensembling averages overlapping outputs. RTC (Black et al., 2025a) uses ΠGDM-style posterior guidance and backpropagates through the policy during denoising. BID samples multiple candidates and selects using backward coherence and forward contrast (Liu et al., 2025). Past-Token Prediction trains policies to predict past actions alongside future ones and uses past-action consistency to select candidates at inference (Villasevil et al., 2025). Legato and FASTER change policy training or action generation (Liu et al., 2026; Lu et al., 2026). More broadly, classifier and classifier-free guidance steer generative sampling (Dhariwal & Nichol, 2021; Ho & Salimans, 2022), while inverse-problem methods introduce measurement constraints (Meng et al., 2021; Lugmayr et al., 2022; Kawar et al., 2022; Chung et al., 2022). SEAM shares the overlap-conditioning principle of inpainting. Its reference is the mean forward interpolation of the previous action tail, and its soft replacement follows from a quadratic consistency loss on the Euler candidate. This form provides a closed-form correction within an existing flow solver.

Recent methods also address the interaction between continuity and delayed observations. A2C2 corrects each executed action using the latest observation and a lightweight correction head (Sendai et al., 2025). POTR changes RTC's guidance weight and constrains its component orthogonal to the denoising velocity (Fang et al., 2026). Training-time RTC removes inference-time inpainting overhead by learning action-prefix conditioning (Black et al., 2025b), and REMAC learns masked-action corrections with prefix-preserved sampling (Wang et al., 2026). VLASH uses a predicted future execution-time state for asynchronous inference (Tang et al., 2026). SEAM focuses on synchronous overlap consistency with a fixed pretrained policy, an analytical mean reference, and no auxiliary learned component. Its efficiency comparison with RTC concerns inference-time gradient guidance; training-time conditioning provides a different route to low-overhead continuation.

Noise-space attribution experiments show that changing the latent under fixed observations can alter boundary artifacts, and that the relation between artifact magnitude and task outcome depends on the execution context (Wang, 2026). This motivates evaluating action continuity together with task success and examining the contribution of the reference content.

3 Preliminaries

3.1 Flow Matching for Action Generation

Flow matching (Lipman et al., 2022; Tong et al., 2023) formulates generative modeling as learning a velocity field v_θ that transports samples from a noise distribution to the data distribution via an ODE. Given a clean action chunk $\mathbf{x}^{\text{data}} \sim p_{\text{data}}(\cdot \mid \mathbf{o})$ and noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, the forward interpolation defines:

$$\mathbf{x}_t = (1-t)\,\mathbf{x}^{\text{data}} + t\,\boldsymbol{\epsilon}, \quad t \in [0,1] \tag{1}$$

with the conditional velocity field $\mathbf{u}_t = \boldsymbol{\epsilon} - \mathbf{x}^{\text{data}}$. The model $v_\theta(\mathbf{x}_t, t, \mathbf{o})$ is trained to approximate $\mathbf{u}_t$ by minimizing $\|v_\theta(\mathbf{x}_t, t, \mathbf{o}) - \mathbf{u}_t\|^2$. At inference, actions are generated by sampling $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and solving the ODE backward from $t{=}1$ to $t{=}0$ using Euler integration:

$$\mathbf{x}_{t+\Delta t} = \mathbf{x}_t + \Delta t \cdot v_\theta(\mathbf{x}_t, t, \mathbf{o}), \quad \Delta t = -\frac{1}{N} \tag{2}$$

where N is the number of integration steps. The final state $\mathbf{x}_{t=0}$ yields the predicted action sequence.

3.2 Action Chunking and Overlap

An action-chunked policy π_θ generates a chunk $\mathbf{c} \in \mathbb{R}^{H \times D}$ of H future actions at each prediction step, where D is the action dimension. Only the first K actions are executed before re-prediction,

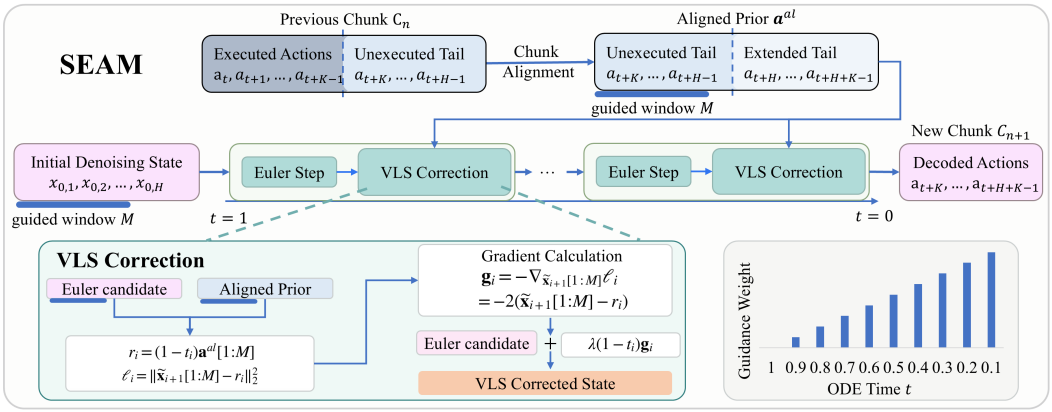


Figure 2: Overview of SEAM. The previous decoded chunk $\mathbf{c}_n$ is split into an executed prefix and an unexecuted tail. SEAM extends this tail into the aligned prior $\mathbf{a}^{\text{al}}$ for the next chunk $\mathbf{c}_{n+1}$. Starting from a standard-Gaussian denoising state, the policy computes an Euler candidate and VLS corrects its guided window M toward a time-scaled reference. The reference and correction weight $\lambda(1-t_i)$ use the current solver time t_i. The corrected state enters the next velocity-field evaluation.

creating an *overlap region* of length $L = H - K$ between consecutive chunks. After executing $\mathbf{c}_n[1{:}K]$, temporal consistency corresponds to $\mathbf{c}_{n+1}[1{:}L] \approx \mathbf{c}_n[K{+}1{:}H]$. Each new chunk combines an independent noise sample $\mathbf{z}_{n+1} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ with an updated observation, allowing its overlap prediction to change.

Stochastic continuation. The conditional action distribution can represent multiple feasible continuations. Independent noise samples can produce different trajectories, while each new observation also supplies information needed to revise the previous plan. An overlap reference therefore provides temporal continuity alongside the policy's observation-conditioned generation.

4 SEAM: Smooth Execution of Action-Chunked Motion

4.1 Overview

SEAM is a plug-and-play inference method for flow matching VLAs. Given the previous chunk's unexecuted tail, it constructs an aligned prior and applies closed-form VLS corrections to the overlap prefix at every Euler step, without retraining the policy, adding auxiliary networks, or backpropagating through the policy at inference time. We use $\mathbf{c}$ for decoded action chunks, $\mathbf{a}$ for executed actions or action-space references, and $\mathbf{x}_i$ for the discrete denoising solver state. We reserve subscript t for continuous ODE time and i for solver iterations. Reference construction and VLS operate in the policy's normalized action coordinates; predictions are unnormalized before execution. Figure 2 shows the complete pipeline.

4.2 Aligned Prior Construction

Given the previous decoded chunk $\mathbf{c}_n$, we define the unexecuted tail as $\mathbf{a}^{\text{tail}} = \mathbf{c}_n[K{+}1{:}H] \in \mathbb{R}^{L \times D}$, where $L = H - K$. We extend this tail to chunk length H by repeating its last entry, producing the *aligned prior*:

$$\mathbf{a}^{\text{al}} = \text{Extend}(\mathbf{a}^{\text{tail}}, H) \in \mathbb{R}^{H \times D}. \tag{3}$$

The aligned prior is the cross-chunk state used by VLS. The guided prefix $M \leq L$ contains actions for the same future execution times as the new chunk's first M positions. The next chunk is initialised from standard Gaussian noise $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})^{H \times D}$, and the prior enters through the denoising corrections.

Algorithm 1 SEAM inference for one chunk query.

Input: observation $\mathbf{o}$, previous chunk tail $\mathbf{a}^{\text{tail}}$, policy velocity field v_θ, horizon H, execution length K, guided window M, strength λ, and ODE steps N.

1: If a previous tail exists, build $\mathbf{a}^{\text{al}} = \text{Extend}(\mathbf{a}^{\text{tail}}, H)$.
2: Sample $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})^{H\times D}$ and set $t_i = 1 - i/N$.
3: **for** $i = 0, \ldots, N-1$ **do**
4: Compute $\widetilde{\mathbf{x}}_{i+1} = \mathbf{x}_i + (t_{i+1} - t_i) v_\theta(\mathbf{x}_i, t_i, \mathbf{o})$.
5: Set $\mathbf{x}_{i+1} = \widetilde{\mathbf{x}}_{i+1}$.
6: **if** a previous tail exists **then**
7: Form $\mathbf{r}_i = (1 - t_i)\mathbf{a}^{\text{al}}[1{:}M]$.
8: Apply Eq. 8 to the first M positions.
9: **end if**
10: **end for**
11: Decode $\mathbf{x}_N$ as the action prediction and execute its first K actions.

Repeating the last tail action makes the reference length-compatible with the model output. Because $M \leq L$, the correction uses the original tail entries; the padded entries lie outside the guided window.

4.3 Velocity-Guided Loss Steering (VLS)

VLS alternates a learned velocity-field step with a local consistency correction. Let $M = \min(L, L_{\max})$ be the guided overlap length, $t_i = 1 - i/N$ the reverse Euler schedule, and $\mathbf{x}_i$ the state before step i. To define the reference at each solver time, we take the mean of the forward interpolation in Eq. 1, conditional on the aligned tail and averaging over $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})^{M\times D}$:

$$\mathbf{r}_i = \mathbb{E}_{\boldsymbol{\epsilon}}\big[(1 - t_i)\mathbf{a}^{\text{al}}[1{:}M] + t_i\boldsymbol{\epsilon} \mid \mathbf{a}^{\text{al}}\big] = (1 - t_i)\mathbf{a}^{\text{al}}[1{:}M]. \tag{4}$$

The mean reference carries the previous plan's temporal structure without introducing an additional random draw into the target. Under the quadratic consistency objective, it gives the exact gradient of the loss averaged over independent reference noise (Appendix A). Generation retains its Gaussian initial state and observation-conditioned velocity field. The reference is evaluated at t_i and applied to the post-Euler candidate, giving an explicit alternating update. At each ODE step, we first compute the standard Euler candidate

$$\widetilde{\mathbf{x}}_{i+1} = \mathbf{x}_i + (t_{i+1} - t_i)\, v_\theta(\mathbf{x}_i, t_i, \mathbf{o}). \tag{5}$$

We define a quadratic consistency objective on the candidate's guided window,

$$\ell_i = \|\widetilde{\mathbf{x}}_{i+1}[1{:}M] - \mathbf{r}_i\|_2^2, \tag{6}$$

whose gradient with respect to the candidate is available in closed form. VLS applies this correction to the guided window:

$$\mathbf{g}_i = -\nabla\ell_i = -2\big(\widetilde{\mathbf{x}}_{i+1}[1{:}M] - \mathbf{r}_i\big), \tag{7}$$

$$\mathbf{x}_{i+1}[1{:}M] = \widetilde{\mathbf{x}}_{i+1}[1{:}M] + \lambda(1 - t_i)\cdot\mathbf{g}_i. \tag{8}$$

Unguided positions keep their Euler-candidate values. Writing $\alpha_i = 2\lambda(1 - t_i)$ makes the correction's linear form explicit:

$$\mathbf{x}_{i+1}[1{:}M] = (1 - \alpha_i)\widetilde{\mathbf{x}}_{i+1}[1{:}M] + \alpha_i\mathbf{r}_i. \tag{9}$$

For $0 \leq \alpha_i \leq 1$, this is a convex blend that reduces the candidate's deviation from the current reference by a factor $1 - \alpha_i$: $\mathbf{x}_{i+1}[1{:}M] - \mathbf{r}_i = (1 - \alpha_i)(\widetilde{\mathbf{x}}_{i+1}[1{:}M] - \mathbf{r}_i)$. The reference selects a temporally aligned continuation of the previous chunk, and the increasing $(1-t_i)$ weight strengthens consistency as denoising progresses. The learned velocity field generates each candidate and is evaluated again at the corrected state on the next step. This recurrent coupling is the role of the velocity field in VLS; the correction itself needs no network derivative. Algorithm 1 gives the complete procedure.

Table 1: Main results on LIBERO-10, with 130 episodes per task and 1,300 episodes per method. BJ and IJ denote boundary and interior action second differences, CD measures chunk transition discontinuity, and BJV is the variance of boundary second-difference norms. Extra and Step are cached-prefix denoising-loop milliseconds per Euler step; Chunk is milliseconds per ten-step query. Exec. is relative denoising compute per ten executed actions, accounting for ACT-TE's ten queries versus one for the other methods.

Method	**Execution quality**					**Denoising-loop cost**			
	Succ.%↑	**BJ↓**	**IJ↓**	**CD↓**	**BJV↓**	**Extra↓**	**Step↓**	**Chunk↓**	**Exec.↓**
$\pi_{0.5}$ (Baseline)	94.8	0.194	0.094	0.172	0.165	0.000	28.225	282.2	1.00×
+ ACT-TE	82.7	**0.031**	**0.031**	**0.062**	**0.006**	0.000	28.225	282.2	10.00×
+ RTC	95.1	0.090	0.075	0.089	0.094	6.236	34.460	344.6	1.22×
+ SEAM	**95.7**	0.141	0.074	0.126	0.094	0.371	28.595	286.0	1.01×

Design choices and efficiency. VLS exposes two local design choices: which action dimensions to guide and how much of the overlap to constrain. It can be applied to any subset of action dimensions and over a bounded prefix of the overlap window ($M \leq L$), allowing the method to trade correction footprint against policy freedom. This locality is also what keeps VLS inexpensive. For a guided dimension set of size D_g, each ODE step adds only slicing, subtraction, scalar weighting, and an overwrite on an $M \times D_g$ block, giving $\mathcal{O}(N \cdot M \cdot D_g)$ additional scalar operations per chunk query. Because the consistency gradient in Eq. 7 is computed in closed form from the post-Euler candidate and the aligned prior, VLS does not require a policy-network backward pass or activation storage for gradient computation. Concrete hyperparameter choices and measured denoising-loop costs are given in the experimental setup.

5 Experiments

5.1 Experimental Setup

Benchmarks and policies. We evaluate on LIBERO-10, Spatial, Goal, and Object (Liu et al., 2023), using $\pi_{0.5}$ (Physical Intelligence et al., 2025) as the primary backbone and SmolVLA (Shukor et al., 2025) for cross-backbone evaluation. All methods use the same pretrained checkpoint within each backbone.

Inference settings. Unless otherwise stated, policies predict $H = 50$ actions with ten Euler steps and execute $K = 10$ actions per query. SEAM uses $\lambda = 0.1$, guided window $M = 20$, all action dimensions, and standard Gaussian initialization. We compare against the unmodified policy, RTC, and ACT-TE; ACT-TE queries every control step. Appendix I gives checkpoint and baseline configurations.

Evaluation. Each task uses 130 episodes, with matched initial states across methods in the controlled comparisons. We report success, action-continuity metrics (BJ, IJ, CD, and BJV), and realized end-effector jerk, averaging tasks equally. Appendix I.3 defines the metrics. Timing uses an RTX 3090 at batch size one: Table 1 reports cached-prefix denoising cost, and Appendix J reports the full policy pipeline.

5.2 Main Results

Table 1 compares SEAM, baseline $\pi_{0.5}$, ACT-TE, and RTC with the same checkpoint, prediction horizon, and Euler-step count. SEAM reduces baseline BJ, IJ, and CD by 27.7%, 21.2%, and 26.7%, respectively, with 95.7% versus 94.8% success. It adds 3.7 ms per ten-step denoising query (1.01× baseline cost).

RTC reaches lower boundary metrics with 95.1% success at 1.22× per-query cost. ACT-TE further lowers boundary metrics, with 82.7% success and ten queries per ten actions. Figure 4 visualizes

Table 2: Cross-suite and cross-backbone comparison, with 130 episodes per task. Success is in percent; EEF jerk is in $\mathrm{m/s^3}$. Best values within each row are bold. Full command metrics appear in Tables 14 and 15.

		Baseline		RTC		SEAM	
Backbone	Suite	Success↑	EEF jerk↓	Success↑	EEF jerk↓	Success↑	EEF jerk↓
$\pi_{0.5}$	Spatial	**98.0**	4.027	96.4	3.471	96.8	**3.315**
	Goal	**99.2**	4.060	96.8	3.644	97.2	**3.493**
	Object	99.2	3.134	**99.6**	**2.670**	**99.6**	2.671
SmolVLA	Spatial	65.6	9.070	60.0	6.228	**72.0**	**3.738**
	LIBERO-10	**45.6**	6.497	36.4	4.413	34.0	**2.517**
	Goal	76.8	7.469	70.8	5.396	**78.4**	**3.454**
	Object	**88.8**	6.643	83.6	5.035	**88.8**	**3.174**

Table 3: Reference-content and solver-placement comparisons on matched LIBERO-10 initial states, with 130 episodes per task and 1,300 episodes per row. Clean IJ excludes boundary-crossing action triples; EEF jerk is in $\mathrm{m/s^3}$.

Method	Success (%) ↑	BJ ↓	CD ↓	Clean IJ ↓	EEF jerk ↓
Baseline	95.6	0.193	0.174	0.080	2.847
RTC	94.8	0.089	0.088	0.079	2.386
Shared initial noise	93.2	0.209	0.185	0.081	2.873
Zero reference	67.6	0.230	0.200	0.064	2.208
Shuffled reference	88.0	0.196	0.171	0.069	2.357
Noisy reference	95.6	0.186	0.157	0.105	2.766
Output blend	90.4	0.153	0.148	0.063	2.346
SEAM ($\lambda = 0.1$)	94.8	0.153	0.139	0.066	2.310
SEAM ($\lambda = 0.15$)	93.6	0.136	0.123	0.062	2.175

this continuity–success–compute trade-off. SEAM's largest task-level gain occurs on T1 (91.5% to 99.2%; Table 11), illustrated in Figure 3.

5.3 Generalization across Suites and Backbones

Table 2 compares baseline, RTC, and SEAM across LIBERO suites and two flow-matching backbones, using the same $H = 50$, $K = 10$, ten Euler steps, and SEAM guidance settings. With $\pi_{0.5}$, SEAM reduces baseline EEF jerk by 14.0–17.7% across Spatial, Goal, and Object. It also lowers EEF jerk relative to RTC on Spatial and Goal; both methods achieve 99.6% success with nearly identical jerk on Object.

With SmolVLA, SEAM improves success on Spatial and Goal, matches baseline on Object, and reduces baseline EEF jerk by 52.2–61.3% across all four suites. On LIBERO-10, lower jerk accompanies lower success (34.0% versus 45.6% for baseline). The full-policy cost is $1.009\times$ baseline per ten actions (Appendix E).

5.4 Reference Content and Solver Placement

We use matched initial states on all ten LIBERO-10 tasks to isolate reference content and correction placement (Table 3). Zero, shuffled, and noisy references test the contribution of a temporally aligned mean reference. Shared initial noise tests whether reducing sampling variability suffices. An output-blending control applies the reference once after denoising, using a weight derived from the VLS schedule (Appendix A.1). All settings retain $H = 50$, $K = 10$, and ten Euler steps; Appendix I specifies the controls. Clean IJ excludes both action triples crossing a chunk boundary.

SEAM reduces baseline BJ, CD, and EEF jerk by 20.9%, 19.9%, and 18.8%, respectively, with 94.8% versus 95.6% success. SEAM and RTC both record 94.8% success in the matched-state

Table 4: Fixed-delay execution on LIBERO-10, with 130 episodes per task. Success is in percent; EEF jerk is in $\mathrm{m/s^3}$. SEAM uses skip-prefix guidance. Best values within each row are bold.

	Baseline		RTC		SEAM	
Delay (ms)	Success↑	EEF jerk↓	Success↑	EEF jerk↓	Success↑	EEF jerk↓
100	95.2	2.852	93.6	2.376	**97.2**	**2.348**
200	93.6	2.882	92.4	2.452	**94.0**	**2.407**
300	88.8	2.999	**91.2**	**2.522**	89.6	2.527

Table 5: Guidance sensitivity on LIBERO-10, with 130 episodes per task. (a) Strength and guided dimensions at $M = 20$. (b) Window length with all dimensions and $\lambda = 0.1$. Best guided values in each panel are bold. Full results appear in Tables 9 and 10.

(a) Strength and dimensions

	Baseline	All dimensions				Position only			
λ	–	0.05	0.10	0.15	0.20	0.05	0.10	0.15	0.20
Succ.%↑	94.8	94.8	**95.7**	92.8	89.5	94.3	94.0	92.6	88.2
BJ↓	0.194	0.162	0.141	**0.140**	0.147	0.179	0.162	0.166	0.168
CD↓	0.172	0.145	**0.126**	0.127	0.135	0.159	0.143	0.145	0.152

(b) Window length

M	Baseline	2	4	6	8	10	12	14	16	18	20
Succ.%↑	94.8	95.5	94.8	**96.3**	95.0	95.9	95.1	95.7	94.7	95.5	95.7
BJ↓	0.194	0.148	0.156	0.147	0.149	0.148	0.153	0.154	0.147	0.148	**0.141**
CD↓	0.172	0.142	0.146	0.137	0.138	0.135	0.140	0.141	0.134	0.135	**0.126**

comparison, with SEAM yielding 3.2% lower mean end-effector jerk. Increasing λ to 0.15 further reduces variation with 93.6% success. Shared initial noise has lower success and higher BJ and CD than SEAM.

Reference correspondence matters: zero and shuffled references yield 67.6% and 88.0% success, with worse BJ and CD than the aligned tail. Their lower temporal command variance (Table 12) therefore does not imply better continuity. The mean reference also improves BJ, CD, and EEF jerk over the noisy reference, with 94.8% versus 95.6% success.

Final-output blending at weight 0.4300 reaches 90.4% success, compared with 94.8% for SEAM, with CD 0.148 versus 0.139 and EEF jerk 2.346 versus 2.310. Applying corrections within the solver allows subsequent velocity evaluations to respond to the corrected state.

5.5 Fixed-Delay Execution

We simulate delivery delays while the preceding plan continues executing (Table 4). SEAM skips correction of the committed prefix. It has the highest success at 100 and 200 ms, with lower EEF jerk than baseline at all three delays. At 300 ms, RTC has higher success and slightly lower jerk. Appendix G reports all command metrics and the prefix-treatment comparison.

5.6 Sensitivity to Guidance and Execution Length

Table 5 varies guidance strength, dimensions, and window length. All-dimension guidance at $\lambda = 0.1$ gives the highest success among the tested strengths; increasing λ to 0.15 barely changes BJ but lowers success by 2.9 percentage points. Position-only guidance has higher BJ and CD at every strength. Every tested window improves boundary metrics over baseline: $M = 20$ gives the lowest BJ and CD, while $M = 6$ has the highest success. Table 8 gives a supplementary configuration comparison on tasks T1, T7, and T9.

Table 6: Execution-length comparison on LIBERO-10, with 130 episodes per task, $H = 50$, and ten Euler steps. Success is in percent; EEF jerk is in $\mathrm{m/s^3}$. SEAM uses $\lambda = 0.1$ and $M = 20$. Best values within each row are bold.

	Baseline		RTC		SEAM	
K	Success↑	EEF jerk↓	Success↑	EEF jerk↓	Success↑	EEF jerk↓
5	**94.8**	3.003	**94.8**	**2.385**	94.0	2.423
10	**95.6**	2.847	94.8	2.386	94.8	**2.310**
20	**94.4**	2.758	92.8	2.332	93.6	**2.258**

Table 7: Motion on jointly successful states with $\pi_{0.5}$ at $K = 10$, averaged equally over tasks. Steps is mean completion length; EEF jerk is in $\mathrm{m/s^3}$. Lowest jerk in each row is bold.

	Baseline		RTC		SEAM	
Suite	Steps	EEF jerk↓	Steps	EEF jerk↓	Steps	EEF jerk↓
LIBERO-10	251.3	2.804	250.4	2.358	252.9	**2.299**
Spatial	102.9	3.873	103.1	3.408	103.0	**3.259**
Goal	107.6	4.020	107.4	3.575	111.0	**3.420**
Object	135.5	3.076	133.0	2.650	137.5	**2.637**

Table 6 compares execution lengths with fixed prediction horizon, solver steps, and guidance parameters. Across $K \in \{5, 10, 20\}$, SEAM reduces baseline EEF jerk by 18.1–19.3%. RTC has lower jerk at $K = 5$, while SEAM has lower jerk at $K = 10$ and 20. At $K = 20$, SEAM reaches 93.6% success, compared with 94.4% for baseline and 92.8% for RTC. Thus, execution cadence changes the relative motion quality even when the guidance setting is fixed. Appendix F reports the complete command metrics.

Excluding gripper guidance retains 94.8% success and most boundary improvements (Appendix C).

5.7 Motion on Jointly Successful Trajectories

Table 7 compares the same initial states where all three methods succeed. SEAM lowers EEF jerk on all four suites with similar completion lengths: all methods take approximately 103 steps on Spatial, while SEAM takes about four additional steps on Goal. Appendix H reports path lengths and other execution cadences.

6 Conclusion

We presented SEAM, a training-free inference method that couples an aligned previous-chunk reference with repeated corrections inside a flow solver. The mean-reference derivation gives a closed-form consistency update without policy-network backpropagation. Matched-state controls establish the contribution of reference correspondence and expose the effect of solver placement. Experiments with $\pi_{0.5}$ and SmolVLA demonstrate lower command-boundary variation and realized end-effector jerk, including improved task success on SmolVLA's LIBERO-Spatial comparison at approximately 1% full-policy overhead. These results show that cross-chunk reference alignment can improve both task completion and physical motion with a lightweight inference correction.

Statement on the Use of AI

Generative AI tools, including OpenAI Codex, were used to polish writing, retrieve and discover related literature, assist with research ideation and execution (experiment planning and monitoring), and draft sections of the manuscript.

Reproducibility Statement

Sections 4 and 5 specify the method, consistency objective, and experimental setup. Algorithm 1 summarizes SEAM inference, and Appendix A derives the mean-reference correction and its accumulation across solver steps. Appendix I documents the policy configurations, evaluation protocols, random generators, and metric definitions. Appendices F and G describe the execution-length and fixed-delay settings; Appendix E provides the SmolVLA checkpoint and evaluation configuration. Appendix J specifies the observation-to-action timing protocol.

References

Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, et al. PaliGemma: A versatile 3B VLM for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

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

Kevin Black, Manuel Galliker, and Sergey Levine. Real-time execution of action chunking flow policies. In *Advances in Neural Information Processing Systems*, volume 38, pp. 33383–33407, 2025a.

Kevin Black, Allen Z. Ren, Michael Equi, and Sergey Levine. Training-time action conditioning for efficient real-time chunking, 2025b. URL https://arxiv.org/abs/2512.05964.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. RT-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

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.

Hyungjin Chung, Jeongsol Kim, Michael T Mccann, Marc L Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. *arXiv preprint arXiv:2209.14687*, 2022.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

Kai Fang, Hailong Pei, and Xuemin Chi. Smoother action chunking flow policy via prior-corrected orthogonal trust-region guidance, 2026. URL https://arxiv.org/abs/2605.24433.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on Gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jindou Jia, Gen Li, Xiangyu Chen, Tuo An, Yuxuan Hu, Jingliang Li, Xinying Guo, and Jianfei Yang. Action-to-action flow matching. *arXiv preprint arXiv:2602.07322*, 2026.

Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. *Advances in Neural Information Processing Systems*, 35:23593–23606, 2022.

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

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

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 Ibn Hamid, Annie Xie, Yoonho Lee, Maximilian Du, and Chelsea Finn. Bidirectional decoding: Improving action chunking via guided test-time sampling. In *International Conference on Learning Representations*, 2025.

Yufeng Liu, Hang Yu, Juntu Zhao, Bocheng Li, Di Zhang, Mingzhu Li, Wenxuan Wu, Yingdong Hu, Junyuan Xie, Junliang Guo, Dequan Wang, and Yang Gao. Learning native continuation for action chunking flow policies. In *Robotics: Science and Systems*, 2026. URL https://arxiv.org/abs/2602.12978.

Yuxiang Lu, Zhe Liu, Xianzhe Fan, Zhenya Yang, Jinghua Hou, Junyi Li, Kaixin Ding, and Hengshuang Zhao. FASTER: Rethinking real-time flow VLAs. *arXiv preprint arXiv:2603.19199*, 2026.

Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. RePaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11461–11471, 2022.

Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

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

Kohei Sendai, Maxime Alvarez, Tatsuya Matsushima, Yutaka Matsuo, and Yusuke Iwasawa. Leave no observation behind: Real-time correction for VLA action chunks, 2025. URL https://arxiv.org/abs/2509.23224.

Nur Muhammad Shafiullah, Zichen Cui, Ariuntuya Arty Altanzaya, and Lerrel Pinto. Behavior transformers: Cloning k modes with one stone. *Advances in Neural Information Processing Systems*, 35:22955–22968, 2022.

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, Simon Alibert, Matthieu Cord, Thomas Wolf, and Remi Cadene. SmolVLA: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025. URL https://arxiv.org/abs/2506.01844.

Jiaming Tang, Yufei Sun, Yilong Zhao, Shang Yang, Yujun Lin, Zhuoyang Zhang, James Hou, Yao Lu, Zhijian Liu, and Song Han. VLASH: Real-time VLAs via future-state-aware asynchronous inference, 2026. URL https://arxiv.org/abs/2512.01031.

Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *arXiv preprint arXiv:2302.00482*, 2023.

Marcel Torne Villasevil, Andy Tang, Yuejiang Liu, and Chelsea Finn. Learning long-context diffusion policies via past-token prediction. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 1744–1755. PMLR, 2025. URL https://proceedings.mlr.press/v305/villasevil25a.html.

Haoxuan Wang, Gengyu Zhang, Yan Yan, Yuzhang Shang, Ramana Rao Kompella, and Gaowen Liu. Real-time robot execution with masked action chunking, 2026. URL https://arxiv.org/abs/2601.20130.

Rui Wang. Noise-space attribution and control of chunk-boundary artifact. *arXiv preprint arXiv:2603.11642*, 2026. URL https://arxiv.org/abs/2603.11642.

Junjie Wen, Minjie Zhu, Jiaming Liu, Zhiyuan Liu, Yicun Yang, Linfeng Zhang, Shanghang Zhang, Yichen Zhu, and Yi Xu. dVLA: Diffusion vision-language-action model with multimodal chain-of-thought. *arXiv preprint arXiv:2509.25681*, 2025a.

Yuqing Wen, Hebei Li, Kefan Gu, Yucheng Zhao, Tiancai Wang, and Xiaoyan Sun. LLaDA-VLA: Vision language diffusion action models. *arXiv preprint arXiv:2509.06932*, 2025b.

Wei Wu, Fan Lu, Yunnan Wang, Shuai Yang, Shi Liu, Fangjing Wang, Qian Zhu, He Sun, Yong Wang, Shuailei Ma, et al. A pragmatic VLA foundation model. *arXiv preprint arXiv:2601.18692*, 2026.

Daichi Yashima, Koki Seno, Shuhei Kurita, Yusuke Oda, and Komei Sugiura. HiFlow: Tokenization-free scale-wise autoregressive policy learning via flow matching. *arXiv preprint arXiv:2603.27281*, 2026.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11975–11986, 2023.

Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pp. 2165–2183. PMLR, 2023.

A Mean Reference and Quadratic Consistency

Fix the aligned tail $\mathbf{a} = \mathbf{a}^{\mathrm{al}}[1{:}M]$ and an Euler candidate $\mathbf{y} = \widetilde{\mathbf{x}}_{i+1}[1{:}M]$. A noisy interpolation reference has the form $\mathbf{R}_i = (1 - t_i)\mathbf{a} + t_i\boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is independent of the candidate's sampling noise. Taking the expectation over this reference noise gives

$$\mathbb{E}_{\boldsymbol{\epsilon}}\big[\|\mathbf{y} - \mathbf{R}_i\|_2^2\big] = \|\mathbf{y} - (1 - t_i)\mathbf{a}\|_2^2 + t_i^2 MD, \tag{10}$$

$$\nabla_{\mathbf{y}}\mathbb{E}_{\boldsymbol{\epsilon}}\big[\|\mathbf{y} - \mathbf{R}_i\|_2^2\big] = 2\big(\mathbf{y} - (1 - t_i)\mathbf{a}\big). \tag{11}$$

Here the norm sums over the MD guided coordinates. The cross term vanishes because the reference noise has zero mean, and its expected squared norm is MD. Thus, VLS evaluates the gradient of the noise-averaged quadratic consistency objective analytically. The aligned tail supplies the temporal content of the reference, while the current observation enters each Euler candidate through v_θ. After correction, the next velocity evaluation uses the updated solver state.

A.1 Accumulation across Solver Steps

Let $\mathbf{y}_i = \mathbf{x}_i[1{:}M]$, $s_i = 1 - t_i$, $\Delta t_i = t_{i+1} - t_i$, and $\mathbf{v}_i = v_\theta(\mathbf{x}_i, t_i, \mathbf{o})[1{:}M]$. Expanding the alternating Euler and VLS updates over one chunk query gives

$$\mathbf{y}_N = P\mathbf{y}_0 + \sum_{i=0}^{N-1} Q_i \Delta t_i \mathbf{v}_i + C\mathbf{a}, \tag{12}$$

where

$$P = \prod_{i=0}^{N-1}(1 - \alpha_i), \qquad Q_i = \prod_{j=i}^{N-1}(1 - \alpha_j), \tag{13}$$

$$C = \sum_{i=0}^{N-1} \alpha_i s_i \prod_{j=i+1}^{N-1}(1 - \alpha_j). \tag{14}$$

The reference and correction weight use the pre-step time t_i; λ controls the per-step correction without step-size normalization, so its cumulative effect depends on the number of denoising steps N. The velocity sequence is evaluated at the actual corrected states, so each local correction also affects subsequent velocity predictions. With $N = 10$ and $\lambda = 0.1$, the implemented schedule gives $C = 0.4300$. We use this explicit reference coefficient as the output-blending weight to compare final-output correction with guidance inside the solver.

B Additional Ablations and Qualitative Rollouts

B.1 Comparison of Guidance Configurations

We compare SEAM guidance configurations on three LIBERO-10 tasks, T1, T7, and T9, with 130 episodes per task, matching the per-task episode budget of the main comparison in Table 1. This gives 390 episodes per configuration. All configurations use policy seed 2027, the same checkpoint, $H = 50$, $K = 10$, and ten Euler steps. Table 8 compares guidance strengths, window lengths, and guided action dimensions, including the original $\lambda = 0.1$, $M = 20$ setting and the $\lambda = 0.15$, $M = 20$ setting.

Table 8: Comparison of SEAM guidance configurations on LIBERO-10 tasks T1, T7, and T9, with 130 episodes per task and 390 episodes per row. SEAM guides all dimensions except in the translation + rotation row, which guides those six controller channels. All guided rows use fresh Gaussian initialization.

Setting	λ	M	Succ.%↑	BJ↓	CD↓
Baseline	–	–	85.3	0.244	0.216
SEAM (original)	0.10	20	89.3	0.153	0.137
SEAM	0.05	20	88.0	0.194	0.174
SEAM	0.15	20	90.7	0.143	0.134
SEAM	0.10	10	89.3	0.150	0.135
SEAM (translation + rotation)	0.10	20	86.7	0.196	0.181

B.2 Guidance Strength and Window Length

Tables 9 and 10 report the strength, dimension, and window-length sweeps summarized in Table 5.

Table 9: Ablation of SEAM guidance strength and dimension selection on LIBERO-10, with 130 episodes per task. "pos": guidance on position dimensions only. "all": all action dimensions. All guided rows use standard Gaussian initialization with M=20. Best guided values are in boldface.

Dims	λ	M	**Succ.%↑**	**BJ↓**	**IJ↓**	**CD↓**
none	–	–	94.8	0.194	0.094	0.172
all	0.05	20	94.8	0.162	0.082	0.145
all	0.10	20	**95.7**	0.141	0.074	**0.126**
all	0.15	20	92.8	**0.140**	0.071	0.127
all	0.20	20	89.5	0.147	**0.067**	0.135
pos	0.05	20	94.3	0.179	0.089	0.159
pos	0.10	20	94.0	0.162	0.085	0.143
pos	0.15	20	92.6	0.166	0.085	0.145
pos	0.20	20	88.2	0.168	0.083	0.152

Table 10: Sweep of SEAM guided-window length M on LIBERO-10, with 130 episodes per task. All SEAM rows use all action dimensions with λ=0.1 and standard Gaussian initialization. Best values are in boldface.

Setting	M	**Succ.%↑**	**BJ↓**	**CD↓**
Baseline	–	94.8	0.194	0.172
SEAM	2	95.5	0.148	0.142
SEAM	4	94.8	0.156	0.146
SEAM	6	**96.3**	0.147	0.137
SEAM	8	95.0	0.149	0.138
SEAM	10	95.9	0.148	0.135
SEAM	12	95.1	0.153	0.140
SEAM	14	95.7	0.154	0.141
SEAM	16	94.7	0.147	0.134
SEAM	18	95.5	0.148	0.135
SEAM	20	95.7	**0.141**	**0.126**

VLS strength trade-off. We sweep the guidance strength λ at fixed M=20 for both all-dimension and position-only guidance (Table 9). In the all-dimension rows, λ=0.05 reaches 94.8% success, while the main λ=0.1 setting reaches 95.7% with BJ 0.141 and CD 0.126. Larger strengths improve some continuity metrics but reduce success to 92.8% at λ=0.15 and 89.5% at λ=0.2. These results favor an intermediate guidance strength in the original comparison, balancing continuity with the policy's ability to revise the preceding plan.

Dimension selection. Table 9 compares all-dimension and position-only guidance at the same strengths and window length. At λ=0.1 and M=20, position-only guidance yields BJ 0.162 and CD 0.143 with 94.0% success, whereas all-dimension guidance yields BJ 0.141 and CD 0.126 with 95.7% success. Guiding the full action prediction therefore provides a stronger continuity–success operating point in this sweep.

Window length. Every tested M reduces BJ and CD relative to baseline (Table 10), with success ranging from 94.7% to 96.3%. The lowest BJ and CD occur at $M = 20$, while the highest recorded success occurs at $M = 6$. The window controls how much of the previous plan enters the consistency objective, providing another way to adjust continuity and task completion.

Qualitative T1 failure patterns. Figure 3 compares illustrative T1 rollouts. The baseline example drops the object during transfer; ACT-TE shows delayed grasp contact; RTC continues an unsuc-

cessful early grasp. The SEAM example completes the task with consistent chunk transitions. These sequences show how action continuity interacts with contact timing during execution.



Figure 3: Illustrative T1 rollouts. The examples show a transfer failure for baseline, delayed contact for ACT-TE, an unsuccessful grasp continuation for RTC, and successful execution for SEAM. Table 11 reports the corresponding task-level evaluation.

Task-level success and inference cost. Table 11 gives the task-level success rates underlying Table 1, and Figure 4 compares success, continuity, and per-query denoising cost.

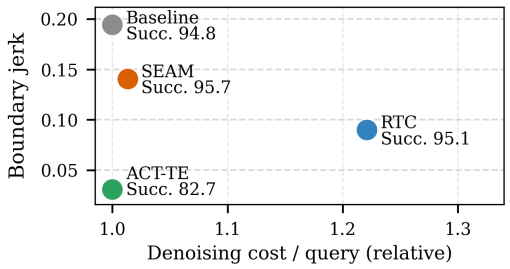


Figure 4: Per-query cost and action continuity. The horizontal axis shows relative cached-prefix denoising-loop cost per query; point labels give success rates. ACT-TE makes ten queries per ten executed actions, giving a cadence-adjusted denoising cost of 10.00× baseline, despite its 1.00× per-query cost shown here.

Table 11: Per-task success rates on LIBERO-10, with 130 episodes per task and method. T1–T10 denote the fixed LIBERO-10 task order used for the aggregate results in Table 1. Best values in each row are in boldface; ties are bolded for all tied methods.

Task	Base.%	SEAM%	RTC%	ACT-TE%
T1	91.5	**99.2**	90.8	58.5
T2	98.5	**100.0**	93.8	93.8
T3	96.9	**100.0**	96.9	93.1
T4	**97.7**	96.9	96.2	91.5
T5	92.3	**95.4**	94.6	87.7
T6	99.2	**100.0**	**100.0**	97.7
T7	93.8	90.8	**99.2**	80.0
T8	**97.7**	96.9	**97.7**	81.5
T9	83.8	83.8	**86.2**	55.4
T10	**96.2**	93.8	95.4	87.7
Overall	94.8	**95.7**	95.1	82.7

C Component Diagnostics

Table 12 separates boundary variation into translation, rotation, and gripper components and reports temporal command variance for the matched-state comparison.

Table 12: Command diagnostics for the matched-state LIBERO-10 comparison, with 130 episodes per task and 1,300 episodes per method. Translation, rotation, and gripper BJ use the same boundary indices as the joint metric. Component norms do not sum to the joint norm. Temporal variance averages per-channel population variance within each episode, then over episodes and tasks.

Method	Translation BJ	Rotation BJ	Gripper BJ	Temporal var.
Baseline	0.114	0.025	0.092	0.181
RTC	0.040	0.008	0.052	0.180
Shared initial noise	0.123	0.026	0.100	0.180
Zero reference	0.138	0.022	0.110	0.109
Shuffled reference	0.116	0.021	0.098	0.153
Noisy reference	0.096	0.020	0.124	0.170
Output blend (0.4300)	0.083	0.017	0.089	0.172
SEAM ($\lambda = 0.1$)	0.082	0.017	0.085	0.169
SEAM (no gripper guidance)	0.080	0.017	0.090	0.178
SEAM ($\lambda = 0.15$)	0.073	0.015	0.079	0.163

Gripper guidance. Table 13 compares baseline, full SEAM, and SEAM with the gripper coordinate excluded from guidance. Excluding gripper guidance matches full SEAM's success and retains most of its boundary improvements. Full guidance gives lower BJ and clean IJ, while realized end-effector jerk is similar. The translation and rotation continuity gains therefore persist when the gripper is not directly guided.

Table 13: Effect of gripper guidance on the matched-state LIBERO-10 comparison, with 130 episodes per task, $K = 10$, and 1,300 episodes per method. Both SEAM settings use $\lambda = 0.1$ and $M = 20$. EEF jerk is in $\mathrm{m/s^3}$.

Method	Success (%) ↑	BJ ↓	CD ↓	Clean IJ ↓	EEF jerk ↓
Baseline	95.6	0.193	0.174	0.080	2.847
SEAM (all dimensions)	94.8	0.153	0.139	0.066	2.310
SEAM (no gripper guidance)	94.8	0.161	0.140	0.072	2.297

D Additional Task Suites

Table 14 reports the additional suite results. All methods retain the same $\pi_{0.5}$ checkpoint and $H = 50$, $K = 10$, and ten Euler steps. On Spatial, SEAM reduces baseline BJ, CD, and realized end-effector jerk by 25.4%, 26.0%, and 17.7%, respectively. On Goal, the corresponding reductions are 10.3%, 9.3%, and 14.0%. SEAM achieves 97.2% success on Goal, compared with 99.2% for baseline and 96.8% for RTC, with lower mean end-effector jerk than both comparators. On Object, SEAM and RTC each achieve 99.6% success, versus 99.2% for baseline. SEAM reduces baseline BJ, CD, and realized jerk by 18.6%, 14.5%, and 14.8%, with jerk nearly identical to RTC.

Table 14: Additional $\pi_{0.5}$ evaluation on LIBERO suites, with 130 episodes per task and 1,300 episodes per method and suite. SEAM uses the original $\lambda = 0.1$, $M = 20$ configuration. Clean IJ excludes both second-difference centers whose action triples cross a chunk boundary. EEF jerk is the mean norm of the third finite difference of end-effector positions at 20 Hz, in $\mathrm{m/s^3}$.

Suite	Method	Success (%) ↑	BJ ↓	CD ↓	Clean IJ ↓	EEF jerk ↓
Spatial	Baseline	98.0	0.199	0.176	0.094	4.027
	RTC	96.4	0.100	0.111	0.091	3.471
	SEAM	96.8	0.149	0.130	0.077	3.315
Goal	Baseline	99.2	0.175	0.154	0.079	4.060
	RTC	96.8	0.103	0.108	0.080	3.644
	SEAM	97.2	0.157	0.140	0.070	3.493
Object	Baseline	99.2	0.189	0.157	0.095	3.134
	RTC	99.6	0.096	0.094	0.085	2.670
	SEAM	99.6	0.154	0.135	0.078	2.671

E Second Flow-Matching Backbone

We use the public `HuggingFaceVLA/smolvla_libero` checkpoint of SmolVLA (Shukor et al., 2025), revision `6721902bc4d61e50a3bfdb11dfb4cb626f05d102`. We retain its pretrained modules, observation preprocessing, action normalization, and float32 inference without automatic mixed precision or model compilation. The comparison uses $H = 50$, $K = 10$, and ten Euler steps for baseline, RTC, and SEAM. SEAM applies the same post-Euler recurrence as Algorithm 1, with $\lambda = 0.1$, $M = 20$, and fresh Gaussian initialization. RTC uses weight five, an exponential prefix-attention schedule, execution horizon ten, and zero delivery delay. Both methods reuse the checkpoint's velocity field and cached observation prefix.

We evaluate all ten tasks in each of LIBERO-Spatial, LIBERO-10, LIBERO-Goal, and LIBERO-Object. Each method uses 130 episodes per task and batch size five, with initial states and environment reset seeds matched across methods. The policy random generator is initialized with seed 2027 once per method and suite, then continues across tasks. The first-terminal masks, action diagnostics, and realized end-effector metrics follow Appendix I. Guidance settings remain fixed across suites and backbones. Table 15 reports success and motion metrics for all four suites.

Table 15: SmolVLA comparison with 130 episodes per task and 1,300 episodes per method and suite. All methods use the same public checkpoint and execution settings. EEF jerk is in $\mathrm{m/s^3}$.

Suite	Method	Success (%) ↑	BJ ↓	CD ↓	Clean IJ ↓	EEF jerk ↓
Spatial	Baseline	65.6	0.708	0.539	0.339	9.070
	RTC	60.0	0.365	0.246	0.306	6.228
	SEAM	72.0	0.352	0.275	0.176	3.738
LIBERO-10	Baseline	45.6	0.646	0.492	0.291	6.497
	RTC	36.4	0.311	0.199	0.253	4.413
	SEAM	34.0	0.356	0.276	0.158	2.517
Goal	Baseline	76.8	0.566	0.443	0.230	7.469
	RTC	70.8	0.256	0.178	0.200	5.396
	SEAM	78.4	0.252	0.197	0.107	3.454
Object	Baseline	88.8	0.617	0.483	0.270	6.643
	RTC	83.6	0.309	0.200	0.256	5.035
	SEAM	88.8	0.325	0.252	0.153	3.174

On Spatial, SEAM achieves 72.0% success, compared with 65.6% for baseline and 60.0% for RTC (Table 15). Its BJ, CD, and realized end-effector jerk are 50.2%, 48.9%, and 58.8% below baseline.

Relative to RTC, SEAM has 3.6% lower BJ and 40.0% lower realized jerk, while RTC has lower CD.

On LIBERO-10, SEAM achieves 34.0% success versus 45.6% for baseline and 36.4% for RTC. Its BJ, CD, and realized jerk are 44.9%, 44.0%, and 61.3% below baseline. RTC has lower boundary metrics and higher success than SEAM, while SEAM has 43.0% lower realized jerk. The reduced motion variation accompanies lower task completion in this long-horizon suite.

On Goal, SEAM achieves 78.4% success versus 76.8% for baseline and 70.8% for RTC. It reduces baseline BJ, CD, and realized jerk by 55.4%, 55.6%, and 53.8%, respectively. SEAM also has 1.5% lower BJ and 36.0% lower realized jerk than RTC; RTC again produces lower CD.

On Object, SEAM and baseline each achieve 88.8% success, compared with 83.6% for RTC. SEAM reduces baseline BJ, CD, and realized jerk by 47.3%, 47.9%, and 52.2%, respectively. RTC has lower boundary metrics, while SEAM produces 36.9% lower realized jerk.

Motion on jointly successful states. The three methods jointly succeed on 39.2% of Spatial episodes, 51.6% of Goal episodes, and 70.8% of Object episodes, with at least one shared successful state in every task. Table 16 applies the same equal-task averaging as Appendix H. SEAM retains lower jerk on these common sets: it takes approximately 107 completion steps on Spatial versus 102 for baseline and 104 for RTC, and 117 steps on Goal versus approximately 113 for both comparators. On Object, SEAM completes the common states in approximately 138 steps, versus 141 for baseline and 142 for RTC. LIBERO-10 has empty three-method success intersections on T1, T7, and T8; the suite-level common-success analysis requires all ten tasks to be represented and therefore covers the other three suites.

Table 16: SmolVLA motion on jointly successful initial states, with tasks weighted equally. Joint success is the percentage of evaluation episodes in which all three methods succeed. EEF jerk is in $\mathrm{m/s^3}$, path length in meters, and completion length in control steps.

Suite	Joint success (%)	Method	Steps	EEF jerk ↓	EEF path
Spatial	39.2	Baseline	102.1	7.688	0.938
		RTC	103.9	5.786	0.936
		SEAM	107.3	3.625	0.908
Goal	51.6	Baseline	112.6	7.299	0.803
		RTC	112.6	5.486	0.813
		SEAM	116.9	3.470	0.774
Object	70.8	Baseline	141.4	6.242	1.127
		RTC	141.8	4.592	1.125
		SEAM	137.8	2.913	1.050

Full-policy timing. We apply the observation-to-action procedure in Appendix J to SmolVLA on an idle RTX 3090, retaining float32 inference and the formal evaluation configurations. We use the same real LIBERO-10 task-0 observation from initial state 25, batch size one, 50 warmup action calls, and 200 measured calls with CUDA synchronization. Each method makes 20 model queries. Table 17 includes preprocessing, observation encoding, denoising, queue handling, postprocessing, and CPU action transfer. SEAM adds 4.6 ms per query (0.8%) and 0.9% to the ten-action total relative to baseline.

Table 17: Full SmolVLA policy timing in milliseconds, with the same protocol and cost definitions as Table 21. Relative cost uses the SmolVLA baseline's ten-action total.

Method	Queries / ten actions	Query (ms)	Ten actions (ms)	Relative cost
Baseline	1	546.9	599.8	1.000×
RTC	1	695.6	748.9	1.249×
SEAM	1	551.5	605.4	1.009×

F EXECUTION LENGTH

Table 18 compares execution lengths while retaining $H = 50$, ten Euler steps, and the original SEAM configuration ($\lambda = 0.1$, $M = 20$). RTC's execution horizon equals K in each setting. Each row uses all ten LIBERO-10 tasks with 130 episodes per task, matched initial states across methods, and zero delivery delay. The $K = 10$ rows reproduce the matched-state comparison in Table 3; the $K = 5$ and 20 runs each use one full-suite process with policy seed 2027. Appendix I specifies the random-generator and trajectory protocol.

Table 18: Execution-length comparison on LIBERO-10, with 130 episodes per task and 1,300 episodes per row. All settings retain the same checkpoint, prediction horizon, solver steps, and SEAM guidance parameters. EEF jerk is measured in $\mathrm{m/s^3}$.

K	Method	Success (%) ↑	BJ ↓	CD ↓	Clean IJ ↓	EEF jerk ↓
5	Baseline	94.8	0.173	0.144	0.088	3.003
5	RTC	94.8	0.099	0.088	0.076	2.385
5	SEAM	94.0	0.139	0.117	0.077	2.423
10	Baseline	95.6	0.193	0.174	0.080	2.847
10	RTC	94.8	0.089	0.088	0.079	2.386
10	SEAM	94.8	0.153	0.139	0.066	2.310
20	Baseline	94.4	0.237	0.214	0.077	2.758
20	RTC	92.8	0.093	0.092	0.076	2.332
20	SEAM	93.6	0.226	0.211	0.066	2.258

At $K = 5$, SEAM reduces BJ, CD, and realized end-effector jerk relative to baseline by 19.3%, 18.6%, and 19.3%, respectively, achieving 94.0% success versus 94.8% for both comparators. RTC has the lowest jerk at this cadence. At $K = 20$, SEAM retains lower realized end-effector jerk and clean interior variation than both comparators. It achieves 93.6% success, compared with 94.4% for baseline and 92.8% for RTC. Its BJ and CD reductions relative to baseline are 5.0% and 1.6%, smaller than at $K = 10$, while its realized end-effector jerk reduction remains 18.1%. RTC produces the lowest boundary metrics at all three execution lengths. Thus, execution cadence changes the boundary-continuity benefit even when the guidance parameters remain fixed.

G FIXED-DELAY EXECUTION

We emulate delivery delays of $d = 2$, 4, and 6 control steps (100, 200, and 300 ms at 20 Hz). A query uses the observation at request time while the preceding plan supplies the next d executed actions. On delivery, execution begins at positions $d + 1$ through $d + K$ of the new chunk. The first chunk supplies $K + d$ actions; subsequent delivered chunks supply K. We retain the full checkpoint, $H = 50$, $K = 10$, ten Euler steps, and the same matched initial-state protocol and 130 episodes per task as the synchronous mechanism comparison.

Table 19 compares baseline and RTC with SEAM at the original $\lambda = 0.1$, $M = 20$ setting. The skip-prefix variant applies VLS to positions $d + 1$ through M, leaving the first d candidate positions unchanged. At $d = 4$, we also evaluate a replace-prefix variant that sets those first d positions to $\mathbf{r}_i$ after each Euler step and applies the usual VLS correction to positions $d + 1$ through M. Both references align to the preceding tail at request time. We evaluate action boundaries at execution switches $K + d, 2K + d, \ldots$, using the same first-terminal masks and Cartesian trajectory diagnostic as the synchronous runs.

Table 19: Fixed-delay comparison on LIBERO-10, with 130 episodes per task and 1,300 episodes per row. Each delay step d is 50 ms. Boundary metrics use the actual delayed execution switches. EEF jerk is in $\mathrm{m/s^3}$.

d	Method	Success (%) ↑	BJ ↓	CD ↓	Clean IJ ↓	EEF jerk ↓
2	Baseline	95.2	0.204	0.177	0.078	2.852
2	RTC	93.6	0.084	0.084	0.082	2.376
2	SEAM (skip prefix)	97.2	0.167	0.145	0.067	2.348
4	Baseline	93.6	0.235	0.213	0.079	2.882
4	RTC	92.4	0.107	0.090	0.086	2.452
4	SEAM (replace prefix)	90.8	0.166	0.159	0.067	2.293
4	SEAM (skip prefix)	94.0	0.190	0.172	0.068	2.407
6	Baseline	88.8	0.270	0.249	0.075	2.999
6	RTC	91.2	0.124	0.100	0.080	2.522
6	SEAM (skip prefix)	89.6	0.225	0.209	0.069	2.527

At $d = 2$, skip-prefix SEAM achieves 97.2% success, compared with 95.2% for baseline and 93.6% for RTC. It reduces baseline BJ, CD, and realized end-effector jerk by 18.2%, 18.4%, and 17.7%, respectively, with 1.2% lower jerk than RTC. At $d = 4$, skipping the committed prefix achieves 94.0% success, compared with 93.6% for baseline and 92.4% for RTC. Replacing the prefix yields lower BJ and CD than skipping it, with 90.8% success. Prefix treatment changes the balance between continuity and task completion even at the same delivery delay and guidance strength.

At $d = 6$, skip-prefix SEAM achieves 89.6% success versus 88.8% for baseline and 91.2% for RTC. SEAM reduces baseline BJ, CD, and realized end-effector jerk by 16.5%, 16.0%, and 15.7%, respectively. RTC and SEAM have similar mean end-effector jerk (2.522 and 2.527), with RTC attaining higher success at this longer delay. RTC produces the lowest boundary metrics at all three delays.

H Motion on Jointly Successful Initial States

We identify initial states for which baseline, RTC, and SEAM all complete the task. These jointly successful episodes account for 88.4%, 88.8%, and 85.2% of LIBERO-10 evaluations at $K = 5$, 10, and 20, respectively, and 94.4% on LIBERO-Spatial, 93.6% on LIBERO-Goal, and 98.4% on LIBERO-Object at $K = 10$. For each method, we average episode-level jerk, completion steps, and end-effector path length over this common set within each task, then average the ten tasks equally. Table 20 shows that SEAM's smoother successful motion accompanies similar completion lengths. On LIBERO-Spatial, all three methods average approximately 103 control steps, while SEAM reduces mean jerk from 3.873 to 3.259 relative to baseline and from 3.408 to 3.259 relative to RTC. On Goal, SEAM has lower jerk at 111 steps versus approximately 107 for both comparators. On Object, SEAM reaches jerk 2.637 at 137 steps, compared with baseline's 3.076 at 136 steps and RTC's 2.650 at 133 steps. At $K = 20$ on LIBERO-10, the three methods average approximately 251 steps, with SEAM again producing the lowest realized jerk. At $K = 5$, RTC has slightly lower jerk than SEAM, which takes approximately seven additional steps on the shared successful states.

Table 20: Motion on jointly successful initial states. Joint success is the percentage of evaluation episodes in which all three methods succeed for each suite and execution length. EEF jerk is in $\mathrm{m/s^3}$, path length in meters, and completion length in control steps. The three methods use the same states within each group, with tasks weighted equally.

Suite	K	Joint success (%)	Method	Steps	EEF jerk ↓	EEF path
LIBERO-10	5	88.4	Baseline	257.4	2.954	1.583
			RTC	258.1	2.395	1.581
			SEAM	264.8	2.414	1.559
LIBERO-10	10	88.8	Baseline	251.3	2.804	1.588
			RTC	250.4	2.358	1.586
			SEAM	252.9	2.299	1.554
LIBERO-10	20	85.2	Baseline	251.4	2.690	1.599
			RTC	251.3	2.262	1.591
			SEAM	251.1	2.195	1.557
LIBERO-Spatial	10	94.4	Baseline	102.9	3.873	0.948
			RTC	103.1	3.408	0.946
			SEAM	103.0	3.259	0.927
LIBERO-Goal	10	93.6	Baseline	107.6	4.020	0.778
			RTC	107.4	3.575	0.779
			SEAM	111.0	3.420	0.778
LIBERO-Object	10	98.4	Baseline	135.5	3.076	1.113
			RTC	133.0	2.650	1.104
			SEAM	137.5	2.637	1.099

I Evaluation Details

I.1 Benchmarks and Policy Configurations

Benchmark. We evaluate on LIBERO-10 (Liu et al., 2023), a suite of 10 long-horizon tabletop manipulation tasks in the LIBERO simulation environment. Each task requires multi-step reasoning with diverse objects, involving skills such as picking, placing, opening, and closing.

Backbone Policies. Our primary backbone is $\pi_{0.5}$ (Physical Intelligence et al., 2025), using the public LIBERO-fine-tuned checkpoint `lerobot/pi05_libero_finetuned`. It employs a PaliGemma (Beyer et al., 2024) vision-language model with a SigLIP (Zhai et al., 2023) vision encoder and a Gemma-300M (Gemma Team et al., 2024) action expert. It generates $H{=}50$ actions with $N{=}10$ Euler steps and executes $K{=}10$ per query (overlap $L{=}40$). We also evaluate SmolVLA (Shukor et al., 2025) with the same H, K, and N (Appendix E).

SEAM Setting. Unless otherwise noted, SEAM uses VLS guidance with $\lambda{=}0.1$, an increasing $(1{-}t)$ guidance schedule, guided window $M{=}20$, all action dimensions, and standard Gaussian initialization. With $N = 10$, the blend coefficients are $\alpha_i = 0, 0.02, \ldots, 0.18$. The aligned prior repeats the final unexecuted-tail action to fill the remaining $H{-}L$ positions.

Baseline settings. ACT temporal ensembling uses coefficient $k{=}0.01$ and queries the policy at every environment step. RTC uses ΠGDM guidance with maximum weight $\beta{=}5$, execution horizon ten, and exponential prefix-attention decay.

The complete ten-task suite uses 1,300 episodes per method; the three-task configuration comparison uses 390. SmolVLA checkpoint and inference details appear in Appendix E.

I.2 Evaluation Protocols

The main comparison uses sequential environment initial-state selection and the original rollout mask, which retains one step after the first terminal flag when available. The controlled comparisons below use explicitly matched initial states and truncate trajectories at the first terminal step. We report these evaluation cohorts separately.

The matched-state comparisons use all ten tasks in each named LIBERO suite, 130 episodes per task, batch size five, and policy seed 2027. Initial states and environment reset seeds are matched across methods. Unless otherwise stated, we use $H = 50$, $K = 10$, ten Euler steps, and bfloat16 inference without model compilation. Trajectories end at their first terminal step. The original SEAM setting uses $\lambda = 0.1$ and $M = 20$; we also evaluate $\lambda = 0.15$ with the same window length.

The $\pi_{0.5}$ $K = 10$ synchronous RTC comparisons on LIBERO-10 and LIBERO-Spatial use fixed task groups to parallelize evaluation: $(0, 1, 2)$, $(3, 4, 5)$, $(6, 7)$, and $(8, 9)$ on LIBERO-10, and $(0, 1, 2, 3)$, $(4, 5, 6)$, and $(7, 8, 9)$ on LIBERO-Spatial. Each process initializes the policy generator with seed 2027 and then evaluates its tasks in order. Environment initial states match the other methods; policy noise follows this group-specific sequence. Results average tasks equally, irrespective of group size. Other matched-state comparisons process all ten tasks sequentially, initializing the policy generator with seed 2027 at the start of each method.

The zero-reference control replaces the aligned tail with zero. The shuffled-reference control takes the tail from another parallel rollout of the same task through a cyclic batch shift. Both retain the original correction schedule and guided window. The noisy-reference control uses $\mathbf{r}_i = (1 - t_i)\mathbf{a}^{\mathrm{al}}[1{:}M] + t_i\boldsymbol{\epsilon}_{\mathrm{ref}}$, drawing one independent Gaussian reference tensor per chunk and holding it fixed across solver steps. A separate random generator with seed 2028 supplies these tensors. All reference controls retain fresh Gaussian initialization for generation.

The shared-initial-noise baseline samples one Gaussian tensor per rollout and reuses it without shifting at every query; observations continue updating and the full solver runs without guidance. Output blending instead applies one correction to the final prediction over the same $M = 20$ window. Its fixed weight is $C = 0.4300$, the explicit tail coefficient in the unrolled VLS recurrence (Appendix A.1).

I.3 Metric Definitions

Action metrics use the seven unnormalized controller channels: three relative translations, three relative rotations, and one gripper command.

For an action sequence $\{\mathbf{a}_t\}_{t=0}^{T-1}$, define

$$j_t = \|\mathbf{a}_{t+1} - 2\mathbf{a}_t + \mathbf{a}_{t-1}\|_2 \tag{15}$$

at each valid center t. Let $\mathcal{V}_j$ be the set of $t \in \{1, \ldots, T-2\}$ whose action triples $(\mathbf{a}_{t-1}, \mathbf{a}_t, \mathbf{a}_{t+1})$ are valid, $B_j = \{t \in \mathcal{V}_j \mid t \bmod K = 0\}$, and $I = \mathcal{V}_j \setminus B_j$. For discontinuity, let B_d be the valid boundary indices $t \in \{1, \ldots, T-1\}$ for which $(\mathbf{a}_{t-1}, \mathbf{a}_t)$ is valid and $t \bmod K = 0$. The four motion metrics are

$$\mathrm{BJ} = \frac{1}{|B_j|} \sum_{t \in B_j} j_t, \tag{16}$$

$$\mathrm{IJ} = \frac{1}{|I|} \sum_{t \in I} j_t, \tag{17}$$

$$\mathrm{CD} = \frac{1}{|B_d|} \sum_{t \in B_d} \|\mathbf{a}_t - \mathbf{a}_{t-1}\|_2, \tag{18}$$

$$\mathrm{BJV} = \frac{1}{|B_j|} \sum_{t \in B_j} (j_t - \mathrm{BJ})^2. \tag{19}$$

Metrics are accumulated over valid rollout steps within each task and then averaged across the ten tasks.

Given simulator end-effector positions $\mathbf{p}_t$ at 20 Hz, physical jerk is $\|\mathbf{p}_{t+3} - 3\mathbf{p}_{t+2} + 3\mathbf{p}_{t+1} - \mathbf{p}_t\|_2/(0.05\,\mathrm{s})^3$. Clean IJ removes second-difference centers at both $mK-1$ and mK in synchronous execution, with the corresponding shift for delayed switches. Temporal command variance averages per-channel population variances within each episode, then over episodes and tasks. Action and physical-jerk metrics accumulate valid samples within each task before averaging tasks equally.

J Full Observation-to-Action Timing

Timing Protocol. We report denoising-loop timing on an RTX 3090 in Table 1, averaging 20 queries after five warmup queries at batch size one. Chunk measures the cached-prefix ODE loop for one query with $N = 10$ Euler steps, and Step is Chunk divided by ten. This scope excludes observation encoding, environment stepping, and queue handling. ACT-TE uses the baseline denoising path but queries every environment step, whereas the other methods query every ten steps. Thus, their denoising cost per ten executed actions is approximately 2822 ms for ACT-TE, 282.2 ms for baseline, 344.6 ms for RTC, and 286.0 ms for SEAM. These cadence-adjusted values describe denoising compute, with relative costs $10.00\times$, $1.00\times$, $1.22\times$, and $1.01\times$, respectively.

Table 21 reports the complete policy path on an RTX 3090 at batch size one: CPU observation preprocessing, tokenization, image encoding, denoising, action queuing or temporal ensembling, action postprocessing, and transfer to a CPU action array. We replay a real LIBERO-10 task-0 observation from initial state 25, using the same checkpoint, $H = 50$, $K = 10$, ten Euler steps, and bfloat16 inference. After 50 warmup action calls, we measure 200 calls with CUDA synchronization around each wall-clock measurement. This produces 20 model queries for baseline, RTC, and SEAM, and 200 for ACT-TE. Environment stepping is outside the measured path.

Table 21: Full policy pipeline timing in milliseconds. Query averages calls that generate a new chunk. Ten actions accumulates all policy calls needed for ten control steps, including calls that retrieve queued actions. Relative cost uses this ten-action total.

Method	Queries / ten actions	Query (ms)	Ten actions (ms)	Relative cost
Baseline	1	401.7	442.2	$1.000\times$
ACT-TE	10	400.7	4006.8	$9.061\times$
RTC	1	475.1	513.9	$1.162\times$
SEAM	1	407.5	443.8	$1.004\times$

SEAM's mean full-query overhead is 5.8 ms, or 1.4% of baseline query time. The ten-action totals retain preprocessing and queue-handling costs on every call. ACT-TE has a similar per-query cost to baseline and a much higher execution-cadence cost because it generates a fresh chunk at every control step. These measurements complement the cached-prefix denoising times in Table 1 by including the rest of the policy pipeline.