BATON: PASSING DENOISING TRAJECTORIES ACROSS REPLANS FOR PARALLEL SAMPLING OF ROBOT POLICIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion and flow policies built on large action transformers generate each action chunk through many narrow, dependent model calls. We present BATON, a training-free sampler that evaluates the sampling steps of a frozen receding-horizon policy in parallel. BATON is based on two designs. First, it transports the previous denoising trajectory into the next replan by shifting the action horizon, completing the tail, and restoring the current noise boundary. Second, it starts two parallel Picard rounds over the full ten-step schedule from this trajectory and combines them with a depth-one Anderson extrapolation that adds no model calls. Ten dependent transformer calls thus become two wider ones. Without assuming contraction, we bound the error of the raw parallel rounds linearly in the initializer error, which is bounded by the change of the policy's plan between replans and the drift of its clean predictions, and explain the Anderson step as averaging an oscillation of the raw rounds. On 450 paired LIBERO-Plus instances at a fixed execution stride, BATON reduces Fast-WAM's median action-expert latency by $2.80\times$ and backend latency by $1.73\times$, and solves 311 instances versus 304 for serial sampling, a difference that is not statistically significant. Paired ablations show that removing either transport or Anderson extrapolation lowers success. The speedups carry over to Fast-WAM on RoboTwin 2.0 ($2.74\times$ action expert, $1.68\times$ backend), and with $\pi_{0.5}$ on LIBERO-Plus the compiled full policy call becomes $1.30\times$ faster, again without a significant change in success.

1 INTRODUCTION

Diffusion and flow policies generate robot actions by repeatedly evaluating a learned denoising model (Chi et al., 2023; Black et al., 2025a; Yuan et al., 2026). For a large action transformer, each evaluation is narrow: one call processes only a few dozen action tokens, and the sampler repeats this call at every sampling step. In one optimized Fast-WAM profile, the ten-step action expert accounts for 72.4% of backend CUDA time.

Driven by this cost, a question arises: can the sampling steps be evaluated together rather than one after another? Batching several denoising states into one call would amortize parameter-serving and launch costs. An obstacle is that each step takes the output of the previous one as its input.

This dependency, however, can be relaxed. Parallel samplers such as ParaDiGMS and ParaTAA solve for the whole denoising trajectory as a fixed point, so that all steps are updated at once (Shih et al., 2023; Tang et al., 2024). When the policy already uses only ten steps, a different difficulty is exposed: the fixed-point iteration must converge within very few rounds to be worth running. Started from noise, two parallel rounds solve 6 of 450 LIBERO-Plus instances, whereas the serial sampler solves 306 (Table 2).

Receding-horizon control suggests where a better starting point lies. After the robot executes the first S actions of a chunk, the next horizon overlaps the remaining $H - S$ actions of the previous one. The previous solver trajectory therefore already contains much of the next one, shifted along action time. Prior methods reuse historical actions to shorten sequential denoising (Duan et al., 2025; Chen et al., 2025a); we reuse the entire denoising trajectory to initialize a parallel solve. Appendix A discusses related work in more detail.

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.

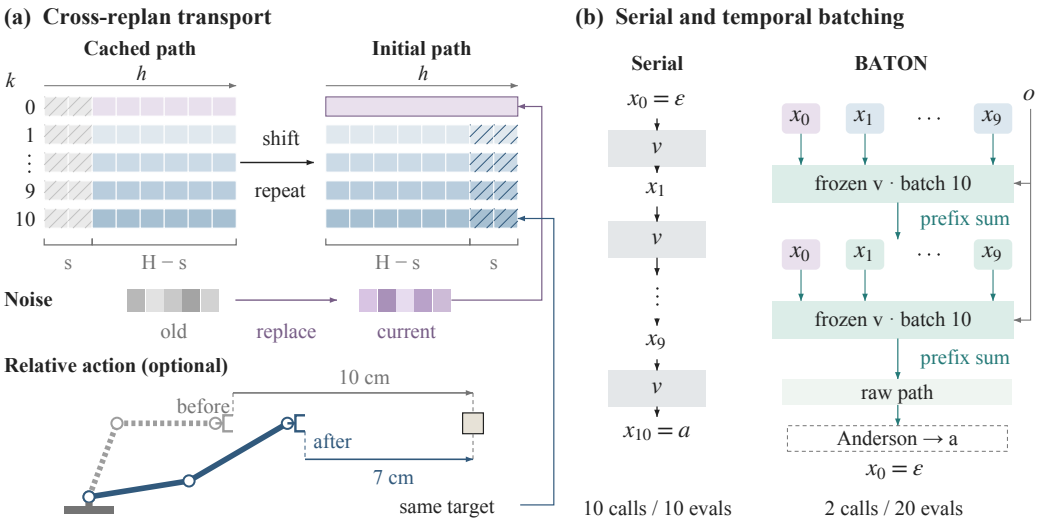

Figure 1: **BATON converts cached solver history into two temporal batches.** (a) Rows are denoising stages; columns are future action positions. Shift every row by the executed span s, repeat the final action to fill the tail, and restore the current noise boundary. Hatching marks consumed actions and the completed tail. The robot inset illustrates the optional coordinate correction: moving 3 cm toward a fixed target changes its relative displacement from 10 to 7 cm. (b) Serial evaluates one state per call. BATON evaluates all ten states together in each of two rounds, integrates their outputs by a prefix sum, and applies Anderson to obtain the action chunk a. Here o is the current observation context and v is the frozen velocity model.

In this paper, we address the few-round regime by introducing BATON, a training-free sampler that transports solver history across replans, as a relay runner hands over the baton. Instead of starting each parallel solve from noise, we shift every cached denoising node by the executed span, complete the new tail, and restore the current noise boundary (Figure 1). The transported trajectory initializes two parallel Picard rounds, and a depth-one Anderson extrapolation combines them without another model call. We hypothesize that transport places the solver close to the current serial solution, so that few rounds are needed.

Transport adds neither parameters nor model calls, and the checkpoint is left unchanged. Ten dependent calls thus become two wider calls that evaluate twenty states in total.

We evaluate BATON with Fast-WAM and $\pi_{0.5}$ in LIBERO-Plus (Fei et al., 2026) and RoboTwin 2.0 (Chen et al., 2025b) simulation. We show that: 1) on 450 paired Fast-WAM instances, BATON reduces median action-expert latency by $2.80\times$ and backend latency by $1.73\times$ relative to ten-step serial sampling, with 311 versus 304 successes, a difference that is not statistically significant; 2) at this two-round budget, removing either transport or Anderson extrapolation lowers success in paired ablations; 3) with shared inputs, transport lowers the two-round error of executed actions at 399 of 408 checkpoints.

Lower latency also changes when a controller can act on its next observation. Executing S actions at native rate f_a leaves S/f_a seconds for the next replan, so faster inference adds timing margin at a fixed stride and can permit a smaller one (Section 2.2). Our analysis bounds the initializer error by plan change and clean-prediction drift, and derives a stride-dependent upper bound under temporal-consistency assumptions. A zero-sensitivity model explains the observed raw-round oscillation and motivates the Anderson correction (Section 3).

2 TRANSPORTED PARALLEL SAMPLING

BATON is a sampling procedure for a frozen receding-horizon policy. Like other parallel samplers, it solves for the whole denoising trajectory as a fixed point. Unlike them, it starts from the previous

replan's trajectory, moved into the current sampling problem. Figure 1 illustrates the idea, introduced next.

2.1 TWO TIME AXES IN RECEDING-HORIZON SAMPLING

Let us consider replan r. A frozen policy receives observation context o_r and generates an action chunk $A \in \mathbb{R}^{H\times d}$, that is, H future actions of dimension d. We write the sampler in flow-matching form. The policy is trained on noisy chunks $x_\sigma = (1-\sigma)A + \sigma\epsilon$ with Gaussian noise ϵ, and it generates a chunk by integrating its learned velocity field v_θ from noise at $\sigma = 1$ to clean actions at $\sigma = 0$:

$$\frac{\mathrm{d}x_\sigma}{\mathrm{d}\sigma} = v_\theta(x_\sigma, \sigma; o_r), \qquad x_1 = \epsilon_r, \tag{1}$$

where ϵ_r is the noise prescribed for replan r. On a grid $1 = \sigma_0 > \sigma_1 > \cdots > \sigma_N = 0$, the serial sampler takes N Euler steps and produces the trajectory $X_{r,0:N}$, with $X_{r,0} = \epsilon_r$ and $X_{r,i}$ approximating x_{σ_i}:

$$X_{r,i+1} = X_{r,i} + \Delta_i v_{r,i}(X_{r,i}; o_r), \qquad \Delta_i = \sigma_{i+1} - \sigma_i, \qquad i = 0, \dots, N-1, \tag{2}$$

where $v_{r,i}(\cdot\,; o_r) = v_\theta(\cdot, \sigma_i; o_r)$. We call σ_k and $\alpha_k = 1 - \sigma_k$ the noise and clean coefficients of node k. Deterministic diffusion samplers take the same form with their own step direction $v_{r,i}$ and coefficients satisfying $\sigma_0 = 1$, $\sigma_N = 0$, $\alpha_0 = 0$, and $\alpha_N = 1$. Here i indexes *denoising time*, whereas each node holds H positions along *action time*. After the first s_r actions are executed, the next horizon overlaps the remaining $H - s_r$ positions. BATON reuses this overlap; the checkpoint and the serial fixed point stay unchanged.

Because one batched call can evaluate several states, we count dependent network calls and state evaluations separately. Unless noted, we use $N = 10$, $H = 32$, and executed stride $s_r = S = 10$.

2.2 FROM INFERENCE LATENCY TO FEEDBACK FREQUENCY

Executing S actions at native action rate f_a gives a nominal replanning rate $f_r = f_a/S$. A look-ahead executor starts inference while the current plan runs and installs the new chunk after S ticks. For observation-to-ready service latency ℓ, the update deadline and the buffer horizon are

$$\ell \le D_S = \frac{S}{f_a}, \qquad f_r = \frac{f_a}{S}, \qquad D_H = \frac{H}{f_a}. \tag{3}$$

A late result extends the old plan, and after D_H the action buffer runs out (Appendix C.7).

For a tail budget $\ell_\beta = Q_{1-\beta}(\ell)$, the smallest stride that meets the deadline is $S_\beta = \max\{1, \lceil f_a \ell_\beta \rceil\}$, provided $S_\beta \le H$. Faster inference therefore adds margin at a fixed stride, and it raises the nominal feedback rate only when a smaller integer stride becomes feasible. At $f_a = 20$ Hz, strides $S = 10, 5, 1$ correspond to $2, 4, 20$ Hz and to deadlines of $500, 250, 50$ ms. Smaller strides cost more inference per second, and a higher native rate needs a longer horizon to keep the same lookahead.

2.3 TRANSPORTING THE TRAJECTORY

Let $Y = Y_{r-1,0:N}$ be the trajectory cached at the previous replan: the serial trajectory after the first replan, and the output of Eq. (8) afterwards. Each node Y_k is an action chunk at noise level σ_k. For a chunk A, let A_h denote its action at position h. The shift operator R_s drops the first s actions and fills the tail by repeating the last action:

$$[\mathsf{R}_s A]_h = A_{\min\{h+s, H-1\}}, \qquad h = 0, \dots, H-1. \tag{4}$$

Transport shifts every node by the executed span s_r and then corrects the noise and the action coordinates:

$$(\mathcal{T}_r Y)_k = \mathsf{R}_{s_r} Y_k + \sigma_k\big(\epsilon_r - \mathsf{R}_{s_r} Y_0\big) + \alpha_k C_r, \qquad k = 0, \dots, N. \tag{5}$$

The first term aligns the overlapping actions. The second replaces the shifted old noise $\mathsf{R}_{s_r} Y_0$ with the current noise ϵ_r, in proportion to each node's noise coefficient; since $\sigma_0 = 1$, the transported trajectory starts at $(\mathcal{T}_r Y)_0 = \epsilon_r$. The last term adds a clean-action correction $C_r \in \mathbb{R}^{H\times d}$, which vanishes at the noise end and is fully applied at the clean end, where transport returns $\mathsf{R}_{s_r} Y_N + C_r$.

The correction is needed when actions are expressed relative to the observed robot state. Suppose an action encodes a target state as $a = D(q_{\text{target}} - q_{\text{obs}})$, where q is the robot state, q_{obs} its observed value, and D a diagonal scale. A target planned from state q_{r-1} at the previous replan is then expressed relative to the current state q_r by $[C_r]_h = D(q_{r-1} - q_r)$. Dimensions without such an encoding, including grippers, use zero correction. Fast-WAM on LIBERO-Plus uses $C_r = 0$. Appendix C gives the affine-path analysis, and Appendix B.2 lists the action encodings.

2.4 Two parallel rounds

For the current observation and noise boundary, the serial trajectory is the fixed point of the prefix map

$$[G_r(X)]_0 = \epsilon_r, \qquad [G_r(X)]_{i+1} = \epsilon_r + \sum_{j=0}^{i} \Delta_j v_{r,j}(X_j; o_r). \tag{6}$$

Just as in parallel Picard sampling (Shih et al., 2023; Tang et al., 2024), the N velocity evaluations are independent given a trajectory guess, so each round costs a single batched network call followed by a prefix integration. However, we start from the transported history and run only two rounds:

$$Z_r^{(0)} = \mathcal{T}_r Y_{r-1}, \qquad Z_r^{(1)} = G_r(Z_r^{(0)}), \qquad Z_r^{(2)} = G_r(Z_r^{(1)}). \tag{7}$$

Two rounds need not reach the fixed point; Section 3 bounds the remaining error.

2.5 Deployed output and resets

We then apply depth-one global Anderson extrapolation (Anderson, 1965; Walker & Ni, 2011). Let $F^{(0)} = Z_r^{(1)} - Z_r^{(0)}$ and $\Delta F = (Z_r^{(2)} - Z_r^{(1)}) - F^{(0)}$. On the flattened trajectories we compute

$$\beta = -\frac{\langle F^{(0)}, \Delta F \rangle}{\|\Delta F\|_F^2}, \qquad Y_r = (1-\beta) Z_r^{(1)} + \beta Z_r^{(2)}. \tag{8}$$

We execute the configured prefix of $Y_{r,N}$ and cache the full Y_r for the next replan. The extrapolation reuses the two existing updates and adds no network call. We add it because two raw rounds lose closed-loop success (Section 5); Appendix B.3 specifies the numerical handling of β.

The first replan runs the original serial solver to supply the initial actions and trajectory, and later replans follow the fixed two-call schedule. Reset rules and pseudocode are in Appendix B.

Why wider calls can be faster. Let $t_\theta(M)$ be network latency at effective token dimension M. The two schedules have approximate critical paths

$$T_{\text{serial}} \simeq N t_\theta(H) + T_{\text{aux}}, \qquad T_{\text{parallel}} \simeq R t_\theta(NH) + T_{\text{transport}} + T_{\text{aux}}, \tag{9}$$

with $M = H$ for serial sampling and $M = NH$ for temporal batching. For $N = 10$, $H = 32$, and $R = 2$, ten calls at $M = 32$ become two calls at $M = 320$. The batched schedule does twice the work, twenty state evaluations instead of ten, yet it can be faster because wider matrix multiplications reuse parameter tiles across more states and amortize per-call overhead, which shortens the critical path whenever $2t_\theta(320) + T_{\text{transport}} < 10 t_\theta(32)$. Appendix B gives the resource accounting.

3 Analysis

We analyze one replan of the Euler sampler of Eq. (2) in exact arithmetic, with $N \geq 2$ and $\alpha_k = 1 - \sigma_k$. We write $\|U\| = \max_{k,h} \|U_{k,h}\|_2$, and compared trajectories share the current noise. Proofs are in Appendix C.

Structure of the serial trajectory. Let $D_{r,i} = X_{r,i} - \sigma_i v_{r,i}(X_{r,i}; o_r)$ be the clean chunk predicted at node i, the same network output in clean-action form. Substituting $v = (x - D)/\sigma$ into Eq. (2) gives, for $0 < k < N$,

$$X_{r,k} = \sigma_k \epsilon_r + \alpha_k \bar{m}_{r,k}, \qquad \bar{m}_{r,k} = \frac{\sigma_k}{\alpha_k} \sum_{i<k} \Big(\frac{1}{\sigma_{i+1}} - \frac{1}{\sigma_i} \Big) D_{r,i}, \tag{10}$$

and $X_{r,N} = D_{r,N-1}$. Each node thus mixes the current noise with a convex combination of the clean predictions. Transport, Eq. (5), keeps this form: it replaces the noise and shifts the clean part.

The transported initializer. Let $a_r = X^\star_{r,N}$ be the serial chunk, $\eta_r = \|\mathcal{T}_r X^\star_{r-1} - X^\star_r\|$ the transport defect, and $\delta_r = \max_{0\le i<N} \|D_{r,i} - a_r\|$ the drift of the clean predictions during the serial solve.

Proposition 1 (Initializer error). $\|\mathsf{R}_{s_r} a_{r-1} + C_r - a_r\| \le \eta_r \le \|\mathsf{R}_{s_r} a_{r-1} + C_r - a_r\| + \delta_{r-1} + \delta_r$.

On 28 Fast-WAM checkpoints from the common-input diagnostic (Section 4), η_r equals the lower bound, the change of the policy's plan (median ratio 1.000). With a fixed noise draw, the plan change is at most $\gamma + (LW + V_a)S/f_a$ if the chunk is L-Lipschitz in the observation, agrees with its shifted plan up to γ when the observation evolves as planned, unplanned observation change accrues at rate at most W, and adjacent planned actions differ by at most V_a/f_a (Appendix C.3). The defect bound is therefore affine in the stride S.

Two raw rounds. We do not assume that G_r contracts, only that each clean prediction is Lipschitz, $\|D_{r,j}(x) - D_{r,j}(y)\| \le \ell_j \|x - y\|$, with ℓ_j unrestricted. Let $E^{(m)}_k = \|Z^{(m)}_{r,k} - X^\star_{r,k}\|$ be the error of node k after m rounds ($E^{(m)}_0 = 0$) and $b_j = (\sigma_j - \sigma_{j+1})/\sigma_j \in (0, 1]$.

Theorem 1 (Finite-round error). *Let $M \in \mathbb{R}^{N\times N}$ be strictly lower triangular with $M_{kj} = m_j = b_j(1 + \ell_j)$ for $1 \le j < k \le N$. For $R \ge 1$, elementwise over $k = 1, \dots, N$,*

$$E^{(R)} \le M^R E^{(0)}, \qquad E^{(0)}_k \le e_{r-1} + \eta_r, \tag{11}$$

where $e_{r-1} = \|Y_{r-1} - X^\star_{r-1}\|$ is the error of the cache. Moreover, $(M^R)_{kj} = m_j\, s_{R-1}(m_{j+1}, \dots, m_{k-1})$, where s_{R-1} sums the products of all $R-1$ distinct arguments ($s_0 = 1$). It vanishes for $k - j < R$, so nodes $k \le R$ are exact.

For a fixed round budget, the bound is linear in $e_{r-1} + \eta_r$. It needs no contraction; a weighted norm recovers one with any rate $q \in (0, 1)$, at a q-dependent conversion factor (Appendix C). Rebuilding the nodes through Eq. (10) from the same network calls removes the 1 in m_j, so each round contracts by $\max_j \ell_j$ (Appendix C.6).

Why Anderson can help raw rounds. Since $\sigma_N = 0$, $b_{N-1} = 1$ for any schedule, so $\|M\|_\infty \ge 1$ and the bound does not certify that a raw round shrinks the error. For a concentrated policy, whose clean predictions barely depend on the noisy input, the error map linearizes to $-B$ with $B_{kj} = b_j$ below the diagonal, so $e^{(1)} \approx -Be^{(0)}$ and $e^{(2)} \approx B^2 e^{(0)}$ alternate in sign. If the clean predictions are input-independent and each action coordinate of $e^{(0)}$ keeps one sign across nodes, then $\beta \in (0, 1]$ (Appendix C.5), and Anderson forms a convex combination of the two rounds. For a clean-action offset, B alone gives $\beta = 0.64$ on our schedule. In the 450 headline episodes, the logged β lies in $(0, 1)$ at all 11,948 BATON replans (median 0.668), and on the 28 checkpoints the first-round error points against the initial error in every case. This supports oscillation cancellation as an explanation of Anderson's ablation gain (Section 5).

4 EXPERIMENTAL SETUP

We evaluate frozen Fast-WAM on 450 LIBERO-Plus instances (Fei et al., 2026) with $N = 10$, $H = 32$, $S = 10$, and $C_r = 0$, on NVIDIA H100 80GB GPUs. Serial sampling and BATON share the checkpoint, initialization, preprocessing, and sampling-noise key. After a common serial bootstrap, each method executes its own actions, and BATON executes and caches the Anderson output of Eq. (8). We also evaluate $\pi_{0.5}$ on LIBERO-Plus and both policies on RoboTwin 2.0 (Black et al., 2025a; Chen et al., 2025b). Each study keeps its own instances and serial reference (Appendix D.1).

Ablations. A separate 450-instance study removes one component at a time. Two raw rounds start from current noise (*cold*), from the previous trajectory copied unchanged (*unaligned*), or from Eq. (5) (*transport*); further arms add Anderson to the last two and run three raw rounds. Each arm feeds back its own output (Appendix D.3).

Sampling budgets. To compare BATON with other ways of spending the sampling budget, a 450-instance sweep covers serial $N \in \{2, 3, 5, 10, 20\}$, warm serial $K \in \{1, 2, 3\}$, BATON $N \in$

Table 1: **Two temporal batches accelerate the ten-step Fast-WAM policy.** BATON solves 311 of 450 paired LIBERO-Plus instances versus serial's 304, with action-expert and backend speedups of $2.804\times$ and $1.730\times$. Times are steady-state medians per action chunk. The paired success difference is $+1.56$ pp (95% CI $[-2.29, +5.62]$; exact McNemar $p = 0.349$).

	Serial	BATON	Change
Successes / 450	304	311	$+7$
Action Expert CUDA (ms)	29.919	10.669	$2.804\times$ faster
Steady backend wall (ms)	43.248	25.003	$1.730\times$ faster
Dependent model calls	10	2	$5\times$ fewer
Denoiser-state evaluations	10	20	$2\times$ more

$\{3, 5, 10\}$, RTI-DP-clip $K \in \{1, 2, 3\}$, and FALCON $\epsilon \in \{0.005, 0.01, 0.02\}$. Warm serial is our re-noising control. RTI-DP-clip (Duan et al., 2025) shifts clean final actions, repeats the tail, and applies the last K native flow updates. FALCON (Chen et al., 2025a) screens cached noisy candidates under the current observation and resumes a serial suffix. Both published methods are adapted to the frozen Fast-WAM flow sampler with $H = 32$, $S = 10$, and no retraining. Latency uses the 322 instances run on a single H100 host and excludes the first replan; implementation details are in Appendix D.5.

Common-input diagnostic. To measure initializer error without closed-loop feedback, a serial policy drives 28 standard LIBERO tasks (Liu et al., 2023), seven per suite, with $H = 32$ and $S = 10$ on H20 GPUs. At each later replan, the three initializers share the observation, noise, schedule, previous serial trajectory, and serial reference, and their outputs are compared after zero to three raw rounds without being executed, over 408 checkpoints. Task summaries weight tasks equally and use 95% bootstrap intervals resampled within suites (Appendix D.4).

Metrics. We report task success with paired confidence intervals and tests. *Action Expert CUDA* is solver time measured with CUDA events. *Steady backend wall* covers context, vision, noise generation, action inference, host transfer, and history cloning. Headline latency is reported per chunk and excludes the first three replans of each rollout; the budget sweep excludes only the bootstrap replan. Physics pauses during inference, so success measures capability at a fixed stride rather than under real-time execution (Appendix D).

5 RESULTS

5.1 SPEED AND CAPABILITY

Table 1 compares BATON with ten-step serial sampling at the primary operating point. We have three observations. First, BATON is faster although it does more work: median action-expert latency drops from 29.919 to 10.669 ms, while the number of evaluated denoiser states doubles. Second, the backend gain is smaller, from 43.248 to 25.003 ms, because context encoding and vision are unchanged; both gains hold at p99 ($2.403\times$ and $1.521\times$; Appendix D.2). Last, success does not differ significantly: BATON and serial solve 311 and 304 instances ($+1.56$ pp, 95% CI $[-2.29, +5.62]$).

Comparisons across sampling budgets. Figure 2 places seventeen configurations on the same 450 instances. BATON $N = 10$ lies on the observed frontier, with 310 successes at 9.210 ms. Doubling the serial depth to $N = 20$ adds two successes at $6.31\times$ the latency ($p = 0.883$). FALCON reaches similar success but is 1.49 to $4.07\times$ slower, and RTI-DP-clip $K = 1$ is faster, at 5.380 ms, with 293 successes. The other frontier points are warm serial $K = 1, 2$, RTI $K = 1$, BATON $N = 3$, and serial $N = 20$. No success contrast against the published-method adaptations is significant after Holm correction across sixteen comparisons (Appendix D.5).

5.2 TRANSPORT IMPROVES THE INITIALIZER

Figure 3 isolates the numerical effect of transport. All initializers see the same inputs and the same previous serial trajectory, and none of their outputs is executed. After two raw rounds, transport error

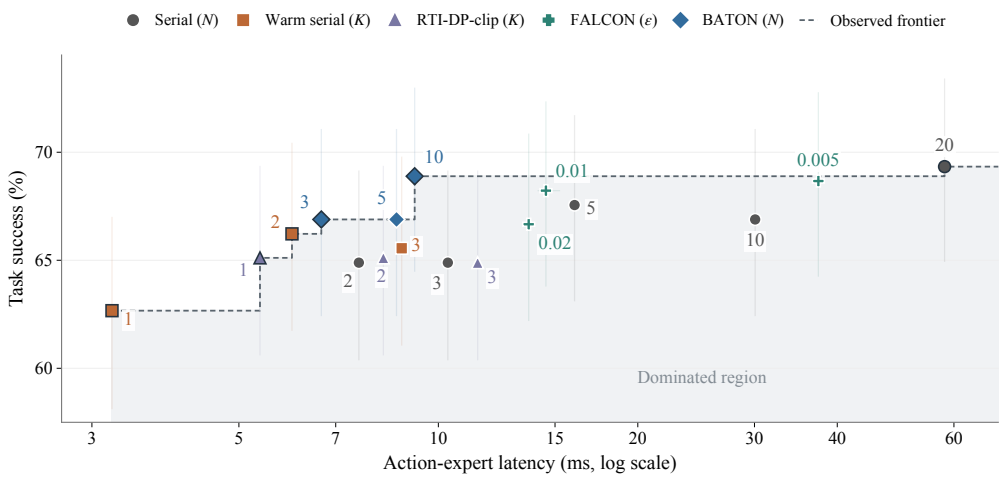

Figure 2: **Success and inference latency across seventeen sampling configurations.** Higher and farther left is better. Success uses 450 shared LIBERO-Plus instances; latency is the median of episode-level Action Expert CUDA medians on the 322 instances run on a single H100 host. Vertical bars are Wilson 95% intervals and horizontal bars are bootstrap 95% intervals. Dark outlines identify the observed frontier; the shading marks its dominated region, where a measured point offers at least as much success at no greater latency. Numbers beside markers give the parameter named in the legend: sampling depth N, serial suffix length K, or FALCON threshold ϵ. BATON uses two rounds at each N. Warm serial is our re-noising control; RTI-DP-clip and FALCON are flow adaptations described in Section 4.

is lowest over the overlapping positions and rises toward the completed tail (Figure 3a). Across all 408 checkpoints, transport lowers executed-prefix error at 399, with a pooled median transport/copy ratio of 0.176 (Figure 3c). These results support our motivation (Section 1) that the transported trajectory starts the solver close to the serial solution. Whether this advantage survives closed-loop execution is studied next.

5.3 Solver components

Table 2 removes one component of BATON at a time on the same 450 instances. Each arm caches its own output as the next replan's history. The serial reference of this campaign solves 306, and all eight configurations are in Table 6.

Table 2: **Transport and Anderson each improve capability at a two-round budget.** All parallel variants use $N = 10$, $H = 32$, $S = 10$, and two rounds; serial uses ten sequential steps. This campaign is separate from Table 1; its serial reference solves 306. Removing transport copies the previous trajectory unchanged; removing Anderson executes the raw second iterate. AA denotes depth-one Anderson. Mean steps use each arm's own successful episodes; the noise-start mean covers only six successes. Complete ablations and paired step comparisons are in Appendix D.3.

Configuration	Successes / 450	Mean steps (success)
Serial ($N = 10$)	306	161.9
Noise start (2 raw rounds)	6	127.5
BATON w/o transport (copy + AA)	294	165.5
BATON w/o Anderson (transport only)	264	159.2
BATON (transport + AA)	**311**	163.2

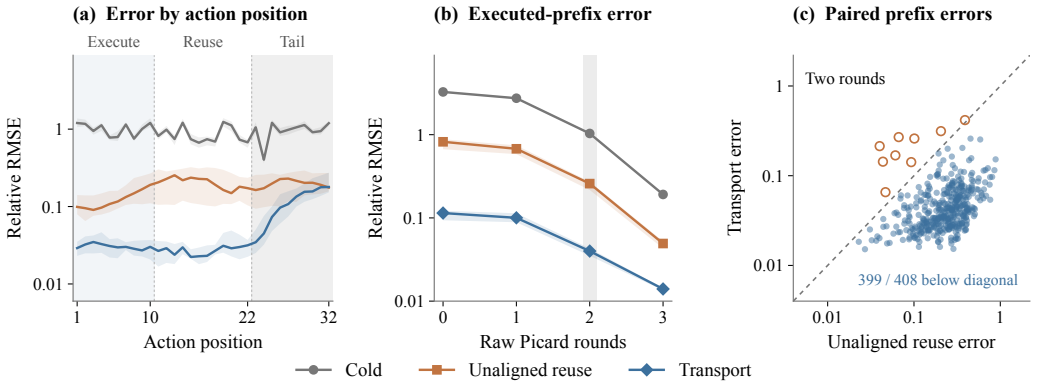

Figure 3: **Transport reduces action error within a two-round budget.** Errors compare continuous actions with the ten-step serial reference under shared inputs and serial-generated history, using raw Picard updates. (a) Error by action position at the first valid-history checkpoint of 28 LIBERO tasks; lines and bands show medians and interquartile intervals. Actions 1 to 10 are executed, 1 to 22 overlap the shifted history, and 23 to 32 fill the new tail. (b) Executed-prefix relative RMSE versus rounds over 408 checkpoints; curves summarize task medians with 95% task-bootstrap intervals. The strip highlights two rounds. (c) Each point compares two-round prefix errors at one checkpoint. Points below the diagonal favor transport; orange points mark the nine checkpoints favoring copy.

History makes two rounds viable. Started from noise, two raw rounds solve only 6 instances. With the previous trajectory as the start, they solve 264 with transport and 306 with unaligned copy. Copy succeeds more often, but in longer episodes: on the 230 instances both solve, transport uses 13.31% fewer control steps.

Anderson repairs the raw rounds. Adding Anderson to transport raises success from 264 to 311 (+10.44 pp; exact McNemar $p = 5.99 \times 10^{-7}$). Three raw rounds reach 309 ($p = 0.871$ against Anderson) with 50% more calls. This matches Section 3: raw rounds oscillate, and Anderson averages them.

Transport helps once the rounds are repaired. With Anderson fixed, transport raises success over copy from 294 to 311 (+3.78 pp; 95% CI $[0.44, 7.11]$; $p = 0.036$), with nearly equal episode lengths on joint successes (159.0 and 158.7 steps). Removing either component therefore lowers success at the two-round budget.

Stride. In exploratory probes at $S = 5$ and $S = 20$ (nominal 4 and 1 Hz), serial and BATON solve 314 and 319, and 244 and 250, of 450. Both lose success at the larger stride, so sampler latency should be read together with the stride. The main study fixes $S = 10$ (Appendix D.8).

5.4 PORTABILITY AND COST

BATON transfers to another policy and benchmark without a significant change in success (Appendix Table 10). With $\pi_{0.5}$ on LIBERO-Plus, serial and BATON solve 423 and 427 of 450 cases ($p = 0.523$), and the full policy call becomes $1.30\times$ faster. On RoboTwin 2.0, Fast-WAM solves 125 of 150 stable-fixture cases with both samplers ($p = 1.0$), with $2.74\times$ action-expert and $1.68\times$ backend speedups, close to its LIBERO-Plus values, and $\pi_{0.5}$ solves 67 of 150 with both samplers. Evaluation sets and action encodings are in Appendix D.

The saving grows with depth: under matched inputs, the action expert is $1.95\times$, $2.93\times$, and $3.80\times$ faster at $N = 5, 10, 20$ (Appendix D.6). The wider batches cost little memory: 30.584 MiB of eager allocation next to 1.902 GiB of BF16 action-expert weights, and 10.000 MiB of reserved memory on the CUDA-Graph path (Appendix B.4).

6 Discussion and Conclusion

In Fast-WAM, most backend CUDA time is spent in ten narrow, dependent calls to the action expert. BATON shortens this chain using structure that receding-horizon control already provides: consecutive sampling problems overlap, so the previous solver trajectory is a good starting point for the next one. With this initializer, two parallel rounds and a zero-call Anderson update replace ten serial calls on the full sampling grid. On the primary evaluation, this gives $2.80\times$ action-expert and $1.73\times$ backend speedups with 311 versus 304 successes. The completed sweep places BATON on the observed Pareto frontier alongside lower-latency warm serial and RTI configurations; the FALCON adaptations reach similar success at higher measured latency. The preferred sampler therefore depends on the inference budget.

On the other hand, we note the limits of this evidence. Capability is measured at a fixed stride in simulation that pauses during inference; running at a smaller stride on a robot, where latency changes the observations, remains to be tested. Our bounds assume only Lipschitz clean predictions, but our account of the Anderson step is a linearization for concentrated policies, checked against logged coefficients rather than proved for the deployed output. Perception bounds the share of backend latency that any sampler can remove, and longer horizons increase the work per call. We hope that carrying the full solver trajectory between replans will be useful for other iterative policies.

AI use statement

Generative AI tools were used for literature organization, proof checking, code assistance, and drafting and editing text. The authors verified cited sources, mathematical arguments, implementation details, and empirical claims, and take responsibility for the paper's contents.

Ethics statement

This work studies inference scheduling for robot policies. The reported experiments are conducted in simulation and do not involve human subjects. Faster inference does not by itself establish safe robot behavior: our solver analysis is not a plant-stability or collision-avoidance guarantee, and deployment on physical systems requires independent safety constraints, monitoring, and validation. We report failed rollouts and computational overheads rather than treating latency as a safety proxy.

Reproducibility statement

Section 2 specifies the execution transformation and its state-reset rules; Appendix B gives pseudocode and work/dependency accounting. Appendix C contains full proofs and the checks behind them. Sections 4 and 5 define paired evaluation units and latency scopes, and Appendix D records counts, confidence procedures, and protocol boundaries. Code, frozen protocols, raw per-rollout records, and aggregation scripts will be included in the anonymous supplement.

References

Donald G. Anderson. Iterative procedures for nonlinear integral equations. *Journal of the ACM*, 12 (4):547–560, 1965. doi: 10.1145/321296.321305.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: 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. PMLR, 2025a. URL https://proceedings.mlr.press/v305/black25a.html.

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. Curran Associates, Inc., 2025b. doi: 10.52202/085713-1122. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/300ccb2187dedd4edcc07f7e76d8e553-Abstract-Conference.html.

Haojun Chen, Minghao Liu, Chengdong Ma, Xiaojian Ma, Zailin Ma, Huimin Wu, Yuanpei Chen, Yifan Zhong, Mingzhi Wang, Qing Li, and Yaodong Yang. Falcon: Fast visuomotor policies via partial denoising. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 8552–8573. PMLR, 2025a. URL https://proceedings.mlr.press/v267/chen25ao.html.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, Weiliang Deng, Yubin Guo, Tian Nian, Xuanbing Xie, Qiangyu Chen, Kailun Su, Tianling Xu, Guodong Liu, Mengkang Hu, Huan-ang Gao, Kaixuan Wang, Zhixuan Liang, Yusen Qin, Xiaokang Yang, Ping Luo, and Yao Mu. RoboTwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025b. URL https://arxiv.org/abs/2506.18088.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin C. M. Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.026.

Yufei Duan, Hang Yin, and Danica Kragic. Real-time iteration scheme for diffusion policy. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 11758–11764. IEEE, 2025. doi: 10.1109/IROS60139.2025.11247391. URL https://arxiv.org/abs/2508.05396.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, Jinlan Fu, Jingjing Gong, and Xipeng Qiu. LIBERO-Plus: A progressive robustness benchmark for visual-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 38574–38583, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Fei_LIBERO-Plus_A_Progressive_Robustness_Benchmark_for_Visual-Language-Action_Models_CVPR_2026_paper.html.

Sigmund H. Høeg, Yilun Du, and Olav Egeland. Fast policy synthesis with variable noise diffusion models. In *2025 IEEE International Conference on Robotics and Automation*, pp. 4821–4828. IEEE, 2025. doi: 10.1109/ICRA55743.2025.11127858. URL https://arxiv.org/abs/2406.04806.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 44776–44791. Curran Associates, Inc., 2023. doi: 10.52202/075280-1939. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/8c3c666820ea055a77726d66fc7d447f-Abstract-Datasets_and_Benchmarks.html.

Aaditya Prasad, Kevin Lin, Jimmy Wu, Linqi Zhou, and Jeannette Bohg. Consistency policy: Accelerated visuomotor policies via consistency distillation. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.071.

Andy Shih, Suneel Belkhale, Stefano Ermon, Dorsa Sadigh, and Nima Anari. Parallel sampling of diffusion models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 4263–4276. Curran Associates, Inc., 2023. doi: 10.52202/075280-0188.

Zhiwei Tang, Jiasheng Tang, Hao Luo, Fan Wang, and Tsung-Hui Chang. Accelerating parallel sampling of diffusion models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 47800–47818. PMLR, 2024. URL https://proceedings.mlr.press/v235/tang24f.html.

Homer F. Walker and Peng Ni. Anderson acceleration for fixed-point iterations. *SIAM Journal on Numerical Analysis*, 49(4):1715–1735, 2011. doi: 10.1137/10078356X.

Zhendong Wang, Max Li, Ajay Mandlekar, Zhenjia Xu, Jiaojiao Fan, Yashraj Narang, Linxi Fan, Yuke Zhu, Yogesh Balaji, Mingyuan Zhou, Ming-Yu Liu, and Yu Zeng. One-step diffusion policy: Fast visuomotor policies via diffusion distillation. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 63399–63416. PMLR, 2025. URL https://proceedings.mlr.press/v267/wang25ba.html.

Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-WAM: Do world action models need test-time future imagination? *arXiv preprint arXiv:2603.16666*, 2026. URL https://arxiv.org/abs/2603.16666.

A Related Work

Accelerating frozen robot policies. Diffusion Policy, $\pi_{0.5}$, and Fast-WAM generate actions iteratively (Chi et al., 2023; Black et al., 2025a; Yuan et al., 2026). Consistency and one-step policies reduce sampling through training, and Streaming Diffusion Policy trains partially denoised actions across cycles (Prasad et al., 2024; Wang et al., 2025; Høeg et al., 2025). These methods change the model; BATON keeps the checkpoint fixed. Closest to our work, RTI-DP shifts the previous clean action chunk to initialize a shortened sequential denoising, and FALCON selects historical partially denoised candidates under the current observation (Duan et al., 2025; Chen et al., 2025a). Both reuse actions; BATON reuses every node of the solver trajectory and updates denoising time in parallel.

Parallel sampling. ParaDiGMS introduced Picard-parallel diffusion sampling, including experiments with Diffusion Policy, and ParaTAA developed triangular Anderson acceleration, initializing a solve from the trajectory of a related text-to-image prompt (Shih et al., 2023; Tang et al., 2024). BATON instead transports the solver trajectory across robot replans, after part of the chunk has been executed, and then runs two parallel rounds. Real-time chunking addresses how inference and control overlap (Black et al., 2025b); faster chunk production is complementary to it.

B Algorithm and Implementation

B.1 Reference procedure

Inputs: current observation o_r, boundary noise ϵ_r, executed span s_r, cached trajectory Y_{r-1}, and schedules $(\sigma_{0:N}, \alpha_{0:N})$.
On reset or missing history: run the original serial N-step sampler; execute its clean action prefix; cache the full trajectory.
Otherwise:

1. Apply R_{s_r} with repeat-last completion to every denoising node.
2. Add $\sigma_k(\epsilon_r - \mathsf{R}_{s_r} Y_{r-1,0})$ to restore the current noise boundary.
3. Add $\alpha_k C_r$ on affine relative-action dimensions under the sampler's clean schedule; use zero on unsupported dimensions.
4. Apply two raw prefix Picard maps, each using one N-state temporal Action/DiT batch.
5. For the full evaluated method, compute the zero-call extrapolation in Eq. (8); for a raw-core ablation, omit this step.
6. Execute the configured clean prefix and cache the output trajectory. Full BATON executes and caches the extrapolated Y_r; the theorem-aligned raw core instead executes and caches $Z_r^{(2)}$.

Figure 4: **BATON at one replan.** Initialize from cached history, apply two temporal batches, and execute the action prefix after Anderson extrapolation. Serial initialization supplies the first trajectory and handles resets.

B.2 Implementation conventions

The headline Fast-WAM sampler is deterministic and uses $C_r = 0$. For affine relative-action encodings, the coordinate correction applies to the supported continuous dimensions; discrete grippers and other dimensions use zero. On RoboTwin 2.0, $\pi_{0.5}$ uses a scaled approximate correction (Appendix D.7). Exact overlap alignment requires the transport shift to equal the executed span; the Fast-WAM RoboTwin configuration violates this (Appendix D.7). Stochastic samplers require a consistent complete forcing-noise tape.

Episode or task resets, action-space changes, and invalid or non-finite history clear the cache and trigger serial initialization. Valid-history replans follow a fixed schedule, with no oracle solve and no success-dependent early exit.

B.3 Anderson extrapolation

The full method uses $R = 2$ and a depth-one global Anderson extrapolation with no extra Action/DiT call (Anderson, 1965; Walker & Ni, 2011). For $R \geq 2$, flatten the trajectory and set $F^{(0)} = Z^{(R-1)} - Z^{(R-2)}$ and $\Delta F = (Z^{(R)} - Z^{(R-1)}) - F^{(0)}$. The FP32 least-squares coefficient is $\beta = -\langle F^{(0)}, \Delta F\rangle / \|\Delta F\|_F^2$ when the denominator exceeds machine epsilon; otherwise $\beta = 1$ returns the latest raw iterate $Z^{(R)}$. The coefficient is not clipped and has no ridge regularization. The output $X^A = (1-\beta)Z^{(R-1)} + \beta Z^{(R)}$ keeps node zero at the current noise. The full method executes X^A and caches it for the next replan. Appendix C.5 explains the observed range of β; no error guarantee is claimed for X^A.

B.4 Work, dependency, and memory

Table 3: **Network calls, state evaluations, and history storage.** Logical states sum the denoising states processed across calls. Two BATON batches process twenty states using two dependent model calls.

Schedule	Dependent calls	Logical states	Relative work	History
Serial	10	10	$1\times$	none required
BATON raw core	2	20	$\approx 2\times$	full $11 \times H \times d$ path
BATON + extrapolation	2	20	$\approx 2\times$	plus low-cost extrapolation

The two-batch schedule wins only when $2t_\theta(10H)$ plus transport and integration is below $10t_\theta(H)$, which must be measured for each device and policy stack. We have not measured HBM traffic, so the reduction from N to R parameter-serving passes is execution accounting.

Table 4: Incremental CUDA allocator footprint (MiB), measured separately for eager and CUDA Graph execution.

Path and metric	Serial	Candidate	Candidate minus serial
Eager allocated	3.292	33.876	+30.584
Graph allocated	35.292	33.111	−2.181
Graph reserved	40.000	50.000	+10.000

Eager allocated memory grows $10.3\times$ but includes activations, workspace, and output, so 30.584 MiB is a conservative upper bound on activation overhead. Relative to one $B = 10$ round, the second round plus extrapolation adds 5,120 allocated bytes in either path and zero reserved bytes.

C Theory and Proofs

This appendix proves the statements of Section 3 and records the checks behind them. As there, $\|U\| = \max_{k,h} \|U_{k,h}\|_2$, and E_k denotes the error $\max_h \|\cdot\|_2$ of node k.

C.1 Structure of the serial trajectory

With $D_i = X_i - \sigma_i v_i(X_i)$, the Euler step reads $X_{i+1} = X_i + \Delta_i(X_i - D_i)/\sigma_i = (\sigma_{i+1}/\sigma_i)X_i - (\Delta_i/\sigma_i)D_i$. Dividing by $\sigma_{i+1} > 0$ gives

$$\frac{X_{i+1}}{\sigma_{i+1}} = \frac{X_i}{\sigma_i} + \Big(\frac{1}{\sigma_{i+1}} - \frac{1}{\sigma_i}\Big) D_i, \tag{12}$$

and summing from $X_0/\sigma_0 = \epsilon$ gives Eq. (10). The weights $w_{k,i} = \sigma_k(1/\sigma_{i+1} - 1/\sigma_i)$ are positive and sum to $\sigma_k(1/\sigma_k - 1) = \alpha_k$. At the last step $\Delta_{N-1} = -\sigma_{N-1}$, so $X_N = D_{N-1}$. We write $\bar{m}_N = D_{N-1} = a$. The identity holds for any network; no property of the policy is used.

C.2 Transport

Proposition 2 (Transport)**.** *For Eq. (5), $(\mathcal{T}_r Y)_0 = \epsilon_r$ and $(\mathcal{T}_r Y)_N = \mathsf{R}_{s_r} Y_N + C_r$. If $U_0 = V_0$, then $\|\mathcal{T}_r U - \mathcal{T}_r V\| \le \|U - V\|$. If every node has the form $Y_k = \sigma_k \eta + \alpha_k A$, then $(\mathcal{T}_r Y)_k = \sigma_k \epsilon_r + \alpha_k(\mathsf{R}_{s_r} A + C_r)$.*

Proof. The endpoint identities follow from $(\sigma_0, \alpha_0) = (1, 0)$ and $(\sigma_N, \alpha_N) = (0, 1)$. If $U_0 = V_0$, the noise terms cancel and $\mathcal{T}_r U - \mathcal{T}_r V = \mathsf{R}_{s_r}(U - V)$ node by node. Each output position of the repeat-last shift copies one input position, so the shift does not increase the block maximum. For the last claim, substitute Y_k into Eq. (5); the $\sigma_k \mathsf{R}_{s_r} \eta$ terms cancel. □

By Eq. (10), the last form is exact for the serial trajectory whenever all clean predictions coincide with a single chunk A. Deterministic sampling alone does not imply coincident clean predictions. Proposition 1 quantifies the general case.

C.3 Initializer error

Proof of Proposition 1. Node zero has zero defect. For $1 \le k \le N$, Eq. (10) for both replans and the rebase term of Eq. (5) give

$$(\mathcal{T}_r X^\star_{r-1})_k - X^\star_{r,k} = \alpha_k\big(\mathsf{R}_{s_r} \bar{m}_{r-1,k} + C_r - \bar{m}_{r,k}\big). \tag{13}$$

At $k = N$ this equals $\mathsf{R}_{s_r} a_{r-1} + C_r - a_r$, which gives the lower bound. Since $\bar{m}_{r,k}$ is a convex combination of the $D_{r,i}$, $\|\bar{m}_{r,k} - a_r\| \le \delta_r$, and likewise for $r - 1$. The shift is non-expansive and $\alpha_k \le 1$, which gives the upper bound. □

Plan change under continuous observations. Assume $1 \le s_r < H$ and a position-independent correction $[C_r]_h = c_r$. Fix the noise draw, as in the Fast-WAM protocol, so that the chunk is a function $a = \mathcal{A}(o)$ of the observation, and let $\hat{o}_r$ be the observation that the plan made at $r - 1$ anticipates after s_r actions. Assume: (i) $\|\mathcal{A}(o) - \mathcal{A}(o')\| \le L\|o - o'\|$; (ii) $\|[\mathsf{R}_{s_r}\mathcal{A}(o_{r-1}) + C_r - \mathcal{A}(\hat{o}_r)]_h\|_2 \le \gamma$ for the overlapping positions $h < H - s_r$; (iii) $\|o_r - \hat{o}_r\| \le W s_r / f_a$; (iv) $\|[\mathcal{A}(o)]_h - [\mathcal{A}(o)]_{h'}\|_2 \le V_a|h - h'|/f_a$. On overlapping positions, (i) to (iii) give $\|[\mathsf{R}_{s_r} a_{r-1} + C_r - a_r]_h\|_2 \le \gamma + LW s_r / f_a$. On a tail position $h \ge H - s_r$, repeat-last completion copies position $H - s_r - 1$, and C_r is the same at every position, so (iv) adds at most $V_a s_r / f_a$. Hence

$$\|\mathsf{R}_{s_r} a_{r-1} + C_r - a_r\| \le \gamma + (LW + V_a)\frac{s_r}{f_a}. \tag{14}$$

Assumption (ii) states that the policy is consistent with its own plan; γ collects the remaining inconsistency, including a mode switch. Combining Eq. (14) with Proposition 1 adds $\delta_{r-1} + \delta_r$ to this bound. A comparison across strides requires uniform γ, L, W, V_a and drift bounds; it does not prove monotonicity of the realized defect. With changing noise, an additional bound on noise-induced plan change is required.

Measurement. The 28 saved checkpoints of the common-input diagnostic (Appendix D.4) contain the previous and current serial trajectories and raw rounds zero to three from transport. We recover $D_{r,i}$ from consecutive serial nodes. Both bounds of Proposition 1 hold at all 28 checkpoints, and η_r equals the lower bound (median ratio 1.000, median $\eta_r = 0.522$ in normalized action units); the maximum defect is attained at the clean endpoint. This does not imply zero drift: the median δ_{r-1} and δ_r are 0.685 and 0.691.

C.4 FINITE-ROUND ERROR

Proof of Theorem 1. Let $e = Z - X_r^\star$ with $e_0 = 0$. Since $G_r(X_r^\star) = X_r^\star$ and $v = (x - D)/\sigma$,

$$[G_r(Z) - X_r^\star]_k = -\sum_{j<k} b_j \big(e_j - [D_j(Z_j) - D_j(X_{r,j}^\star)]\big), \tag{15}$$

where D_j abbreviates $D_{r,j}$. Taking norms, using the Lipschitz bound, and noting $e_0 = 0$, $E'_k \le \sum_{1\le j<k} b_j(1+\ell_j)E_j = (ME)_k$. Because M is nonnegative, iterating gives $E^{(R)} \le M^R E^{(0)}$. Each path from j to k in M^R passes through strictly increasing indices $j = i_0 < i_1 < \cdots < i_{R-1} < k$, and $M_{i_{t+1}i_t} = m_{i_t}$ depends only on the column, so the path contributes $m_j m_{i_1} \cdots m_{i_{R-1}}$. Summing over the choices of $i_1 < \cdots < i_{R-1}$ in $\{j+1, \ldots, k-1\}$ gives $m_j\, s_{R-1}(m_{j+1}, \ldots, m_{k-1})$; no such choice exists when $k - j < R$. Thus nodes $k \le R$ are exact, and $M^N = 0$. The serial fixed point is unique by forward substitution in Eq. (2). For the initializer, Proposition 2 and the triangle inequality give $\|\mathcal{T}_r Y_{r-1} - X_r^\star\| \le \|Y_{r-1} - X_{r-1}^\star\| + \eta_r$; both cached and serial trajectories start at ϵ_{r-1}, and Anderson preserves node zero. □

Weighted contraction. For any $q \in (0, 1)$, set $\omega_1 = 1$ and $\omega_{k+1} = \omega_k(1 + m_k/q)$ for $1 \le k < N$. Telescoping gives $\sum_{1\le j<k} m_j\omega_j = q(\omega_k - 1) \le q\omega_k$, so $\max_k E'_k/\omega_k \le q \max_k E_k/\omega_k$. The same inequality holds between any two boundary-matched trajectories on the domain where the stated Lipschitz constants apply. Converting back gives $E_N^{(R)} \le \omega_N q^R \max_k E_k^{(0)}/\omega_k$. The weights depend on q; choosing a small q does not by itself give a small bound in the original norm.

Across replans. Consider the raw-core variant that caches $Z_r^{(R)}$, and assume a common matrix M bounds every replan. Define $e_r^{\mathrm{raw}} = \|Z_r^{(R)} - X_r^\star\|$ and $c_R = \max_k \sum_j (M^R)_{kj}$. Then $e_r^{\mathrm{raw}} \le c_R(e_{r-1}^{\mathrm{raw}} + \eta_r)$. If $c_R < 1$ and $\eta_r \le \bar\eta$, iteration yields $\limsup_r e_r^{\mathrm{raw}} \le c_R\bar\eta/(1 - c_R)$. These are errors relative to the serial solution at each current observation, not differences between autonomous closed-loop rollouts. The recursion does not apply unchanged when the cache contains Anderson outputs, and c_R need not be below one.

C.5 RAW ROUNDS AND THE ANDERSON COEFFICIENT

If each D_j is input-independent on the domain, Eq. (15) is exactly linear: $e^{(m+1)} = -Be^{(m)}$ with $B_{kj} = b_j$ for $1 \le j < k$. The residuals of Eq. (8) are $F^{(0)} = -(I+B)e^{(0)}$ and $\Delta F = (I+B)^2 e^{(0)}$, so, for $e^{(0)} \ne 0$, with $u = (I + B)e^{(0)}$, the flattened Euclidean inner product and Frobenius norm give

$$\beta = \frac{\langle u, (I+B)u\rangle}{\|(I+B)u\|_F^2}, \qquad 0 < \beta < 1 \iff \langle u, Bu\rangle > \max\{-\|u\|_F^2, -\|Bu\|_F^2\}. \tag{16}$$

If each action coordinate of $e^{(0)}$ keeps one sign across nodes, as for a clean-action offset $\alpha_k \Delta a$, then so does u, and $\langle u, Bu\rangle \ge 0$ because B is nonnegative. Hence $0 < \beta \le 1$, with $\beta < 1$ if and only if $Bu \ne 0$. For example, a nonzero error confined to node N gives $Bu = 0$ and $\beta = 1$. These statements concern the analytic quotient. The implementation returns $\beta = 1$ when its denominator is below the threshold in Appendix B.3, including $e^{(0)} = 0$. Small nonzero input derivatives give an approximation, not an unconditional interval guarantee. The coefficient depends only on the schedule and the error profile. For the Fast-WAM schedule, the profile α_k gives $\beta = 0.640$, and random constant-sign profiles give a 5 to 95% range of $[0.566, 0.818]$. The logged values in the headline campaign have median 0.668 and a 5 to 95% range of $[0.630, 0.794]$; the seed-43 extension gives nearly identical quantiles.

The linear system also explains the oscillation. The residual satisfies $Z - G_r(Z) = (I+B)(Z - X_r^\star)$, and $(I + B)^{-1} = I - W$, where $W_{kj} = b_j \prod_{j<\ell<k}(1 - b_\ell)$ for $1 \le j < k \le N$ and zero otherwise. For $k < N$, these are the unnormalized weights $w_{k,j}$ above. At $k = N$, only $W_{N,N-1} = 1$ is nonzero, avoiding division by $\sigma_N = 0$. The row sums of W are at most one, so $\|(I + B)^{-1}\|_\infty \le 2$. Serial sampling applies this bounded inverse by forward substitution. Raw Picard instead expands it as $\sum_m (-B)^m$, which ends at $m = N - 1$ because $B^N = 0$. Its terms need not decay monotonically, as $\|B\|_\infty \ge b_{N-1} = 1$; on our schedule $\|B\|_\infty = 1.89$. Two raw updates yield the errors $-Be^{(0)}$ and

$B^2 e^{(0)}$. Anderson minimizes the residual surrogate $\|F^{(0)} + \beta \Delta F\|_F^2$, not the error of the deployed mixture. It need not improve on the second raw iterate: for $N = 2$, $B_{21} = 1$ and $e^{(0)} = (1, 0)^\top$, the second raw error is zero, while $\beta = 3/5$ gives Anderson error $(0, -2/5)^\top$.

On the 28 checkpoints with the real model, the prediction $e^{(1)} \approx -Be^{(0)}$ has median relative error 0.218, the first-round error points against the initial error in all 28, and the median largest node error is 0.522, 0.485, 0.174, and 0.038 after zero to three raw rounds and 0.064 after two rounds with Anderson (β median 0.641).

C.6 Clean-form rounds

Eq. (10) gives a second prefix map that uses the same network calls as G_r. From a guess Z with $Z_0 = \epsilon_r$, one batched call gives $D_{r,j}(Z_j) = Z_j - \sigma_j v_{r,j}(Z_j; o_r)$ for $j < N$, and we set

$$\begin{aligned} [\tilde{G}_r(Z)]_k &= \sigma_k \epsilon_r + \sum_{j<k} w_{k,j} D_{r,j}(Z_j) \quad (0 \le k < N), \\ [\tilde{G}_r(Z)]_N &= D_{r,N-1}(Z_{N-1}). \end{aligned} \tag{17}$$

By Eq. (10), $X_r^\star$ is a fixed point, and forward substitution shows that it is the only one. Where G_r adds the velocities, $\tilde{G}_r$ treats the linear part x/σ of $v = (x - D)/\sigma$ exactly, as exponential integrators do.

Proposition 3 (Clean-form rounds)**.** *Under the assumption of Theorem 1, let $\tilde{M}_{kj} = W_{kj}\ell_j$ for $1 \le j < k \le N$ and zero otherwise, with W as above, and let $\bar{\ell} = \max_{1 \le j < N} \ell_j$. After R rounds of $\tilde{G}_r$, elementwise $E^{(R)} \le \tilde{M}^R E^{(0)}$, so*

$$\max_k E_k^{(R)} \le \bar{\ell}^{\,R} \, (e_{r-1} + \eta_r), \tag{18}$$

and nodes $k \le R$ are exact. If the cache holds the raw clean-form output, $\bar{\ell} < 1$ for every replan, and $\eta_r \le \bar{\eta}$, then $\limsup_r e_r \le \bar{\ell}^{\,R} \bar{\eta} / (1 - \bar{\ell}^{\,R})$.

Proof. Subtracting Eq. (10) from Eq. (17), the noise terms cancel and $[\tilde{G}_r(Z) - X_r^\star]_k = \sum_{j<k} W_{kj} \big(D_{r,j}(Z_j) - D_{r,j}(X_{r,j}^\star)\big)$, where the $j = 0$ term vanishes because $e_0 = 0$. Taking norms gives $E' \le \tilde{M} E$. The rows of W sum to $\alpha_k \le 1$ for $k < N$ and to one for $k = N$, so $\|\tilde{M}\|_\infty \le \bar{\ell}$ and $\|\tilde{M}^R\|_\infty \le \bar{\ell}^{\,R}$. Exact nodes follow as in Theorem 1, since $\tilde{M}$ is strictly lower triangular. The initializer bound is that of Theorem 1, and the recursion $e_r \le \bar{\ell}^{\,R}(e_{r-1} + \eta_r)$ gives the limit. □

Compared with $M_{kj} = b_j(1 + \ell_j)$, the guess error no longer re-enters each round directly; it passes only through the sensitivity of the clean predictions. The factor $\|M\|_\infty \ge 1$ that forces the Anderson step disappears, and for input-independent clean predictions one round is exact, whereas a velocity-form round returns $-Be^{(0)}$. For the Bayes-optimal clean prediction $D_j(x) = \mathbb{E}[A \mid x_{\sigma_j} = x, o]$, Tweedie's formula gives $\partial D_j = (1 - \sigma_j)\sigma_j^{-2} \operatorname{Cov}(A \mid x_{\sigma_j} = x, o)$. Hence ℓ_j is small when the policy's conditional action distribution is concentrated on the scale σ_j. The smallest nonzero noise level on our schedule is $\sigma_{N-1} = 0.358$, so the regime $\sigma \to 0$, where $\ell_j \to 1$, is not on the grid. The Lipschitz constants are local, however, and far from the solution $\bar{\ell}$ can exceed one.

On the 28 checkpoints of Appendix C.5, evaluated offline from the stored rounds without extra calls, one clean-form round from the transported initializer gives a median largest node error of 0.023. It is worse than one velocity-form round in 6 of the 28, all with first-round velocity error above 0.8. A velocity-form round followed by a clean-form round, with two calls as in BATON, gives 0.020 and is below the deployed output (0.064) on all 28 checkpoints; the reference trajectories are stored in BF16. In closed loop on the 450 instances of the budget sweep, under its protocol, the velocity-then-clean and clean-then-clean variants solve 304 and 298, against 310 for BATON $N = 10$ (-1.33 pp, $p = 0.392$; -2.67 pp, $p = 0.096$). On the 322 instances with oracle logs, their median final-action error against the nonexecuted serial solution is 0.010 and 0.007, against 0.061 for BATON, but their 99th percentiles are larger, 0.865 and 1.428 against 0.694. A lower typical error therefore does not raise success here; the heavier tail, where $\bar{\ell}$ exceeds one, may offset it. We keep the Anderson step as the deployed method.

C.7 Deadlines and the deployment clock

Let f_a be the checkpoint's native action-token rate. A periodic look-ahead executor starts inference while a fresh H-token plan is active and targets installation after S ticks. Token zero of a completed chunk is the next executable command, and installation refreshes all H slots. The nominal update deadline is $D_S = S/f_a$, the hard buffer deadline is $D_H = H/f_a$, and $(H-S)/f_a$ is only late-arrival slack. A late result first continues the old plan and, after exhaustion, triggers a predefined hold; every result is eventually installed, and dropping, superseding, or queueing results is excluded.

Lock-step evaluation. In our simulator, the transition $x_{r+1} = \Phi_{S/f_a}(x_r, a_r)$ occurs only after the policy returns, so success does not depend on latency L_r.

Nominal deadlines. Let $L_R(H; f_a)$ be the latency through chunk production, $g \geq 0$ fixed non-model delay or a conservative guard, and $\beta \in [0, 1)$ a miss allowance. The smallest nominal stride meeting the D_S quantile deadline is

$$S_\beta(R, H, f_a) = \max\left\{1, \left\lceil f_a\left[Q_{1-\beta}(L_R(H; f_a)) + g\right]\right\rceil\right\}, \quad f_{r,\beta} = \frac{f_a}{S_\beta}, \quad \tau_H = \frac{H}{f_a}. \tag{19}$$

This nominal stride is feasible only if $S_\beta \leq H$. Inside one integer cell, faster inference adds slack but does not change the nominal rate; it raises the rate only when a smaller integer stride becomes feasible. For the 20 Hz checkpoint, $S = 10$ gives 2 Hz and $S = 1$ is the 20 Hz ceiling.

Achieved rate under misses. For $1 \leq S \leq H$, let $j_r = \lceil f_a(L_r + g)\rceil$ and $k_r = \max\{S, j_r\}$. Continuation and hold ticks are

$$c_r = [\min\{k_r, H\} - S]_+, \qquad u_r = [k_r - H]_+, \qquad k_r = S + c_r + u_r. \tag{20}$$

With $\tau_r = k_r/f_a$, for n cycles,

$$T_n = \sum_{r=1}^{n} \tau_r = \frac{nS + \sum_{r=1}^{n}(c_r + u_r)}{f_a}, \qquad f_{r,n}^{\text{real}} = \frac{n}{T_n} \longrightarrow \frac{f_a}{S + \mathbb{E}[(j-S)_+]}, \tag{21}$$

where the limit requires stationary ergodic j_r with finite mean. T_n is the time for n installed updates, not task-completion time. Late results therefore lower the achieved rate even when the median latency meets the deadline.

D Experimental Details

D.1 Protocol, latency, and statistics

Campaigns. The headline evaluation, the solver ablations, and the budget sweep are distinct campaigns with their own serial references. They share instance keys; follow-up arms ran on later dates. Each nonserial arm starts from a serial bootstrap and then executes and caches its own approximate trajectory. The simulator pauses during inference, so success is measured at a fixed stride.

Latency scopes.

- **Action Expert CUDA**: CUDA-event span of the Action/DiT solver graph per generated chunk. It excludes context, vision, CPU transfer, and the simulator.
- **Steady backend wall**: backend wall span including context inputs and cache, vision, noise, the action expert, action transfer to host, and history cloning. It excludes observation preprocessing, prompt pre-encoding, and the simulator.

For the headline Fast-WAM comparison, both scopes and both arms discard the first three replans of each rollout to remove bootstrap, first-use graph, and lazy effects, leaving 11,170 serial and 11,048 BATON calls from 450 rollouts per arm. Hardware, scopes, and builds are never pooled.

Statistics. Paired success differences use percentile-bootstrap intervals over instance keys (20,000 resamples) and exact two-sided McNemar tests. Mean steps to success use each arm's own successful episodes; paired step comparisons use joint successes.

Comparison scope. The cold arm is a fixed-budget Picard control, not a reproduction of adaptive ParaDiGMS or ParaTAA. Warm serial is our re-noising control. RTI-DP-clip and FALCON are adaptations to the Fast-WAM flow sampler (Appendix D.5).

D.2 FAST-WAM ON LIBERO-PLUS

The 450 pairs give 304 serial and 311 BATON successes, with 24 BATON-only, 17 serial-only, and 409 tied outcomes. Table 5 and Figure 5 report latency percentiles.

Table 5: **Headline latency at the median and upper percentiles.** Times are milliseconds per generated chunk, measured with the same runtime after excluding the first three replans of each rollout. Speedup is the serial-to-BATON ratio at each percentile.

Scope	Arm	p50	p95	p99
Action Expert CUDA	Serial	29.919	30.989	32.472
	BATON	10.669	11.775	13.512
	Speedup	2.804×	2.632×	2.403×
Steady backend wall	Serial	43.248	46.265	50.922
	BATON	25.003	28.205	33.488
	Speedup	1.730×	1.640×	1.521×

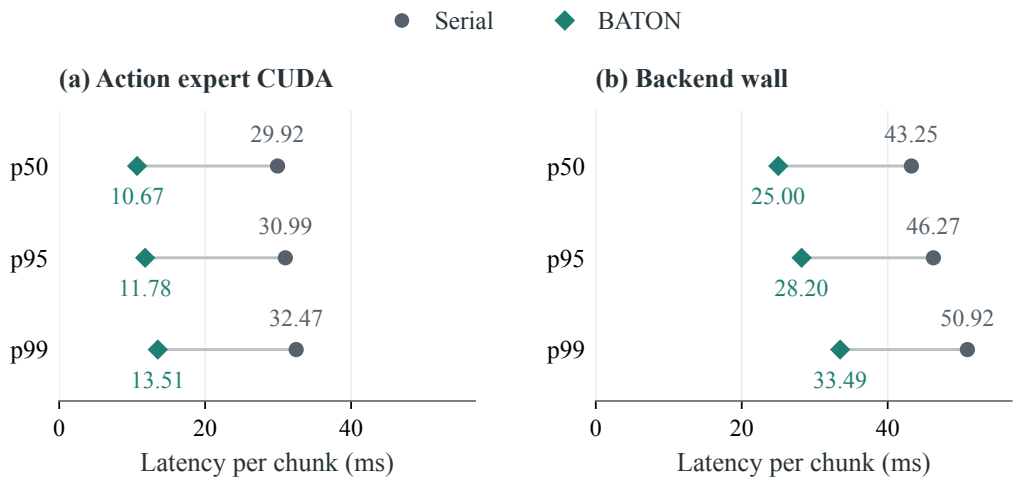

Figure 5: **The latency distribution shifts downward with BATON.** Circles and diamonds compare serial and BATON at p50, p95, and p99; connectors pair the same percentile. Values are milliseconds per chunk after excluding the first three replans, using the timing scopes in Table 1.

A separate raw-core build gives 2.777/2.430/2.101× Action Expert and 1.845/1.640/1.498× steady-wall speedups at p50/p95/p99. In the optimized serial stack profiled for Section 1, same-run backend CUDA p50 is 40.575 ms, of which the ten-step Action/DiT takes 29.356 ms.

D.3 SOLVER ABLATIONS

All eight rows of Table 6 use the same 450 instance keys (350 single-factor and 100 official-mix) and share the checkpoint, VAE, dataset statistics, exact-prefill setting, and evaluator. The original raw

campaign gives 6, 306, 264, and 306 successes for cold, unaligned, transport, and serial; every raw arm uses two Picard rounds without Anderson (Figure 6). The follow-up transport-plus-Anderson arm is a new run on host A, one of the two H100 hosts of Appendix D.5; its 311 successes coincide with, but are separate from, the headline campaign.

Table 6: **Complete initializer and solver ablations.** All eight configurations use the same 450 instances with $N = 10$, $H = 32$, and $S = 10$. AA denotes depth-one Anderson; terminal B2 corrects the last two denoising states after one parallel round. Mean steps count control actions over each arm's own successful episodes, including the serial bootstrap. Every arm caches its own output.

Initializer	Updates	Successes	Mean steps (success)
Serial	10 sequential	306	161.9
Cold / noise-like	2 raw rounds	6	127.5
Unaligned reuse	2 raw rounds	306	186.0
Unaligned reuse	2 rounds + AA	294	165.5
Transport	2 raw rounds	264	159.2
Transport	1 round + terminal B2	291	163.0
Transport	3 raw rounds	309	163.5
Transport	2 rounds + AA	311	163.2

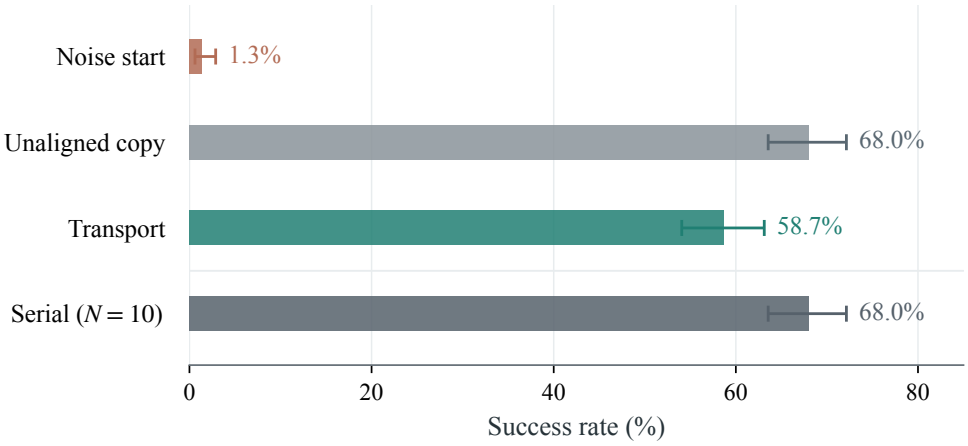

Figure 6: **History initialization increases success within two raw rounds.** Bars report success rates over 450 matched instances; whiskers show marginal 95% Wilson intervals. Noise start, unaligned copy, and transport use two raw Picard rounds without Anderson; the serial reference uses ten sequential steps.

All raw initializers use the same two calls and shapes. In a separate 28-case schedule audit on a later backend build, transport versus serial gives Action Expert CUDA p50 of 10.809 versus 30.017 ms and steady-wall p50 of 21.078 versus 38.887 ms.

The copy-plus-Anderson arm solves 294 of 450; 20 instances that failed at runtime on host B were rerun on host A, and valid task failures were never rerun.

Table 7: Paired success differences in the solver ablations, first arm minus second, in percentage points. Intervals are pointwise paired bootstrap intervals over 450 instance keys; exact McNemar tests are two-sided and not multiplicity-adjusted.

Comparison	Δ pp	95% CI	p
Transport 2R+AA − unaligned 2R+AA	$+3.78$	$[0.44, 7.11]$	0.0363
Transport 2R+AA − transport 2R	$+10.44$	$[6.44, 14.44]$	5.99×10^{-7}
Transport 2R+AA − transport 3R	$+0.44$	$[-2.22, 3.11]$	0.8714
Transport 2R+AA − transport 1R+B2	$+4.44$	$[0.89, 8.00]$	0.0225
Transport 2R − unaligned 2R	-9.33	$[-13.78, -4.89]$	7.69×10^{-5}

Transport versus copy, both with Anderson, has 38 transport-only and 21 copy-only successes; their 273 joint successes take 43,406 and 43,336 control steps. Anderson versus raw two rounds has 68 Anderson-only and 21 raw-only successes. The three-round comparison has 20/18 discordance. The 1R+B2 arm runs one ten-state batch and then a two-state terminal correction, so it is not a pure one-call sampler.

Table 6 reports mean control steps over each arm's own successes, counting environment actions including the serial bootstrap. Serial, unaligned reuse, and transport with Anderson average 161.9, 186.0, and 163.2 steps; copy with Anderson averages 165.5; the cold mean of 127.5 covers only six successes. On joint successes, raw unaligned reuse needs 15.81% more steps than serial (263 instances), whereas raw transport needs 3.23% more (241 instances). Against each other, raw transport and raw unaligned take 34,547 and 39,853 steps over 230 joint successes (-13.31%).

D.4 Common-input initializer diagnostic

Collection. We select task indices $\{0, 1, 3, 4, 6, 8, 9\}$ from each of LIBERO Spatial, Object, Goal, and Long before collection, with initial-state index zero and seed 42. Rollouts use 30 settling steps, $H = 32$, $S = 10$, and the standard limits (400 action steps, 700 for Long), stopping on success. A ten-step serial policy supplies the executed actions and the cached solver history, so the study measures refinement at common states rather than the accumulation of approximate history. All 408 checkpoints after the bootstrap replan are retained. At each, the cold, unaligned, and transport arms share the observation, proprioception, language context, noise, schedule, previous serial trajectory, and serial reference, and are computed without execution. We record rounds zero to three without Anderson (4,896 records).

Metrics. Relative path RMSE is the RMSE over all nodes, horizon slots, and action dimensions divided by the reference RMS. Final-action error uses the last node and the six continuous normalized coordinates; the prefix metric uses the first ten horizon slots. Gripper disagreement is recorded separately after binarization. Plotted errors are distances to the serial reference, not update residuals.

Figure 3. Panel (a) uses the first valid-history checkpoint of each task. At each position, RMS error over the six continuous coordinates is divided by that snapshot's full-horizon reference RMS; curves and bands are medians and interquartile ranges after two raw rounds. With $H = 32$ and shift 10, positions 1 to 22 overlap the previous chunk, positions 23 to 32 repeat its final action, and positions 1 to 10 are executed. Panel (b) takes each task's median prefix error, then the median over 28 equally weighted tasks; bands are percentile intervals from 5,000 paired bootstrap draws (seed 20260922) that resample seven tasks within each suite. Panel (c) plots all 408 paired two-round prefix errors: 399 favor transport and nine favor unaligned reuse, with pooled median paired ratio 0.1764. Full-path and full-horizon continuous-action errors improve in all 28 task medians, with median taskwise transport/copy ratios 0.322 and 0.368.

D.5 Budget sweep

Cohort. Figure 2 uses all 450 instance keys in each of seventeen arms. All arms share the checkpoint, $H = 32$, $S = 10$, and instance seeds; the sweep varies serial depth, warm-serial suffix length, or BATON's grid. Instances run on two H100 hosts, 322 on host A and 128 on host B; instances that crashed were rerun on host A for every affected arm. The serial and BATON $N = 10$

counts, 301 and 310, fall within the preregistered consistency windows 306 ± 10 and 311 ± 10. The $N = 5$ points reuse an earlier campaign. Warm serial re-noises a shifted clean chunk.

RTI-DP-clip. Following the official implementation, the three arms shift the clean history, repeat the last action, and treat discrete actions with its sign and factor rule (factor 10, fixed in advance), without re-noising. We apply the last $K \in \{1, 2, 3\}$ steps of Fast-WAM's flow Euler schedule and execute ten actions per replan without retraining (Duan et al., 2025). The original method uses DDPM with $S = 1$.

FALCON. The adaptation uses a capacity-20 buffer, exploration probability 0.1, temperature 1, minimum noise $\sigma = 0.1$, and thresholds $\epsilon \in \{0.005, 0.01, 0.02\}$. It screens unshifted historical noisy states under the current observation with the flow endpoint $x_\sigma - \sigma v$ and the official $\ell_2/154$ score, then runs a serial suffix or falls back to the full serial sampler. We cache pre-update latents and refresh the clean reference every replan, following the paper's state convention rather than the released DDPM code line by line. All screening, selection, buffer, suffix, and fallback work is timed.

Serial $N = 20$. This arm runs the native 20-step Euler schedule without history reuse.

Table 8: Budget sweep. Success uses the same 450 instances. CUDA latency uses the original 322 host-A instances and excludes the first replan. Mean steps count executed control actions over each method's own successes. RTI and FALCON are native-flow adaptations.

Method	Successes	Success (%)	CUDA (ms)	Mean steps
Serial $N = 2$	292	64.89	7.586	164.0
Serial $N = 3$	292	64.89	10.335	161.5
Serial $N = 5$	304	67.56	16.054	161.5
Serial $N = 10$	301	66.89	30.070	160.9
Serial $N = 20$	312	69.33	58.083	163.0
Warm serial $K = 1$	282	62.67	3.217	163.2
Warm serial $K = 2$	298	66.22	6.013	163.4
Warm serial $K = 3$	295	65.56	8.804	161.7
RTI-DP-clip $K = 1$	293	65.11	5.380	161.4
RTI-DP-clip $K = 2$	293	65.11	8.260	165.6
RTI-DP-clip $K = 3$	292	64.89	11.464	158.6
FALCON $\epsilon = 0.005$	309	68.67	37.459	164.9
FALCON $\epsilon = 0.01$	307	68.22	14.533	163.0
FALCON $\epsilon = 0.02$	300	66.67	13.688	163.3
BATON $N = 3$	301	66.89	6.659	162.5
BATON $N = 5$	301	66.89	8.647	160.4
BATON $N = 10$	310	68.89	9.210	165.4

Latency. Success pools the two hosts, but latency uses only the original 322 host-A keys, because the hosts run different drivers (560.35.05 and 595.71.05). Each point is the median of per-episode CUDA medians. The warm and BATON arms ran a nonexecuted serial oracle outside the timed region; the earlier $N = 5$ campaign, RTI, and FALCON ran without it. Collection dates still differ across arms, so small latency differences are not conclusive.

Table 9: BATON $N = 10$ success minus each comparator on 450 paired instances. Pointwise 95% intervals use 20,000 paired bootstrap resamples stratified by the original reference-arm host and cohort. Exact McNemar tests are two-sided; Holm correction covers all sixteen contrasts.

Comparator	Δ pp	95% CI	p	Holm p
Serial $N = 2$	+4.00	[0.67, 7.11]	0.0222	0.2768
Serial $N = 3$	+4.00	[0.89, 7.11]	0.0222	0.2768
Serial $N = 5$	+1.33	[−1.78, 4.22]	0.4709	1.0000
Serial $N = 10$	+2.00	[−0.67, 4.67]	0.1877	1.0000
Serial $N = 20$	−0.44	[−3.33, 2.44]	0.8830	1.0000
Warm serial $K = 1$	+6.22	[3.11, 9.33]	0.0002	0.0028
Warm serial $K = 2$	+2.67	[−0.44, 5.78]	0.1337	1.0000
Warm serial $K = 3$	+3.33	[0.44, 6.22]	0.0315	0.3154
RTI-DP-clip $K = 1$	+3.78	[0.67, 6.67]	0.0213	0.2768
RTI-DP-clip $K = 2$	+3.78	[0.89, 6.67]	0.0137	0.1921
RTI-DP-clip $K = 3$	+4.00	[1.33, 6.89]	0.0079	0.1187
FALCON $\epsilon = 0.005$	+0.22	[−2.67, 3.11]	1.0000	1.0000
FALCON $\epsilon = 0.01$	+0.67	[−2.00, 3.33]	0.7493	1.0000
FALCON $\epsilon = 0.02$	+2.22	[−1.11, 5.56]	0.2370	1.0000
BATON $N = 3$	+2.00	[−1.33, 5.33]	0.3057	1.0000
BATON $N = 5$	+2.00	[−0.89, 4.89]	0.2327	1.0000

Contrasts. Only the contrast with warm $K = 1$ remains below 0.05 after Holm correction. Warm $K = 3$ is within 4.41% of BATON's measured latency, inside the preregistered 10% matching tolerance; its contrast is significant before but not after Holm correction.

Execution length. On joint successes with BATON $N = 10$, warm $K = 2$, serial $N = 20$, RTI, and FALCON all differ by less than 2% in total control steps, so none of them trades latency for longer executions.

D.6 Latency scaling with sampling depth

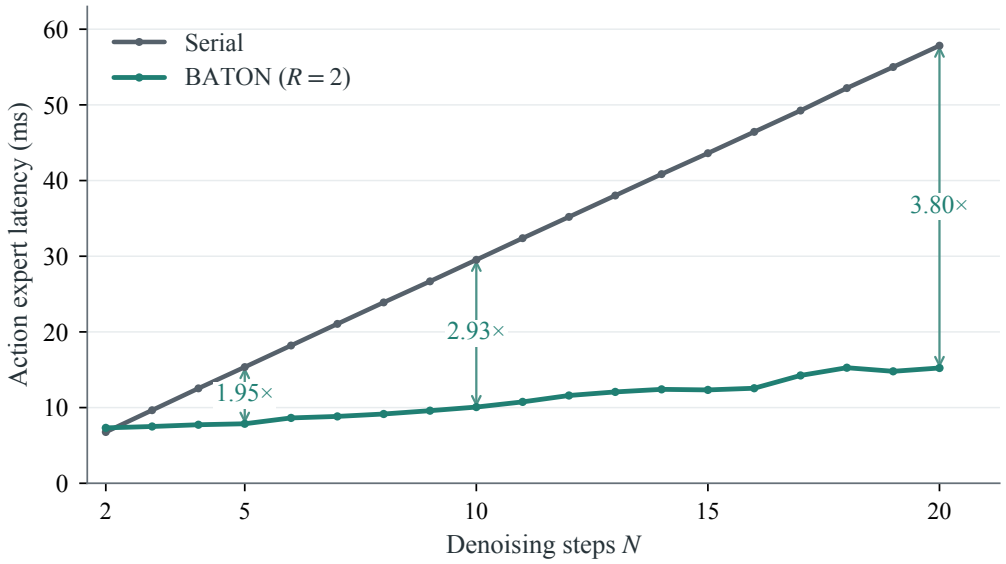

Figure 7: **Two parallel rounds scale more gently with sampling depth.** Serial and BATON action-expert CUDA latency on one H100, measured at every integer $N = 2, \ldots, 20$. Points are medians of 100 measurements after ten warmups; double-headed arrows connect the two curves at $N = 5, 10, 20$, with the serial/BATON ratio at each midpoint. BATON includes transport and global Anderson.

Both samplers use the same GPU, checkpoint, BF16 precision, $H = 32$, and compiled runtime, with transport and Anderson inside the timed region. Inputs are fixed real observations with cached text, so this is a microbenchmark separate from the rollout latencies. At $N = 5, 10, 20$, serial/BATON latency is $15.354/7.862$, $29.527/10.069$, and $57.824/15.233$ ms, giving $1.95\times$, $2.93\times$, and $3.80\times$. At $N = 2$, BATON is slower ($0.925\times$) because the two-round overhead outweighs the short serial chain.

D.7 PORTABILITY

Table 10: **Paired capability across policies and benchmarks.** Each row uses its own paired evaluation set. W/L counts BATON-only/serial-only successes; differences use unrounded success rates. [1] Stable-fixture set, $H = 32$, $S = 10$. [2] All 50 tasks with three seeds, $H = 32$, and $S = 10$.

Policy / benchmark	n	Serial (%)	BATON (%)	Δ pp	W/L
Fast-WAM LIBERO-Plus	450	67.56	69.11	+1.56	24/17
$\pi_{0.5}$ LIBERO-Plus	450	94.00	94.89	+0.89	13/9
Fast-WAM RoboTwin 2.0 [1]	150	83.33	83.33	0.00	5/5
$\pi_{0.5}$ RoboTwin 2.0 [2]	150	44.67	44.67	0.00	13/13

$\pi_{0.5}$ on LIBERO-Plus. The 450-pair campaign uses $N = 10$, $H = 10$, and $S = 5$ with Inductor max-autotune compilation. Serial and BATON solve 423 and 427, with 414 both-success, 13 BATON-only, 9 serial-only, and 14 neither ($p = 0.523$). Timing covers one full policy call, from input to output transforms, excluding the simulator. Over steady replans, the median falls from 25.066 to 19.320 ms ($1.297\times$) and the p95 from 40.960 to 33.849 ms ($1.210\times$).

$\pi_{0.5}$ on RoboTwin 2.0. The campaign covers all 50 tasks with evaluation seeds 42, 43, and 44, giving 150 paired cases, and runs without compilation. It uses $H = 32$, $S = 10$, $N = 10$, two parallel rounds, and correction weight $\lambda = 0.75$. Both arms solve 67 cases (44.67%), with 13 pairs favoring each ($p = 1.0$). Over valid-history replans, pooled full-policy CUDA p50 falls from 157.696 to 62.098 ms ($2.539\times$); this is not the episode-median estimand of Figure 2. Compilation shortens each narrow serial step much more than a batched round, so this uncompiled ratio is not comparable with the compiled LIBERO-Plus one. The protocol uses three evaluation streams per task rather than the official 100 episodes.

Fast-WAM on RoboTwin 2.0. The stable-fixture campaign covers 50 tasks and evaluation streams 42, 43, and 44, for 150 paired cases, with $H = 32$ and $S = 10$. Serial and BATON each solve 125, with 120 both, 5 BATON-only, 5 serial-only, and 20 neither ($p = 1.0$). Seven of the 150 cases are preregistered repairs of unstable simulator setups. Latency uses the scopes and steady rule of Table 1 (3,398 serial and 3,578 BATON calls). Median Action Expert CUDA falls from 30.581 to 11.166 ms ($2.739\times$; $2.581\times$ at p95 and $2.475\times$ at p99), and steady backend wall falls from 45.119 to 26.796 ms ($1.684\times$; $1.594\times$ and $1.530\times$).

RoboDojo. We exclude RoboDojo because repeated evaluation of an unchanged baseline is too noisy: across seven identical repeats on `stack_bowls`, 110 of 420 pairwise layout outcomes flip (26.2%; the pairs share repeats, so we give no interval), comparable with the observed serial/BATON differences (18/18, 16/13, and 16/12 over three seeds).

D.8 EXPLORATORY DEPTH AND STRIDE

These probes use non-frozen protocols.

Table 11: **Capability at two exploratory execution strides.** Each row reports 450 paired episodes. Strides $S = 5$ and $S = 20$ correspond to nominal replanning rates of 4 Hz and 1 Hz at the native 20 Hz action rate.

S	Serial	Candidate	Difference	Candidate/serial-only; p
5	314	319	+1.11 pp	17/12; 0.4583
20	244	250	+1.33 pp	32/26; 0.5118

From formal $S = 10$ to exploratory $S = 20$, serial and BATON success fall by 13.33 and 13.56 pp ($p = 5.5\times10^{-9}$ and 4.4×10^{-10}). The degradation is similar for both samplers. Action Expert CUDA p50 is $29.928 \rightarrow 10.755$ ms ($2.783\times$) at $S = 5$ and $29.134 \rightarrow 11.000$ ms ($2.649\times$) at $S = 20$.

E LIMITATIONS

The bounds in Section 3 hold under Lipschitz clean predictions, but their constants are not measured for the deployed networks, and the plan-change bound rests on the consistency assumption (ii) of Appendix C.3. The explanation of the Anderson step linearizes a concentrated policy and gives no error bound for the deployed output.

BATON reduces dependency latency, not logical work. It uses more evaluations and a larger live working set, and it can lose when temporal batches do not improve accelerator utilization or when perception dominates the policy call. Stochastic samplers require consistent transport of every forcing-noise increment; nonlinear coordinates, saturation, and hybrid actions require model-specific transports. The fixed-round approximation need not preserve the serial output distribution.

The evidence is simulator-only and hardware-specific. Fixed-stride results do not establish a realized frequency gain or real-robot safety.