OVERCOMING DYNAMICS-BLINDNESS: TRAINING-FREE PACE-AND-PATH CORRECTION FOR VLA MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-Language-Action (VLA) models achieve remarkable flexibility and generalization beyond classical control paradigms. However, most prevailing VLAs are trained under a single-frame observation paradigm, which leaves them structurally blind to temporal dynamics. Consequently, these models degrade severely in non-stationary scenarios, even when trained or fine-tuned on dynamic datasets. Existing approaches either require expensive retraining, or suffer from latency bottlenecks and poor temporal consistency across action chunks. We propose **Pace-and-Path Correction (PPC)**, a training-free, closed-form inference-time operator that wraps any chunked-action VLA. Under a local motion model, joint minimization of a single quadratic cost yields an interior solution with two orthogonal channels. The *pace channel* compresses execution along the planned direction, while the *path channel* applies an orthogonal spatial offset, jointly absorbing the perceived dynamics within the chunk window. We evaluate our approach on a comprehensive diagnostic benchmark MOVEBENCH designed to isolate motion as the sole controlled variable. Under the controlled MOVEBENCH protocol, PPC outperforms the evaluated training-free wrappers and dynamic-adaptive baselines, raising success by up to 28.8% and 25.9% in absolute terms over foundational VLAs in dynamic-only and static-dynamic mixed environments, respectively. We further validate PPC on a real robot with a backbone fine-tuned only on static demonstrations, increasing dynamic grasping success from 0% to 68.3%.

1 INTRODUCTION

Robotic manipulation in real-world settings frequently involves environments whose state changes during policy execution, ranging from regular motions such as objects on a conveyor belt to unexpected events such as external perturbations (Zhang et al., 2025d; Fang et al., 2026; Xie et al., 2026). Handling such dynamic conditions has therefore become a central requirement for general-purpose manipulation policies (Hu et al., 2023; Ma et al., 2024). Among recent approaches, Vision-Language-Action (VLA) models map visual observations and language instructions directly to low-level control, and have emerged as a promising candidate for this setting (Team et al., 2024; Zitkovich et al., 2023; Kim et al., 2024). However, most current VLAs adopt action chunking, where the model predicts a fixed-length sequence of future actions from a single visual frame at each inference call, and the robot executes them open-loop before the next chunk is generated (Black et al., 2024; Zhao et al., 2023). While this design improves stability and amortizes inference cost, it leaves the policy structurally blind to dynamics (Fang et al., 2026; Chi et al., 2025). Each chunk is generated from an initial static snapshot without object-motion supervision and executed without visual feedback, leaving any scene changes during execution unseen until the next chunk is generated (Zitkovich et al., 2023; NVIDIA et al., 2025). As a result, even state-of-the-art VLAs that excel on static tasks degrade sharply once the task itself demands temporal awareness. Beyond methods, the evaluation landscape itself offers limited support for diagnosing motion robustness. Existing manipulation benchmarks rarely isolate motion as a primary axis, instead entangling it with perception, generalization, or scene difficulty, which makes the dynamics-blindness failure mode hard to characterize precisely (Fang et al., 2026; Xie et al., 2026; Liu et al., 2023; Mees et al., 2022).

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.

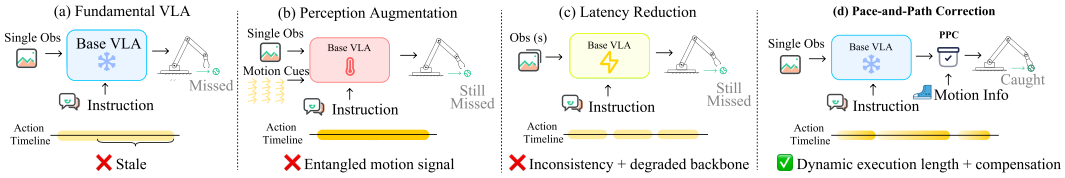


Figure 1: **Comparison of methods.** (a) Single-frame VLAs produce stale actions in dynamic scenes. (b) Perception augmentation requires retraining. (c) Faster inference can disrupt inter-chunk consistency. (d) PPC jointly adapts execution pace and corrects the spatial path.

Recent work targeting this gap broadly falls into two strands. One injects motion or temporal cues into the input through historical optical flow (Zhong et al., 2026; Fang et al., 2026; 2025), visual prompting (Zheng et al., 2025), memory banks (Shi et al., 2026), or motion predictors (Zhang et al., 2025c; Cen et al., 2025; Zhang et al., 2026), but these methods rely on expensive retraining and per-backbone architectural changes. Extraction latency and forecasting hallucinations make these methods unreliable at the timescale of dynamic interaction (Zhong et al., 2026; Sendai et al., 2025; Jiang et al., 2026). More fundamentally, a manipulator's visual stream is dominated by its own ego-motion, leaving genuine object motion as a small residual hard to capture (Zhi et al., 2025; Kambara et al., 2026). A second strand reduces inference latency through compact backbones (Xie et al., 2026), parallel decoding (Song et al., 2025), or compressed action tokenizers (Liu et al., 2025b), trading away the backbone capacity that gives larger VLAs their generalization while still leaving each newly issued chunk blind to motion within the previous one. Indiscriminate re-inference can also break the temporal smoothness across chunks and degrade long-horizon coherence (Black et al., 2025a). Other methods include asynchronous inpainting (Black et al., 2025a), rejection sampling (Liu et al., 2025c), temporal ensembling (Zhao et al., 2023), adaptive chunk sizing (Wen et al., 2026), and learned correction heads (Sendai et al., 2025), which improve reactivity indirectly through smoother seams or more frequent re-planning. However, the chunks themselves still treat the environment as static, and any learnable corrector still faces the latency-capacity dilemma and ego-motion problem (Zhao et al., 2023; Huang et al., 2026). Without external dynamics information, identical initial observations with different target velocities make intra-chunk correction underdetermined.

Therefore, we propose **Pace-and-Path Correction (PPC)**, a closed-form, training-free, inference-time wrapper. PPC reads an external dynamics signal in the form of velocity, which can be supplied by external tracking or depth-sensing pipelines. As illustrated in Fig. 1, unlike prior remedies that augment the input, shrink the backbone, or smooth chunk boundaries, PPC directly addresses the chunk interior where dynamics blindness actually resides, through a principled, physics-grounded formulation. It solves a single quadratic cost balancing per-waypoint tracking against per-step offset effort in closed form, whose interior solution decomposes orthogonally into two channels. **Pace** adaptively compresses the chunk in time to absorb the plan-parallel component of the disturbance, while **Path** adds per-step spatial offsets to absorb the plan-perpendicular component. A **Hierarchical 2-EMA Latch Stabilizer** further detects motion regimes and shortens the execution horizon for necessity under chronic instability. By decoupling perception from correction, PPC inherits the maturity of dedicated tracking pipelines, sidesteps the latency-capacity dilemma that constrains any learnable corrector, and avoids the ego-motion confound that handicaps in-backbone perception. The resulting wrapper is agnostic to the underlying backbone, requires negligible compute, and recovers the baseline VLA under zero motion input, preserving its behavior through the static bypass. To rigorously study PPC and the broader question of motion robustness, we further construct **MoveBench**, a controlled benchmark that isolates motion regime as the primary evaluation axis while holding tasks, objects, and scenes fixed. Our key contributions are:

- We propose **Pace-and-Path Correction (PPC)**, a closed-form, training-free, inference-time wrapper for general VLAs that explicitly compensates for environment dynamics with no learnable parameters and no backbone modification or specification.
- We construct **MoveBench**, a benchmark dedicated to systematically isolating and evaluating VLA performance across diverse motion patterns and speeds.
- Under the controlled MoveBench protocol, PPC outperforms the evaluated wrappers and dynamic-adaptive baselines, gaining up to 28.8 and 25.9 percentage points in dynamic-only and mixed environments. With a backbone fine-tuned only on static data, it raises real-robot dynamic grasping success from 0% to 68.3%.

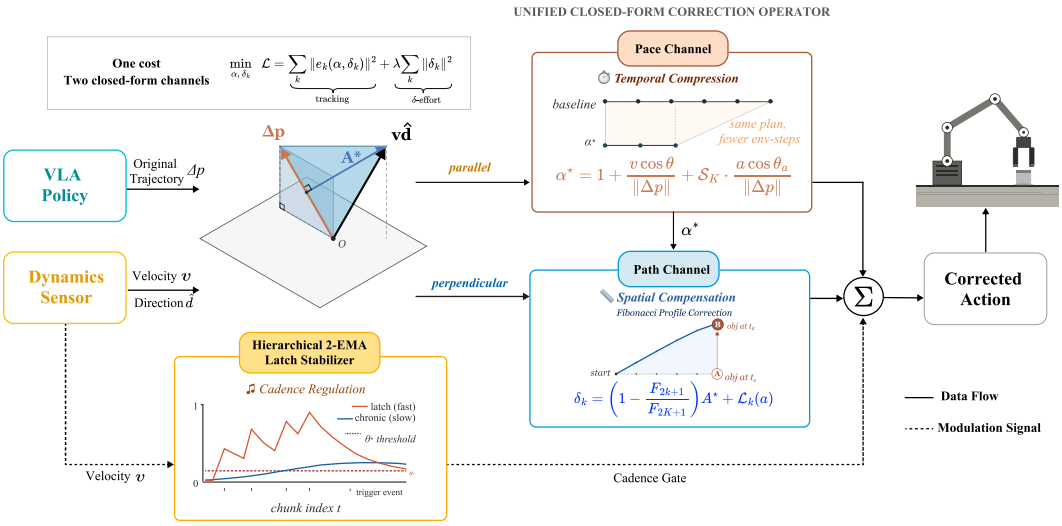


Figure 2: **Framework overview.** PPC wraps a frozen VLA with two closed-form channels. *Pace* adapts execution timing, while *Path* corrects spatial drift. A 2-EMA latch stabilizer regulates execution length under irregular motion. Corrections are training-free with no learnable parameters.

2 Related Work

Vision-Language-Action Models. VLAs adapt pretrained vision-language backbones for robot control by mapping multimodal observations and language to action sequences (Zitkovich et al., 2023; Kim et al., 2024; Black et al., 2024). Early designs decode actions autoregressively as discrete text tokens, enabling reuse of language-modeling objectives but limited by the resolution of binned actions and the cost of token-by-token decoding (Zitkovich et al., 2023; Kim et al., 2024; Brohan et al., 2022; Collaboration et al., 2025). More recent generalist policies attach a diffusion- or flow-matching action expert that emits continuous action chunks, recovering high-frequency control at the cost of grafting newly initialized weights onto the backbone (Chi et al., 2025; Black et al., 2024; Intelligence et al., 2025; NVIDIA et al., 2025; Shukor et al., 2025; Liu et al., 2025a). Across both lines, action chunking has emerged as the de facto control unit, where each inference call produces a fixed-horizon sequence executed open-loop before the next observation, trading reactivity for inference amortization (Zhao et al., 2023; Liu et al., 2024; Zhang et al., 2025b).

Dynamic Manipulation Benchmarks. Robot manipulation benchmarks have largely standardized around static settings, with widely used VLA evaluation suites such as LIBERO (Liu et al., 2023), CALVIN (Mees et al., 2022), ManiSkill (Mu et al., 2021; Gu et al., 2023), RoboCasa (Nasiriany et al., 2024), and VLABench (Zhang et al., 2025a) measuring long-horizon planning, language grounding, or skill transfer while keeping objects stationary. Dynamic settings have only recently entered the VLA picture, primarily through DOM (Xie et al., 2026) and DOMINO (Fang et al., 2026) as VLA-paired benchmarks targeting moving objects. These efforts establish that dynamic conditions degrade VLA performance, yet they treat motion as one axis intermixed with perception, generalization, or scene difficulty, and the underlying motion regimes are typically limited to uniform translation or simple acceleration (Zhang et al., 2025d; Burgess-Limerick et al., 2022; Hassan et al., 2025). A controlled evaluation that varies motion alone, across uniform, accelerated, and irregular regimes, while holding tasks, objects, and scenes fixed remains an open need.

Dynamics-Aware Vision-Language-Action Models. Existing remedies broadly follow two threads. The first injects temporal or predictive cues into the backbone: FlowVLA (Zhong et al., 2026), PUMA (Fang et al., 2026), and LaMP (Wang et al., 2026) feed historical optical or scene flow, TraceVLA (Zheng et al., 2025) overlays visual traces, MemoryVLA (Shi et al., 2026) retrieves from episodic memory banks, and DreamVLA (Zhang et al., 2025c), WorldVLA (Cen et al., 2025), 4D-VLA (Zhang et al., 2026), FUTURE-VLA (Fan et al., 2026), and SC-VLA (Liu et al., 2026a) forecast future states through world models or predictive heads, all requiring retraining and architecture-specific integration (Fang et al., 2025). The second reduces inference latency while retaining

the single-frame paradigm: DynamicVLA (Xie et al., 2026) shrinks the backbone to 0.4B, PD-VLA (Song et al., 2025) parallelizes autoregressive decoding, FASTer (Liu et al., 2025b) compresses action tokenization, and others accelerate through token caching (Xu et al., 2025; Tan et al., 2025), discrete diffusion (Liang et al., 2025), or asynchronous inference (Zhang et al., 2024). Orthogonal efforts repair chunk boundaries at inference time through temporal ensembling (Zhao et al., 2023), guided rejection sampling (Liu et al., 2025c), asynchronous inpainting (Black et al., 2025a;b), learned correction heads (Sendai et al., 2025), native continuation (Liu et al., 2026b), or adaptive chunk sizing (Wen et al., 2026), smoothing inter-chunk seams without addressing intra-chunk drift.

3 METHODOLOGY

3.1 PROBLEM FORMULATION

A VLA policy maps an observation o_t and a language instruction to an action chunk $A_t = (a_t, \dots, a_{t+H-1})$, where each a_k encodes an end-effector delta $\Delta p_k \in \mathbb{R}^3$ together with rotation and gripper commands. The robot executes the first $K \le H$ entries open-loop before re-querying the policy, with T denoting the full chunk length. Let Δp denote the representative per-step delta under a locally constant plan approximation, so the nominal trajectory is $p_k = k\Delta p$ for $k = 1, \dots, K$. Here v is a scalar target-displacement magnitude per control step (physical speed multiplied by the control timestep), and $v\hat{d}$ is the displacement vector along unit direction $\hat{d}$. When the target moves during execution, the waypoints to track shift to $\tilde{p}_k = k(\Delta p + v\hat{d})$, while the chunk continues toward p_k, yielding a tracking error $\|p_k - \tilde{p}_k\| = kv$ that grows linearly with disturbance magnitude and step index and remains invisible to the policy until the next chunk is queried. To close this gap at inference time, we introduce a temporal-compression scalar $\alpha \ge 1$ and per-step spatial offsets $\{\delta_k\}_{k=0}^{K-1} \in \mathbb{R}^3$ on the chunk interior, so that the corrected delta at env-step k becomes $u_k = \alpha\Delta p + \delta_k$. Introducing the residual disturbance $A := v\hat{d} - (\alpha - 1)\Delta p$ and the cumulative spatial offset $\sigma_k := \sum_{i=0}^{k-1} \delta_i$, the per-waypoint tracking error becomes $e_k(\alpha, \{\delta_i\}) = -kA + \sigma_k, \qquad k = 1, \dots, K$. We then choose $(\alpha, \{\delta_k\})$ by minimizing

$$\min_{\alpha, \{\delta_k\}} \mathcal{L} = \sum_{k=1}^{K} \|e_k(\alpha, \{\delta_i\})\|^2 + \lambda \sum_{k=0}^{K-1} \|\delta_k\|^2,$$

where $\lambda > 0$ balances waypoint tracking against the effort of spatial deviation. For an interior pace solution, this local quadratic admits a closed-form minimizer with orthogonal channels relative to the plan direction. Joint stationarity yields

$$\Delta p \cdot \sum_{k=1}^{K} k\, e_k = 0, \qquad \lambda\, \delta_k + \sum_{j=k+1}^{K} e_j = 0.$$

We next derive the two channels sequentially from these interior stationarity conditions.

3.2 PACE CHANNEL CORRECTION

Rotational invariance of the cost forces every δ_k at the optimum to inherit the direction of A, so $\sum_j j\, e_j$ lies parallel to A and the first stationarity condition collapses to $\Delta p \cdot A = 0$. Expanding this orthogonality yields

$$\alpha^\star = 1 + \frac{v \cos\theta}{\|\Delta p\|}, \qquad \cos\theta = \hat{d} \cdot \widehat{\Delta p}.$$

The cosine factor ensures that only the disturbance component aligned with the plan modulates the pace, and substituting $\alpha^\star$ back into A produces the orthogonal residual $A^\star = v\hat{d} - v\cos\theta\, \widehat{\Delta p} = v\hat{d}_\perp$, which is perpendicular to the plan for the interior optimum. When the lower bound $\alpha \ge 1$ is active, $\alpha = 1$ and the path channel receives the full disturbance (Appendix A.4). Both corrections vanish at zero motion input. At runtime, the compression is realized by setting $K_{\text{exec}} = \max\big(K, \min(\lceil T/\alpha^\star \rceil, T)\big)$. Extending the same cost under $\Delta p \cdot \delta_k = 0$ to an affine disturbance $v(t) = v_0 + at$ with possibly distinct directions $\hat{d}_v, \hat{d}_a$ ($\cos\theta_v = \hat{d}_v \cdot \widehat{\Delta p}$, $\cos\theta_a = \hat{d}_a \cdot \widehat{\Delta p}$) yields

$$\alpha^\star = 1 + \frac{v_0 \cos\theta_v}{\|\Delta p\|} + \frac{3K(K{+}1)}{4(2K{+}1)} \cdot \frac{a \cos\theta_a}{\|\Delta p\|},$$

with the second-order coefficient linear in K, reflecting longer integration of acceleration.

3.3 Path Channel Correction

The Path channel handles the residual $A^\star$, which cannot be absorbed by temporal scaling. Setting $\lambda = 1$, as in our simulation experiments (generalized in Appendix A.7), and differencing the second stationarity condition in k yields the 2D linear recurrence

$$\begin{pmatrix}\delta_{k+1}\\ e_{k+1}\end{pmatrix} = \begin{pmatrix}2 & 1\\ 1 & 1\end{pmatrix}\begin{pmatrix}\delta_k\\ e_k\end{pmatrix} - \begin{pmatrix}A^\star\\ A^\star\end{pmatrix}, \qquad e_0 = 0,\ \delta_K = 0.$$

The companion matrix has eigenvalues $\varphi^{\pm 2}$ where $\varphi = (1+\sqrt{5})/2$ is the golden ratio. Solving the recursion under the boundary conditions and applying the identity $\varphi^n + \varphi^{-n} = \sqrt{5}\,F_n$ for odd n yields

$$\delta_k^\star \;=\; \left(1 - \frac{F_{2k+1}}{F_{2K+1}}\right) v\hat{d}_\perp, \qquad k = 0, \dots, K-1,$$

where F_n is the n-th Fibonacci number. The profile saturates from $\delta_0^\star \approx v\hat{d}_\perp$ at the chunk start to $\delta_{K-1}^\star \to (1-\varphi^{-2})v\hat{d}_\perp \approx 0.618\, v\hat{d}_\perp$ as $K \to \infty$, with $\delta_K = 0$ providing an auxiliary terminal condition for the finite-horizon recurrence. This shape minimizes the joint tracking-and-effort cost, distributing the correction across the executed window. Under the second-order disturbance, the same recurrence acquires an inhomogeneous term proportional to $B^\star := \frac{1}{2}a\hat{d}_{a,\perp}$, and linearity of the recurrence yields an additive decomposition into a Fibonacci first-order branch and a Lucas-polynomial second-order branch $\Lambda_k(K)$,

$$\delta_k^\star \;=\; \left(1 - \frac{F_{2k+1}}{F_{2K+1}}\right) A^\star + \Lambda_k(K)\, B^\star,$$

where the Lucas profile is Fibonacci's natural dual on this eigenvalue structure. With $\alpha^\star$, $u_k = \alpha^\star \Delta p + \delta_k^\star$ follows solely from chunk geometry and dynamics, with no learnable parameters.

The general solution also includes the full-compensation limit $\lambda \to 0$, where $\delta_k^\star \to A^\star$ while the pace solution is retained (Appendix A.7). This gives one closed-form family for different backbone responses. We use the regularized profile with dynamically fine-tuned backbones in simulation and the full-compensation limit with the static-only fine-tuned backbone on the robot (Section 4.5).

3.4 Hierarchical 2-EMA Latch Stabilizer

These closed forms are exact under quasi-stationary disturbance. Irregular regimes such as random walk, stop-and-go, and teleport violate this condition; an instantaneous v reading may briefly mislead $\alpha^\star$ into a long execution contradicted by the next observation. A stateful regime classifier complements the operator, detecting sustained instability rather than single-step transients.

For each chunk reset at index t, the stabilizer reads only the velocity stream and computes a hard-thresholded direction-shift trigger $\tau_t = \mathbf{1}[\rho_{\mathrm{gt}}(t) < 1/2]$ from the cosine similarity $\rho_{\mathrm{gt}}(t) = \max(0,\, v_t \cdot v_{t-1}/(\|v_t\|\|v_{t-1}\|))$, which fires when the disturbance direction shifts beyond the natural midpoint. The stabilizer cascades a slow outer EMA with a fast inner EMA. The outer estimates the chronic trigger rate, $C_t = \beta_{\mathrm{out}}\tau_t + (1-\beta_{\mathrm{out}})C_{t-1}$, and feeds a Kalman-style sticky factor $s_t = C_t/(C_t + R_{\mathrm{TH}})$ that modulates the inner decay,

$$L_t \;=\; \begin{cases}\beta_{\mathrm{in}} + (1-\beta_{\mathrm{in}})\, L_{t-1}, & \tau_t = 1,\\ \big[1 - \beta_{\mathrm{in}}(1-s_t)\big]\, L_{t-1}, & \tau_t = 0.\end{cases}$$

Under chronic instability ($s_t \to 1$) the inner state holds; as $s_t \to 0$, its decay approaches the standard rate β_{in}. The latch fires when L_t exceeds a threshold and caps the executed chunk length under sustained irregularity (cadence gate), $m_t = \mathbf{1}[L_t > L_{\mathrm{th}}]$, $K_{\mathrm{exec}} \le T/4$ when $m_t = 1$. The latch admits a single free hyperparameter, the inner EMA rate β_{in}, while the outer EMA rate $\beta_{\mathrm{out}} = 1 - 2^{-K/T}$ and the threshold $L_{\mathrm{th}} = R_{\mathrm{TH}} = \beta_{\mathrm{in}}(1-\beta_{\mathrm{in}})^2$ are derived from the chunk geometry by matching the outer EMA half-life to one chunk-budget cycle and calibrating L_{th} to two-chunk persistence under standard decay ($s_t = 0$); the sticky factor prolongs persistence when $s_t > 0$.

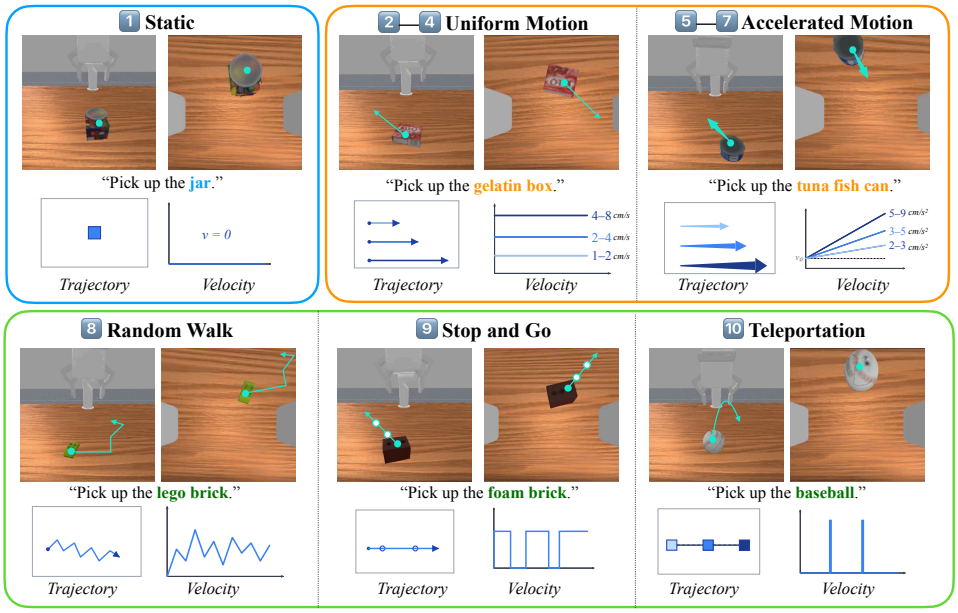


Figure 3: **MOVEBENCH Overview.** MOVEBENCH treats *motion regimes* as the primary evaluation axis, comprising 10,000 trajectories (∼460k frames) across 10 motion settings with everyday household objects randomly sampled across regimes, spanning static, regular, and irregular motion patterns at multiple difficulty levels. Holding non-motion factors fixed isolates motion as the sole variable.

4 EXPERIMENTS

4.1 MOVEBENCH

We construct MOVEBENCH, a benchmark for systematically studying how VLA models behave across environment-motion patterns. Built on ManiSkill with the SAPIEN engine and illustrated in Fig. 3, MOVEBENCH centers on a pick task in which an xArm6 grasps objects of varied shapes, with only the target's motion regime varying across all environments. The regimes form three families (uniform translation, accelerated motion, and irregular motion) plus a static control. Uniform and accelerated regimes are each graded over 3 difficulty levels (detailed in Appendix B). Higher difficulty shrinks the temporal window available to react. The irregular family covers three discrete event types (random walk, stop-and-go, and teleport), each at a single level, since they admit no continuous tunable scalar and instead probe regime-change response. Across the ten environments, each provides 1000 demonstrations, totaling 10K trajectories and ∼460K frames. By fixing the task, manipulator, and scene across environments, MOVEBENCH isolates motion as the sole evaluation axis.

4.2 EXPERIMENTAL SETUP

We compare PPC against 9 baselines spanning 2 categories. The first category covers state-of-the-art foundational VLAs and general-purpose visuomotor policies trained on large-scale robot data. The second category covers training-free inference-time wrappers for chunked-action execution improvements and dynamic-focused methods. PPC is integrated as an inference-time wrapper on top of four foundational backbones as illustrated in Table 1, sharing the same MOVEBENCH-fine-tuned checkpoint with its bare counterpart, without additional training or architectural modification for PPC, while foundational baselines are fine-tuned under their official recipes. All methods share a synchronous simulation protocol that controls inference-delay effects (Appendix D). We choose $\pi_{0.5}$, the strongest foundational VLA, as the backbone for ACT, BID, and VLASH for fairness. We conduct 100 trials per motion setting, totaling 1,000 per method. PPC's configuration is fixed throughout the simulation experiments with $\lambda = 1, T = 16, K = 2$, and Stabilizer EMA rate $\beta_{\text{in}} = 0.3$ (the single free knob), giving an inner-EMA half-life of ~ 2 chunks under standard decay.

Table 1: **Success rate (%) on MOVEBENCH across all motion families.** PPC delivers improvements over every state-of-the-art foundational VLA, and PPC-equipped VLAs surpass all comparison baselines in success rate. The static column for PPC variants is marked with $*$ to indicate that PPC defaults to the underlying VLA when no environmental motion is present. Aggregate subscripts/superscripts give lower/upper offsets (pp) of 95% Clopper–Pearson intervals over pooled trials.

	Method	Static	Uniform Motion			Accelerated Motion			Irregular Motion			Average	
			Moving			Accelerating							
			Easy	Med.	Hard	Easy	Med.	Hard	Rand. Walk	Stop & Go	Teleport	Dyn. Only	All
Found.	Diffusion Policy (Chi et al., 2025)	75	56	60	21	43	28	17	63	50	56	$43.8^{+3.3}_{-3.3}$	$46.9^{+3.1}_{-3.1}$
	GR00T N1.6 (NVIDIA et al., 2025)	**88**	74	64	11	11	6	1	67	35	67	$37.3^{+3.3}_{-3.1}$	$42.4^{+3.1}_{-3.1}$
	SmolVLA (Shukor et al., 2025)	81	76	57	27	41	33	13	53	40	44	$42.7^{+3.3}_{-3.3}$	$46.5^{+3.1}_{-3.1}$
	π_0 (Black et al., 2024)	82	81	63	30	44	30	22	60	43	51	$47.1^{+3.3}_{-3.3}$	$50.6^{+3.1}_{-3.1}$
	$\pi_{0.5}$ (Intelligence et al., 2025)	80	85	78	34	58	43	29	54	48	60	$54.3^{+3.3}_{-3.3}$	$56.9^{+3.1}_{-3.1}$
Comp.	ACT (Zhao et al., 2023)	82	79	77	19	69	50	30	53	48	1	$47.3^{+3.4}_{-3.3}$	$50.8^{+3.1}_{-3.1}$
	BID (Liu et al., 2025c)	79	80	75	29	57	50	33	68	51	48	$54.6^{+3.3}_{-3.4}$	$57.0^{+3.1}_{-3.1}$
	VLASH (Tang et al., 2025)	68	57	47	14	41	26	15	40	36	16	$32.4^{+3.2}_{-3.0}$	$36.0^{+3.0}_{-3.0}$
	DynamicVLA (Xie et al., 2026)	70	73	57	20	45	42	29	49	40	24	$42.1^{+3.3}_{-3.2}$	$44.9^{+3.1}_{-3.1}$
Ours	GR00T + PPC	**88***	86 (+12%)	83 (+19%)	61 (+50%)	70 (+59%)	56 (+50%)	33 (+32%)	**78** (+11%)	54 (+19%)	**74** (+7%)	$66.1^{+3.1}_{-3.2}$ (+28.8%)	$68.3^{+2.9}_{-3.0}$ (+25.9%)
	SmolVLA + PPC	81*	69 (−7%)	69 (+12%)	58 (+31%)	58 (+17%)	59 (+26%)	35 (+22%)	60 (+7%)	**71** (+31%)	53 (+9%)	$59.1^{+3.2}_{-3.3}$ (+16.4%)	$61.3^{+3.0}_{-3.1}$ (+14.8%)
	π_0 + PPC	82*	86 (+5%)	76 (+13%)	67 (+37%)	73 (+29%)	65 (+35%)	57 (+35%)	71 (+11%)	67 (+24%)	52 (+1%)	$68.2^{+3.1}_{-3.1}$ (+21.1%)	$69.6^{+2.8}_{-3.0}$ (+19.0%)
	$\pi_{0.5}$ + PPC	80*	**88** (+3%)	**86** (+8%)	**70** (+36%)	**82** (+24%)	**72** (+29%)	**65** (+36%)	74 (+20%)	66 (+18%)	53 (−7%)	$\mathbf{72.9}^{+2.9}_{-3.0}$ (+18.6%)	$\mathbf{73.6}^{+2.7}_{-2.8}$ (+16.7%)

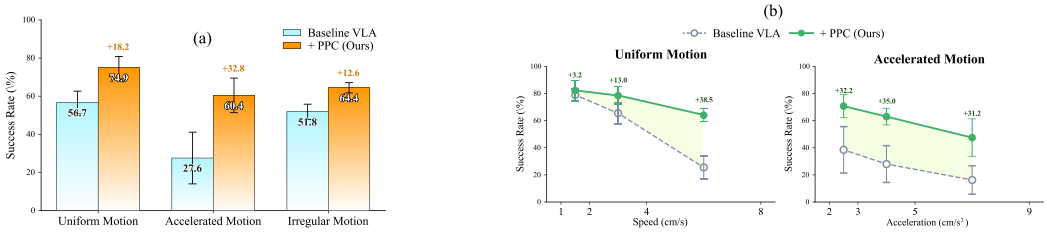


Figure 4: (a) Per-family success rate of baseline VLAs versus their PPC-equipped counterparts, averaged across four foundational backbones. PPC yields the largest gain on accelerated motion (+32.8). (b) Success rate as a function of physical speed and acceleration ranges. The gain from PPC grows monotonically with target speed in the uniform family while remaining consistently around +30 across the entire acceleration range.

4.3 MAIN RESULTS

Table 1 compares all methods across the ten MOVEBENCH environments.

PPC improves every foundational VLA across all motion families. Wrapping the four foundational VLAs with PPC raises their dynamic-only average by +16.4 to +28.8 absolute points, with the best-equipped variant ($\pi_{0.5}$+PPC) reaching 72.9% on dynamic environments and 73.6% overall. Since $\alpha^\star \to 1$ and $\delta_k^\star \to 0$ when $v = 0$, PPC degenerates to the identity by construction, preserving the full static capability without additional computation, while consistently improving performance across both regular and irregular motion regimes.

The gain is largest where dynamics blindness is most severe. Fig. 4 (a) shows that PPC yields its largest per-family improvement on accelerated motion (+32.8 averaged across backbones), followed by uniform (+18.2) and irregular (+12.6). This ordering directly reflects the closed-form structure. The Fibonacci-profile δ channel is designed to absorb the perpendicular residual that accumulates under sustained acceleration, explaining the largest gain in that family. Uniform motion is largely handled by the pace channel alone, while irregular regimes receive smaller but still positive gains as the latch-regulated cadence gate partially compensates for the weakened quasi-stationarity assumption. Fig. 4 (b) further reveals that the PPC gain grows monotonically with target speed in the uniform family (peaking at +38.5 at the hardest tier), while remaining consistently around +30 across the acceleration range, indicating that the second-order extension keeps pace with increasing acceleration.

Comparison with existing methods. Every PPC variant outperforms ACT and BID, supporting explicit motion compensation beyond chunk smoothing. DynamicVLA scores 44.9%, below bare $\pi_{0.5}$ at 56.9%, while VLASH reaches 36.0%, versus 73.6% for $\pi_{0.5}$+PPC. PPC's gains across these comparisons support correcting motion within action chunks. Appendix C provides further analysis.

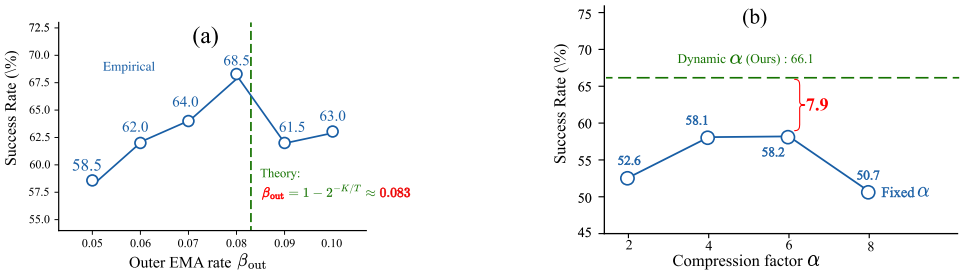


Figure 5: (a) The tested β_{out} sweep peaks near the derived value $\beta_{\text{out}} = 1 - 2^{-K/T} \approx 0.083$, supporting the latch calibration. (b) Dynamic α outperforms every tested fixed factor, supporting per-chunk adaptive compression.

4.4 Ablation Studies

We evaluate GR00T+PPC with 100 rollouts per dynamic environment, isolating the contributions of each correction channel, its closed-form structure, and the latch (Section 4.3). The matched velocity-access and execution-horizon controls below include both GR00T and $\pi_{0.5}$.

Closed-form structural ablations. Removing pace causes broad losses across all motion families (-28.3 points overall), whereas removing path predominantly harms accelerated motion (-24.2 overall). These distinct effects support their complementary roles in timing and spatial compensation. Removing the cosine projection harms accelerated motion but slightly improves irregular performance, illustrating the importance of matching correction direction to the motion regime. Replacing the Fibonacci profile with linear offsets loses 15.2 points overall, showing that the distribution of correction across the chunk matters beyond adding a spatial offset (Table 2).

Table 2: **Closed-form structural ablations.** Removing or modifying any closed-form component consistently degrades performance, confirming the necessity of every design choice.

Variant	Unif.	Accel.	Irreg.	Avg.
α w/o cos-θ	68.7 (-8.0)	29.3 (-23.7)	74.7 (+6.0)	57.6 (-8.5)
Linear δ	61.3 (-15.4)	23.3 (-29.7)	68.0 (-0.7)	50.9 (-15.2)
No α comp.	46.0 (-30.7)	22.3 (-30.7)	45.0 (-23.7)	37.8 (-28.3)
δ w/o $\perp$ proj.	68.7 (-8.0)	46.0 (-7.0)	64.3 (-4.4)	59.7 (-6.4)
No δ offsets	52.0 (-24.7)	9.3 (-43.7)	64.3 (-4.4)	41.9 (-24.2)
PPC (full)	**76.7**	**53.0**	**68.7**	**66.1**

Latch and second-order correction. Removing the latch costs 6.1 points overall, with an 8.0-point loss on irregular motion versus 2.7 on uniform motion. Removing the Lucas branch costs 3.4 points overall, with a 7.3-point loss on accelerated motion versus 0.4 on uniform motion. These distinct effects support selective cadence intervention under regime instability and second-order compensation under acceleration, complementing the first-order operator (Appendix E.1).

Adaptive compression and latch calibration. The EMA sweep peaks at 68.5% for $\beta_{\text{out}} = 0.08$, near the derived 0.083. Dynamic α reaches 66.1%, exceeding the best fixed factor's 58.2% (Fig. 5). The drop at $\alpha = 8$ further shows why uniformly stronger compression cannot replace per-chunk adaptation. These findings support calibrated latch decay and adaptive compression (Appendix E.1).

Robustness to perception noise. Even the lowest success rates across the noise sweep remain above the bare baseline, at 40% for regular motion and 60% for irregular motion (Appendix E.2).

Equal-access velocity control. With the same oracle velocity, PPC exceeds naive feedforward by 7.2 / 11.1 points on $\pi_{0.5}$ / GR00T in a matched rerun. Feedforward provides no gain over bare policies on irregular motion, whereas PPC surpasses it by 15.7/15.3 points on $\pi_{0.5}$/GR00T, isolating the benefit of structured correction beyond velocity access alone (Appendix G.1).

Execution-horizon controls. Shorter bare horizons help $\pi_{0.5}$ (SR 68.4%) but truncate GR00T's late grasp commands (SR 1.7%) with $K_{\text{exec}} = 2$, with 84% of closures occurring in the chunk's back half. PPC's adaptive schedule without correction scores only 61.0% / 35.1% on $\pi_{0.5}$ / GR00T, versus 73.9% / 65.7% for full PPC. These matched results support correction beyond the tested cadence-only controls and show that shortening the horizon alone is not uniformly effective (Appendix G.2).

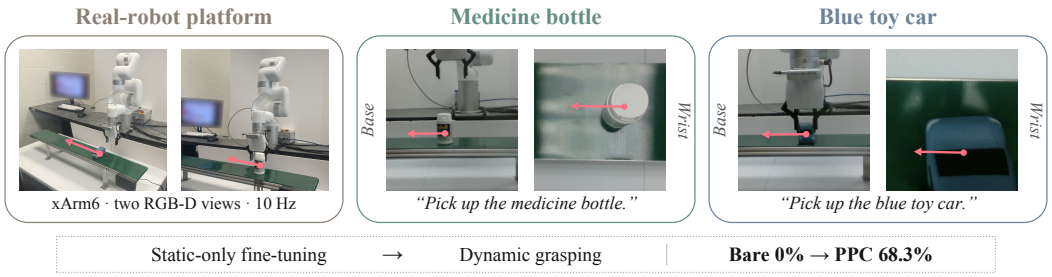


Figure 6: **Real-robot conveyor deployment.** Platform *(left)* and base/wrist views of the medicine bottle *(center)* and toy car *(right)*. Arrows indicate object motion along the conveyor. PPC enables dynamic grasping with a backbone fine-tuned only on static demonstrations.

Table 3: **Real-robot success with non-oracle tracking.** SR (%) with 95% Clopper–Pearson CIs, 20 trials per arm per row. Rows use two-sided Fisher's exact tests.

Regime	Bare SR [95% CI]	PPC SR [95% CI]	p
Static control	95.0 [75.1, 99.9]	100.0 [83.2, 100.0]	1.000
Constant slow (1–4 cm/s)	0.0 [0.0, 16.8]	**80.0** [56.3, 94.3]	1.5×10^{-7}
Constant fast (4–6 cm/s)	0.0 [0.0, 16.8]	**60.0** [36.1, 80.9]	4.5×10^{-5}
Speed-varying (2–6 cm/s)	0.0 [0.0, 16.8]	**65.0** [40.8, 84.6]	1.3×10^{-5}

4.5 Non-Oracle Perception and Real-Robot Validation

Non-oracle perception. In simulation, RGB-D tracking yields 68.7% / 57.1% success on $\pi_{0.5}$ / GR00T, retaining 73% / 70% of PPC's oracle gain. The measured errors $(\sigma_v, \sigma_\theta) = (0.21, 10.9^\circ)$ lie within the noise envelope tested in Appendix E.2. With a moving distractor, non-oracle PPC improves success from 18% to 42% and from 7% to 34%, respectively, supporting correction of the designated target despite competing motion. With oracle velocity, Cartesian-step and acceleration clamps retain gains of 17.3 / 25.5 points over bare, showing that the improvement persists under conservative actuation limits. Appendices G.3, G.5, and G.6 detail these tests.

Static-only fine-tuning, dynamic grasping. On an xArm6 conveyor with RGB-D tracking (Fig. 6), we wrap $\pi_{0.5}$ fine-tuned on 100 static demonstrations of two differently shaped objects. The dynamically fine-tuned simulation backbones use $\lambda = 1$, while hardware uses the same objective's full-compensation limit $\lambda \to 0$ to address the limited motion pursuit of a backbone fine-tuned solely on static demonstrations, retaining pace and latch (Appendix G.4). PPC raises moving-belt success from 0% to 68.3%, with significant gains in all three regimes ($p < 10^{-3}$) and high static success under the identity-pinned control (Table 3). These results validate the full-compensation deployment on a static-only fine-tuned backbone, complementing the regularized simulation experiments. Together, these settings support PPC's unified theoretical formulation for backbones fine-tuned on either static or dynamic demonstrations.

5 Conclusion

We present Pace-and-Path Correction (PPC), a closed-form, training-free, inference-time wrapper that explicitly compensates for environment dynamics in chunked-action VLAs. Decomposing the per-chunk correction objective into orthogonal pace and path channels, PPC introduces no learnable parameters and remains backbone-agnostic, deployable on any released VLA without retraining or architectural changes. We also introduce MoveBench, which isolates motion regime as the sole evaluation axis to systematically study chunked-VLA behavior under diverse motion patterns. PPC consistently improves foundational VLAs and exceeds the evaluated baselines under the controlled MoveBench protocol. Extensive experiments demonstrate that PPC consistently improves foundational VLAs and outperforms state-of-the-art dynamic-adaptive methods. Real-robot experiments further demonstrate PPC's effectiveness in real-world deployment, enabling dynamic manipulation with a backbone fine-tuned exclusively on static demonstrations.

REPRODUCIBILITY STATEMENT

To support reproducibility, Section 3 and Appendix A present PPC's formulation, assumptions, closed-form derivations, and stabilizer. Appendix B describes the construction of MOVEBENCH, including its motion regimes, object pool, and demonstration generation. Appendix D documents backbone fine-tuning, baseline deployment, PPC parameters, computational resources, evaluation seeds, and success criteria. Appendix G provides the protocols and results for additional matched controls and deployment experiments, with the non-oracle perception pipeline and real-robot setup detailed in Appendices G.3 and G.4, respectively. The hardware appendix specifies the full-compensation configuration recorded in session metadata, tracker gating, evaluation protocols, timing, and actuation limits.

AI USE STATEMENT

We used generative AI tools (ChatGPT and Claude) to help correct grammar, refine wording, implement parts of the method, and write scripts for data analysis and aggregation. The method, including its design and algorithmic decisions, was developed by the authors. We did not use generative AI to formulate research hypotheses, design experiments, develop theoretical or mathematical claims or their proofs, or interpret results. Other tasks requiring disclosure under the ICLR generative AI policy were not applicable to this work. The authors reviewed all AI-assisted work and verified and tested all AI-assisted code. We take full responsibility for the final content of this work, including its text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. $\pi 0$: A vision-language-action flow model for general robot control. *ArXiv*, abs/2410.24164, 2024.

Kevin Black, Manuel Y. Galliker, and Sergey Levine. Real-time execution of action chunking flow policies. *ArXiv*, abs/2506.07339, 2025a.

Kevin Black, Allen Z. Ren, Michael Equi, and Sergey Levine. Training-time action conditioning for efficient real-time chunking. *ArXiv*, abs/2512.05964, 2025b.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J. Joshi, Ryan C. Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael S. Ryoo, Grecia Salazar, Pannag R. Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Anand Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Ho Vuong, F. Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-1: Robotics transformer for real-world control at scale. *ArXiv*, abs/2212.06817, 2022.

Ben Burgess-Limerick, Christopher F. Lehnert, J. Leitner, and Peter Corke. Dgbench: An open-source, reproducible benchmark for dynamic grasping. *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3218–3224, 2022.

Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, Jiayan Guo, Xin Li, Yibing Song, Hao Luo, Fan Wang, Deli Zhao, and Hao Chen. Worldvla: Towards autoregressive action world model. *ArXiv*, abs/2506.21539, 2025.

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

Embodiment Collaboration, Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, Albert Tung, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anchit Gupta, Andrew Wang, Andrey Kolobov, Anikait Singh, Animesh Garg, Aniruddha Kembhavi, Annie Xie, Anthony

Brohan, Antonin Raffin, Archit Sharma, Arefeh Yavary, Arhan Jain, Ashwin Balakrishna, Ayzaan Wahid, Ben Burgess-Limerick, Beomjoon Kim, Bernhard Schölkopf, Blake Wulfe, Brian Ichter, Cewu Lu, Charles Xu, Charlotte Le, Chelsea Finn, Chen Wang, Chenfeng Xu, Cheng Chi, Chenguang Huang, Christine Chan, Christopher Agia, Chuer Pan, Chuyuan Fu, Coline Devin, Danfei Xu, Daniel Morton, Danny Driess, Daphne Chen, Deepak Pathak, Dhruv Shah, Dieter Büchler, Dinesh Jayaraman, Dmitry Kalashnikov, Dorsa Sadigh, Edward Johns, Ethan Foster, Fangchen Liu, Federico Ceola, Fei Xia, Feiyu Zhao, Felipe Vieira Frujeri, Freek Stulp, Gaoyue Zhou, Gaurav S. Sukhatme, Gautam Salhotra, Ge Yan, Gilbert Feng, Giulio Schiavi, Glen Berseth, Gregory Kahn, Guangwen Yang, Guanzhi Wang, Hao Su, Hao-Shu Fang, Haochen Shi, Henghui Bao, Heni Ben Amor, Henrik I Christensen, Hiroki Furuta, Homanga Bharadhwaj, Homer Walke, Hongjie Fang, Huy Ha, Igor Mordatch, Ilija Radosavovic, Isabel Leal, Jacky Liang, Jad Abou-Chakra, Jaehyung Kim, Jaimyn Drake, Jan Peters, Jan Schneider, Jasmine Hsu, Jay Vakil, Jeannette Bohg, Jeffrey Bingham, Jeffrey Wu, Jensen Gao, Jiaheng Hu, Jiajun Wu, Jialin Wu, Jiankai Sun, Jianlan Luo, Jiayuan Gu, Jie Tan, Jihoon Oh, Jimmy Wu, Jingpei Lu, Jingyun Yang, Jitendra Malik, João Silvério, Joey Hejna, Jonathan Booher, Jonathan Tompson, Jonathan Yang, Jordi Salvador, Joseph J. Lim, Junhyek Han, Kaiyuan Wang, Kanishka Rao, Karl Pertsch, Karol Hausman, Keegan Go, Keerthana Gopalakrishnan, Ken Goldberg, Kendra Byrne, Kenneth Oslund, Kento Kawaharazuka, Kevin Black, Kevin Lin, Kevin Zhang, Kiana Ehsani, Kiran Lekkala, Kirsty Ellis, Krishan Rana, Krishnan Srinivasan, Kuan Fang, Kunal Pratap Singh, Kuo-Hao Zeng, Kyle Hatch, Kyle Hsu, Laurent Itti, Lawrence Yunliang Chen, Lerrel Pinto, Li Fei-Fei, Liam Tan, Linxi "Jim" Fan, Lionel Ott, Lisa Lee, Luca Weihs, Magnum Chen, Marion Lepert, Marius Memmel, Masayoshi Tomizuka, Masha Itkina, Mateo Guaman Castro, Max Spero, Maximilian Du, Michael Ahn, Michael C. Yip, Mingtong Zhang, Mingyu Ding, Minho Heo, Mohan Kumar Srirama, Mohit Sharma, Moo Jin Kim, Muhammad Zubair Irshad, Naoaki Kanazawa, Nicklas Hansen, Nicolas Heess, Nikhil J Joshi, Niko Suenderhauf, Ning Liu, Norman Di Palo, Nur Muhammad Mahi Shafiullah, Oier Mees, Oliver Kroemer, Osbert Bastani, Pannag R Sanketi, Patrick "Tree" Miller, Patrick Yin, Paul Wohlhart, Peng Xu, Peter David Fagan, Peter Mitrano, Pierre Sermanet, Pieter Abbeel, Priya Sundaresan, Qiuyu Chen, Quan Vuong, Rafael Rafailov, Ran Tian, Ria Doshi, Roberto Martín-Martín, Rohan Baijal, Rosario Scalise, Rose Hendrix, Roy Lin, Runjia Qian, Ruohan Zhang, Russell Mendonca, Rutav Shah, Ryan Hoque, Ryan Julian, Samuel Bustamante, Sean Kirmani, Sergey Levine, Shan Lin, Sherry Moore, Shikhar Bahl, Shivin Dass, Shubham Sonawani, Shubham Tulsiani, Shuran Song, Sichun Xu, Siddhant Haldar, Siddharth Karamcheti, Simeon Adebola, Simon Guist, Soroush Nasiriany, Stefan Schaal, Stefan Welker, Stephen Tian, Subramanian Ramamoorthy, Sudeep Dasari, Suneel Belkhale, Sungjae Park, Suraj Nair, Suvir Mirchandani, Takayuki Osa, Tanmay Gupta, Tatsuya Harada, Tatsuya Matsushima, Ted Xiao, Thomas Kollar, Tianhe Yu, Tianli Ding, Todor Davchev, Tony Z. Zhao, Travis Armstrong, Trevor Darrell, Trinity Chung, Vidhi Jain, Vikash Kumar, Vincent Vanhoucke, Vitor Guizilini, Wei Zhan, Wenxuan Zhou, Wolfram Burgard, Xi Chen, Xiangyu Chen, Xiaolong Wang, Xinghao Zhu, Xinyang Geng, Xiyuan Liu, Xu Liangwei, Xuanlin Li, Yansong Pang, Yao Lu, Yecheng Jason Ma, Yejin Kim, Yevgen Chebotar, Yifan Zhou, Yifeng Zhu, Yilin Wu, Ying Xu, Yixuan Wang, Yonatan Bisk, Yongqiang Dou, Yoonyoung Cho, Youngwoon Lee, Yuchen Cui, Yue Cao, Yueh-Hua Wu, Yujin Tang, Yuke Zhu, Yunchu Zhang, Yunfan Jiang, Yunshuang Li, Yunzhu Li, Yusuke Iwasawa, Yutaka Matsuo, Zehan Ma, Zhuo Xu, Zichen Jeff Cui, Zichen Zhang, Zipeng Fu, and Zipeng Lin. Open x-embodiment: Robotic learning datasets and rt-x models, 2025. URL https://arxiv.org/abs/2310.08864.

Jingjing Fan, Yushan Liu, Shoujie Li, Botao Ren, Siyuan Li, Xiao-Ping Zhang, Wenbo Ding, and Zhidong Deng. Future-vla: Forecasting unified trajectories under real-time execution. *ArXiv*, abs/2602.15882, 2026.

Heng Fang, Shangru Li, Shuhan Wang, Xuanyang Xi, Dingkang Liang, and Xiang Bai. Towards generalizable robotic manipulation in dynamic environments. In *European Conference on Computer Vision (ECCV)*, 2026.

Yu Fang, Kanchana Ranasinghe, Le Xue, Honglu Zhou, Juntao Tan, Ran Xu, Shelby Heinecke, Caiming Xiong, Silvio Savarese, Daniel Szafir, Mingyu Ding, Michael S. Ryoo, and Juan Carlos Niebles. Robotic vla benefits from joint learning with motion image diffusion. *arXiv preprint arXiv:2512.18007*, 2025.

Jiayuan Gu, Fanbo Xiang, Xuanlin Li, Zhan Ling, Xiqiang Liu, Tongzhou Mu, Yihe Tang, Stone Tao, Xinyue Wei, Yunchao Yao, Xiaodi Yuan, Pengwei Xie, Zhiao Huang, Rui Chen, and Hao Su. Maniskill2: A unified benchmark for generalizable manipulation skills, 2023. URL https://arxiv.org/abs/2302.04659.

Mariam Hassan, Sebastian Stapf, Ahmad Rahimi, Pedro Rezende, Yasaman Haghighi, David Brüggemann, Isinsu Katircioglu, Lin Zhang, Xiaoran Chen, Suman Saha, et al. Gem: A generalizable ego-vision multimodal world model for fine-grained ego-motion, object dynamics, and scene composition control. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22404–22415, 2025.

Yafei Hu, Quanting Xie, Vidhi Jain, Jonathan Francis, Jay Patrikar, Nikhil Varma Keetha, Seungchan Kim, Yaqi Xie, Tianyi Zhang, Shibo Zhao, Yu Quan Chong, Chen Wang, Katia P. Sycara, Matthew Johnson-Roberson, Dhruv Batra, Xiaolong Wang, Sebastian Scherer, Zsolt Kira, Fei Xia, and Yonatan Bisk. Toward general-purpose robots via foundation models: A survey and meta-analysis. *ArXiv*, abs/2312.08782, 2023.

Zhiyu Huang, Yun Zhang, Johnson Liu, Rui Song, Chen Tang, and Jiaqi Ma. Tic-vla: A think-in-control vision-language-action model for robot navigation in dynamic environments. *ArXiv*, abs/2602.02459, 2026.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael 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 Rich Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. π0.5: a vision-language-action model with open-world generalization. *ArXiv*, abs/2504.16054, 2025.

Zhennan Jiang, Shangqing Zhou, Yutong Jiang, Zefang Huang, Mingjie Wei, Yuhui Chen, Tianxing Zhou, Zhen Guo, Hao Lin, Quanlu Zhang, Yu Wang, Haoran Li, Chao Yu, and Dongbin Zhao. Wovr: World models as reliable simulators for post-training vla policies with rl, 2026. URL https://arxiv.org/abs/2602.13977.

Motonari Kambara, Koki Seno, Tomoya Kaichi, Yanan Wang, and Komei Sugiura. Lilac: Language-conditioned object-centric optical flow for open-loop trajectory generation. *IEEE Robotics and Automation Letters*, 11: 6767–6774, 2026.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Paul Foster, Grace Lam, Pannag R. Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. *ArXiv*, abs/2406.09246, 2024.

Zhixuan Liang, Yizhuo Li, Tianshuo Yang, Chengyue Wu, Sitong Mao, Tian Nian, Liuao Pei, Shunbo Zhou, Xiaokang Yang, Jiangmiao Pang, et al. Discrete diffusion vla: Bringing discrete diffusion to action decoding in vision-language-action policies. *arXiv preprint arXiv:2508.20072*, 2025.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, qiang liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=xzEtNSuDJk.

Chenyv Liu, Wentao Tan, Lei Zhu, Fengling Li, Jingjing Li, Guoli Yang, and Heng Tao Shen. Self-correcting vla: Online action refinement via sparse world imagination, 2026a. URL https://arxiv.org/abs/2602.21633.

Jiaming Liu, Hao Chen, Pengju An, Zhuoyang Liu, Renrui Zhang, Chenyang Gu, Xiaoqi Li, Ziyu Guo, Sixiang Chen, Mengzhen Liu, Chengkai Hou, Mengdi Zhao, KC alex Zhou, Pheng-Ann Heng, and Shanghang Zhang. Hybridvla: Collaborative diffusion and autoregression in a unified vision-language-action model. *ArXiv*, abs/2503.10631, 2025a.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. *ArXiv*, abs/2410.07864, 2024.

Yicheng Liu, Shiduo Zhang, Zibin Dong, Baijun Ye, Tianyuan Yuan, Xiaopeng Yu, Linqi Yin, Chenhao Lu, Junhao Shi, Luca Jiang-Tao Yu, Liangtao Zheng, Tao Jiang, Jingjing Gong, Xipeng Qiu, and Hang Zhao. Faster: Toward efficient autoregressive vision language action modeling via neural action tokenization. *ArXiv*, abs/2512.04952, 2025b.

Yuejiang Liu, Jubayer Ibn Hamid, Annie Xie, Yoonho Lee, Max Du, and Chelsea Finn. Bidirectional decoding: Improving action chunking via guided test-time sampling. In *The Thirteenth International Conference on Learning Representations*, 2025c. URL https://openreview.net/forum?id=qZmn2hkuzw.

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 *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2026b. doi: 10.15607/RSS.2026.XXII.058.

Yueen Ma, Zixing Song, Yuzheng Zhuang, Jianye Hao, and Irwin King. A survey on vision-language-action models for embodied ai. *ArXiv*, abs/2405.14093, 2024.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 7(3):7327–7334, 2022.

Tongzhou Mu, Z. Ling, Fanbo Xiang, Derek Yang, Xuanlin Li, Stone Tao, Zhiao Huang, Zhiwei Jia, and Hao Su. Maniskill: Generalizable manipulation skill benchmark with large-scale demonstrations. In *NeurIPS Datasets and Benchmarks*, 2021.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. Robocasa: Large-scale simulation of everyday tasks for generalist robots. *ArXiv*, abs/2406.02523, 2024.

NVIDIA, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. Gr00t n1: An open foundation model for generalist humanoid robots, 2025. URL https://arxiv.org/abs/2503.14734.

Kohei Sendai, Maxime Alvarez, Tatsuya Matsushima, Yutaka Matsuo, and Yusuke Iwasawa. Leave no observation behind: Real-time correction for vla action chunks. *ArXiv*, abs/2509.23224, 2025.

Hao Shi, Bin Xie, Yingfei Liu, Lin Sun, Fengrong Liu, Tiancai Wang, Erjin Zhou, Haoqiang Fan, Xiangyu Zhang, and Gao Huang. MemoryVLA: Perceptual-cognitive memory in vision-language-action models for robotic manipulation. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=54U3XHf7qq.

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andrés Marafioti, Simon Alibert, Matthieu Cord, Thomas Wolf, and Rémi Cadène. Smolvla: A vision-language-action model for affordable and efficient robotics. *ArXiv*, abs/2506.01844, 2025.

Wenxuan Song, Jiayi Chen, Pengxiang Ding, Han Zhao, Wei Zhao, Zhide Zhong, Zongyuan Ge, Zhijun Li, Donglin Wang, Lujia Wang, Jun Ma, and Haoang Li. Pd-vla: Accelerating vision-language-action model integrated with action chunking via parallel decoding. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 13162–13169, 2025. doi: 10.1109/IROS60139.2025.11247519.

Xudong Tan, Yaoxin Yang, Peng Ye, Jialin Zheng, Bizhe Bai, Xinyi Wang, Jia Hao, and Tao Chen. Think twice, act once: Token-aware compression and action reuse for efficient inference in vision-language-action models. *arXiv preprint arXiv:2505.21200*, 2025.

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. *ArXiv*, abs/2512.01031, 2025.

Octo Model Team, Dibya Ghosh, Homer Rich Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Pannag R. Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. *ArXiv*, abs/2405.12213, 2024.

Xinkai Wang, Chenyi Wang, Yifu Xu, Mingzhe Ye, Fucheng Zhang, Jialin Tian, Xinyu Zhan, Lifeng Zhu, Cewu Lu, and Lixin Yang. Lamp: Learning vision-language-action policy with 3d scene flow as latent motion prior, 2026. URL https://arxiv.org/abs/2603.25399.

Qingpeng Wen, Haomin Zhu, Yuepeng Zhang, Linzhong Xia, Bo Gao, and Zhuozhen Li. Adaptive action chunking for robotic imitation learning. *Biomimetics*, 2026.

Haozhe Xie, Beichen Wen, Jiarui Zheng, Zhaoxi Chen, Fangzhou Hong, Haiwen Diao, and Ziwei Liu. Dynamicvla: A vision-language-action model for dynamic object manipulation. *arXiv preprint arXiv:2601.22153*, 2026.

Siyu Xu, Yunke Wang, Chenghao Xia, Dihao Zhu, Tao Huang, and Chang Xu. VLA-cache: Efficient vision-language-action manipulation via adaptive token caching. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=QZYZ0Xm58q.

Jiahui Zhang, Yurui Chen, Yueming Xu, Ze Huang, Yanpeng Zhou, Yu-Jie Yuan, Xinyue Cai, Guowei Huang, Xingyue Quan, Hang Xu, et al. 4d-vla: Spatiotemporal vision-language-action pretraining with cross-scene calibration. *Advances in Neural Information Processing Systems*, 38:33914–33937, 2026.

Jianke Zhang, Yanjiang Guo, Xiaoyu Chen, Yen-Jen Wang, Yucheng Hu, Chengming Shi, and Jianyu Chen. Hirt: Enhancing robotic control with hierarchical robot transformers. *arXiv preprint arXiv:2410.05273*, 2024.

Shiduo Zhang, Zhe Xu, Peiju Liu, Xiaopeng Yu, Yuan Li, Qinghui Gao, Zhaoye Fei, Zhangyue Yin, Zuxuan Wu, Yu-Gang Jiang, et al. Vlabench: A large-scale benchmark for language-conditioned robotics manipulation with long-horizon reasoning tasks. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11142–11152. IEEE, 2025a.

Thomas T Zhang, Daniel Pfrommer, Chaoyi Pan, Nikolai Matni, and Max Simchowitz. Action chunking and exploratory data collection yield exponential improvements in behavior cloning for continuous control. *arXiv preprint arXiv:2507.09061*, 2025b.

Wenyao Zhang, Hongsi Liu, Zekun Qi, Yunnan Wang, Xinqiang Yu, Jiazhao Zhang, Runpei Dong, Jiawei He, He Wang, Zhizheng Zhang, Li Yi, Wenjun Zeng, and Xin Jin. Dreamvla: A vision-language-action model dreamed with comprehensive world knowledge. *ArXiv*, abs/2507.04447, 2025c.

Yifan Zhang, Ruiping Wang, and Xilin Chen. Dynamic behavior cloning with temporal feature prediction: Enhancing robotic arm manipulation in moving object tasks. *IEEE Robotics and Automation Letters*, 10: 5209–5216, 2025d.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware, 2023. URL https://arxiv.org/abs/2304.13705.

Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daumé III, Andrey Kolobov, Furong Huang, and Jianwei Yang. Tracevla: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. In *International Conference on Learning Representations*, volume 2025, pp. 54277–54296, 2025.

Hongyan Zhi, Peihao Chen, Siyuan Zhou, Yubo Dong, Quanxi Wu, Lei Han, and Mingkui Tan. 3dflowaction: Learning cross-embodiment manipulation from 3d flow world model, 2025. URL https://arxiv.org/abs/2506.06199.

Zhide Zhong, Haodong Yan, Junfeng Li, Xiangchen Liu, Xin Gong, Tianran Zhang, Wenxuan Song, Jiayi Chen, Xinhu Zheng, Hesheng Wang, et al. Flowvla: Visual chain of thought-based motion reasoning for vision-language-action models. *IEEE Transactions on Robotics*, 2026.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In Jie Tan, Marc Toussaint, and Kourosh Darvish (eds.), *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183. PMLR, 06–09 Nov 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.

A Full Closed-Form Mathematical Derivation

This appendix provides the complete mathematical derivation of the Pace-and-Path Correction operator summarized in Sections 3. The derivation uses the quadratic cost introduced in Section 3.1 under A1–A3. The second-order extension retains the orthogonal pace–path parameterization specified in Appendix A.8.

A.1 Assumptions

Three working assumptions underlie the derivation:

- **A1.** *Quasi-stationary plan.* The per-step delta $\Delta p_i \approx \Delta p$ is approximately constant within the executed chunk window $i = 0, \ldots, K-1$.
- **A2.** *Slowly-varying disturbance.* The velocity v and direction $\hat{d}$ are approximately constant over the K executed env-steps (relaxed to affine variation in Section A.8).
- **A3.** *Small rotation.* Per-step rotations are small enough for additive xyz deltas across env-steps.

The closed-form optimum applies to the local model under these assumptions. Chunkwise recomputation updates the correction as observations change, while the stateful latch limits execution length under detected irregularity.

A.2 Cost Function

Under A1, the corrected env-step k delta is $u_k = \alpha\Delta p + \delta_k$ with $\alpha \geq 1$ and $\delta_k \in \mathbb{R}^3$. The cumulative arm position is $p_j = \sum_{k=0}^{j-1} u_k = j\alpha\Delta p + \sigma_j$ where $\sigma_j := \sum_{k=0}^{j-1} \delta_k$. The ideal trajectory under disturbance $(v, \hat{d})$ is $\tilde{p}_j = j(\Delta p + v\hat{d})$. The cost balances per-waypoint tracking and per-step offset effort:

$$L(\alpha, \{\delta_k\}) = \frac{1}{2}\sum_{j=1}^{K} \|p_j - \tilde{p}_j\|^2 + \frac{1}{2}\sum_{k=0}^{K-1} \|\delta_k\|^2. \tag{1}$$

The penalty is on δ_k only, not on α, because α moves the arm along the planned direction (temporal compression with no directional deviation), whereas δ_k introduces off-plan spatial offset. This asymmetry is the unique choice satisfying three invariants simultaneously: (i) uniqueness of the optimum, (ii) degeneracy to the baseline VLA at $v = 0$, and (iii) the cosine projection structure in $\alpha^\star$.

A.3 Reduced Quantities and Stationarity Conditions

Define the residual disturbance $A := v\hat{d} - (\alpha - 1)\Delta p$. The tracking error simplifies to

$$e_j := p_j - \tilde{p}_j = -jA + \sigma_j, \qquad j = 1, \ldots, K. \tag{2}$$

Setting $\partial L/\partial\alpha = 0$ and $\partial L/\partial\delta_k = 0$ yields the joint stationarity conditions:

$$\frac{\partial L}{\partial \alpha} = -\Delta p \cdot \sum_{j=1}^{K} j\, e_j = 0, \tag{3}$$

$$\frac{\partial L}{\partial \delta_k} = \delta_k + \sum_{j=k+1}^{K} e_j = 0, \qquad k = 0, \ldots, K-1. \tag{4}$$

A.4 Derivation of $\alpha^\star$ (Pace Channel)

The cost (1) is rotationally invariant in $\mathbb{R}^3$. At the optimum, every δ_k inherits the direction of A: $\delta_k = c_k A$ for some scalar c_k. Consequently σ_j and e_j are parallel to A, so $\sum_j j\, e_j$ is also parallel to A. Condition (3) then collapses to

$$\Delta p \cdot A = 0. \tag{5}$$

Expanding $A = v\hat{d} - (\alpha - 1)\Delta p$:

$$\Delta p \cdot \big(v\hat{d} - (\alpha - 1)\Delta p\big) = 0 \quad \Longrightarrow \quad (\alpha - 1)\|\Delta p\|^2 = v\,(\hat{d} \cdot \Delta p) = v\|\Delta p\| \cos\theta, \tag{6}$$

where $\cos\theta := \hat{d} \cdot \widehat{\Delta p}$. Solving:

$$\boxed{\alpha^\star = 1 + \frac{v\cos\theta}{\|\Delta p\|}.} \tag{7}$$

Substituting $\alpha^\star$ back into A yields the orthogonal residual:

$$A^\star = v\hat{d} - v\cos\theta\, \widehat{\Delta p} = v\hat{d}_\perp, \tag{8}$$

which lies entirely perpendicular to the planned direction $\widehat{\Delta p}$.

Clamping. The physical constraint $\alpha \geq 1$ is violated when $\cos\theta < 0$ (antagonistic motion). In this case $\alpha^\star$ is clamped to 1, and the full disturbance $v\hat{d}$ passes to the path channel. The analytical expression has no finite upper bound. Execution is governed by the chunk budget and dynamic horizon (Section A.9); the reported simulation compression is bounded by T/K (Appendix E).

A.5 Derivation of $\delta_k^\star$ (Path Channel, Fibonacci Profile)

The path channel handles $A^\star$, which $\alpha^\star$ cannot absorb. Differencing condition (4) in k and using $\sigma_{j+1} - \sigma_j = \delta_j$ yields the 2D linear recurrence

$$\begin{pmatrix} \delta_{k+1} \\ e_{k+1} \end{pmatrix} = \underbrace{\begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix}}_{M} \begin{pmatrix} \delta_k \\ e_k \end{pmatrix} - \begin{pmatrix} A^\star \\ A^\star \end{pmatrix}, \tag{9}$$

with boundary conditions $e_0 = 0$ and $\delta_K = 0$.

Eigenstructure. The companion matrix M has characteristic polynomial $\lambda^2 - 3\lambda + 1 = 0$, yielding eigenvalues

$$\lambda_\pm = \frac{3 \pm \sqrt{5}}{2} = \varphi^{\pm 2}, \qquad \varphi = \frac{1+\sqrt{5}}{2} \text{ (golden ratio).} \tag{10}$$

Particular solution. Setting $\delta_{k+1} = \delta_k =: \delta^p$ and $e_{k+1} = e_k =: e^p$ in (9) gives $(\delta^p, e^p) = (A^\star, 0)$.

Homogeneous solution. The eigenvectors of M are $w_+ = (\varphi, 1)^\top$ and $w_- = (1, -\varphi)^\top$. The general solution is

$$\begin{pmatrix} \delta_k \\ e_k \end{pmatrix} = \begin{pmatrix} A^\star \\ 0 \end{pmatrix} + c_+ \varphi^{2k} \begin{pmatrix} \varphi \\ 1 \end{pmatrix} + c_- \varphi^{-2k} \begin{pmatrix} 1 \\ -\varphi \end{pmatrix}. \tag{11}$$

Boundary conditions. From $e_0 = 0$: $c_+ - \varphi\, c_- = 0$, so $c_+ = \varphi\, c_-$. From $\delta_K = 0$:

$$A^\star + c_- \left(\varphi^{2K+2} + \varphi^{-2K}\right) = 0. \tag{12}$$

Factoring out φ and applying the identity $\varphi^n + \varphi^{-n} = \sqrt{5}\, F_n$ for odd n (where F_n is the n-th Fibonacci number with $F_1 = F_2 = 1$), we obtain

$$\varphi^{2K+1} + \varphi^{-(2K+1)} = \sqrt{5}\, F_{2K+1}, \tag{13}$$

Hence $c_- = -A^\star / (\varphi \sqrt{5}\, F_{2K+1})$ and $c_+ = \varphi c_-$. Substituting back and collecting:

$$\boxed{\delta_k^\star = \left(1 - \frac{F_{2k+1}}{F_{2K+1}}\right) A^\star, \qquad k = 0, \ldots, K-1.} \tag{14}$$

Profile properties. The Fibonacci ratio F_{2k+1}/F_{2K+1} increases monotonically in k from $F_1/F_{2K+1} = 1/F_{2K+1} \approx 0$ to $F_{2K-1}/F_{2K+1} \to \varphi^{-2} \approx 0.382$ as $K \to \infty$. Thus $\delta_0^\star \approx A^\star$ (near-full compensation at chunk start), decaying to $\delta_{K-1}^\star \approx 0.618\, A^\star$, with $\delta_K^\star = 0$ specifying an auxiliary terminal condition outside the executed range $k < K$.

Verification. For $K = 2$: $F_5 = 5$, giving $\delta_0^\star = (1 - 1/5)A^\star = \frac{4}{5}A^\star$ and $\delta_1^\star = (1 - 2/5)A^\star = \frac{3}{5}A^\star$. Direct substitution into (4) confirms both conditions are satisfied.

A.6 Orthogonal Decomposition

For the first-order interior optimum, the channels act on disjoint subspaces. $\alpha^\star$ absorbs $v \cos\theta\, \widehat{\Delta p}$ (the component of $v\hat{d}$ parallel to Δp), while $\delta_k^\star$ absorbs $A^\star = v\hat{d}_\perp$ (the perpendicular residual). Both closed forms depend on the chunk geometry $(\Delta p, K)$ and dynamics signal $(v, \hat{d})$. When the pace lower bound is active, the residual follows the clamped case in Appendix A.4.

A.7 General λ-Regularization

Replacing the unit weight on the effort term with a general $\lambda > 0$:

$$L_\lambda = \frac{1}{2} \sum_{j=1}^{K} \|e_j\|^2 + \frac{\lambda}{2} \sum_{k=0}^{K-1} \|\delta_k\|^2. \tag{15}$$

$\alpha^\star$ **is λ-independent.** The factor λ multiplies only the δ-effort term. For an interior optimum, rotation invariance still gives $\delta_k \parallel A$ and $\Delta p \cdot A = 0$, so the $\alpha^\star$ formula (7) is unchanged for all $\lambda > 0$.

$\delta_k^\star(\lambda)$ **in hyperbolic cosine form.** The modified recurrence becomes

$$\begin{pmatrix}\delta_{k+1}\\ e_{k+1}\end{pmatrix} = \begin{pmatrix}1+1/\lambda & 1/\lambda\\ 1 & 1\end{pmatrix}\begin{pmatrix}\delta_k\\ e_k\end{pmatrix} + \begin{pmatrix}-A^\star/\lambda\\ -A^\star\end{pmatrix}. \tag{16}$$

The eigenvalues $\mu_\pm$ satisfy $\mu + \mu^{-1} = 2 + 1/\lambda$, giving $\mu_+ = e^{\omega(\lambda)}$ with

$$\omega(\lambda) = \operatorname{arccosh}\left(1 + \frac{1}{2\lambda}\right). \tag{17}$$

Solving under the same boundary conditions:

$$\boxed{\delta_k^\star(\lambda) = \left(1 - \frac{\cosh\left((k+\frac{1}{2})\,\omega\right)}{\cosh\left((K+\frac{1}{2})\,\omega\right)}\right) A^\star.} \tag{18}$$

Limiting cases.

- $\lambda = 1$: $\omega = 2\ln\varphi$, and $\cosh((2k{+}1)\ln\varphi) = \frac{\sqrt{5}}{2}F_{2k+1}$, recovering Fibonacci profile (14).
- $\lambda \to 0$: $\omega \to \infty$, so $\delta_k^\star \to A^\star$ for all $k < K$ (no penalty, full compensation).
- $\lambda \to \infty$: $\omega \to 0$, so $\delta_k^\star \to 0$ (high penalty, no spatial offset).

The weight λ controls how strongly the path channel trades residual tracking error against spatial correction effort. The $\lambda \to 0$ deployment is taken as the limit of the $\lambda > 0$ solutions, retaining their pace solution and, for an interior optimum, orthogonal decomposition. Thus, full compensation and the Fibonacci profile are two instances of the same objective.

In simulation, the configuration uses $\lambda = 1/K_{\text{kal}}$ with the oracle-parity sentinel $K_{\text{kal}} \to 1$, yielding $\lambda = 1$ for the reported results, including non-oracle evaluations. More generally, this mapping suppresses spatial offsets as confidence decreases. The real-robot deployment explicitly selects the $\lambda \to 0$ limit, independently of this mapping; the tracker gates immature estimates while retaining the same sentinel (Appendix G.4).

A.8 Second-Order Extension (Acceleration)

We relax A2 to an affine-in-time disturbance $v(t) = v_0 + at$ with $t = 0, \ldots, K{-}1$. The cumulative target offset under midpoint integration becomes $jv_0 + \frac{1}{2}j^2 a$, modifying the ideal trajectory to $\tilde{p}_j = j\Delta p + (jv_0 + \frac{1}{2}j^2 a)\hat{d}$.

Define the two-component disturbance: $A := v_0\hat{d} - (\alpha - 1)\Delta p$ (first-order) and $B := \frac{1}{2}a\hat{d}$ (second-order). The tracking error becomes $e_j = -jA - j^2 B + \sigma_j$.

Orthogonal channel parameterization. The second-order expressions retain the allocation of parallel tracking to pace and perpendicular tracking to path: $\Delta p \cdot \delta_k = 0$. For constant disturbance this allocation follows from joint stationarity; for the affine extension it specifies the correction class over which the same cost is minimized. Hence $\Delta p \cdot \sigma_j = 0$, and the parallel and perpendicular losses separate. The formulas below give the interior pace proposal before execution bounds.

$\alpha^\star$ **under acceleration.** Under this parameterization, $\Delta p \cdot \sigma_j = 0$. Setting $\partial L/\partial\alpha = 0$ and using the sums $S_2 = K(K{+}1)(2K{+}1)/6$ and $S_3 = [K(K{+}1)/2]^2$:

$$\boxed{\alpha^\star = 1 + \frac{v_0\cos\theta_v}{\|\Delta p\|} + \frac{S_3}{2S_2}\cdot\frac{a\cos\theta_a}{\|\Delta p\|},} \tag{19}$$

where $\cos\theta_v = \hat{d}_v \cdot \widehat{\Delta p}$ and $\cos\theta_a = \hat{d}_a \cdot \widehat{\Delta p}$. The coupling coefficient

$$\frac{S_3}{2S_2} = \frac{3K(K+1)}{4(2K+1)} \xrightarrow{K\to\infty} \frac{3K}{8} \tag{20}$$

scales linearly in K, reflecting longer integration of acceleration. Setting $a = 0$ recovers (7).

$\delta_k^\star$ under acceleration (Lucas profile). In the perpendicular subspace, let $A^\star = v_0 \hat{d}_{v,\perp}$ and $B^\star = \frac{1}{2} a \hat{d}_{a,\perp}$. At $\lambda = 1$, the recurrence (9) acquires quadratic forcing and decomposes additively:

$$\boxed{\delta_k^\star = \underbrace{\left(1 - \frac{F_{2k+1}}{F_{2K+1}}\right) A^\star}_{\text{Fibonacci (first-order)}} + \underbrace{\Lambda_k(K)\, B^\star}_{\text{Lucas-polynomial (second-order)}}}, \tag{21}$$

where $\Lambda_k(K)$ is the Lucas-polynomial profile coefficient:

$$\Lambda_k(K) = (2k+1) - L_{2k+1} + \frac{F_{2k+1}}{F_{2K+1}}\big(L_{2K+1} - (2K+1)\big), \tag{22}$$

with L_n denoting the Lucas numbers ($L_0 = 2$, $L_1 = 1$, $L_n = L_{n-1} + L_{n-2}$). The Lucas profile is the natural dual to Fibonacci on the same eigenvalue structure $\varphi^{\pm 2}$: the j^2-quadratic forcing activates the second homogeneous mode (L_{2k+1} branch), which is silent in first-order because $L_0 = 2$ contradicts the boundary condition $e_0 = 0$.

A.9 Dynamic Execution Horizon K_{exec}

The closed-form pace proposal is coupled to the following dynamic execution-horizon rule:

$$K_{\text{exec}}(\alpha) = \max\Big(K,\ \min\big(\lceil T/\alpha \rceil,\ T\big)\Big). \tag{23}$$

At $\alpha = 1$, $K_{\text{exec}} = T$ (full chunk consumed). As α increases, K_{exec} shrinks, reaching the floor K at $\alpha \geq T/K$. This rule bounds the execution length, not the analytical pace proposal. The reported simulation configuration also limits execution compression to the chunk budget; Appendix E reports this bounded quantity using the notation $\alpha^\star$. The Fibonacci profile (14) uses K_{exec} for K to match the actual execution window.

A.10 Hierarchical 2-EMA Latch Stabilizer

The closed-form correction is exact under A2. Irregular regimes (random walk, stop-and-go, teleport) violate A2 chronically, requiring a sustained cap on K_{exec} beyond what the single-chunk direction trust ρ_t provides. The latch admits a single free hyperparameter β_{in}; all other constants derive from the chunk geometry (K, T).

Direction-shift trigger. At each chunk reset t:

$$\tau_t = \mathbf{1}\big[\rho_{\text{gt}}(t) < \tfrac{1}{2}\big], \qquad \rho_{\text{gt}}(t) = \max\left(0,\ \frac{v_t \cdot v_{t-1}}{\|v_t\|\,\|v_{t-1}\|}\right). \tag{24}$$

The threshold $1/2$ is the natural midpoint of $\rho_{\text{gt}} \in [0, 1]$.

Outer EMA (chronic trigger rate).

$$C_t = \beta_{\text{out}}\, \tau_t + (1 - \beta_{\text{out}})\, C_{t-1}. \tag{25}$$

Sticky factor.

$$s_t = \frac{C_t}{C_t + R_{\text{TH}}}. \tag{26}$$

Inner EMA with sticky-modulated decay.

$$L_t = \begin{cases} \beta_{\text{in}} + (1 - \beta_{\text{in}})\, L_{t-1}, & \tau_t = 1, \\ \big[1 - \beta_{\text{in}}(1 - s_t)\big]\, L_{t-1}, & \tau_t = 0. \end{cases} \tag{27}$$

Under chronic instability ($s_t \to 1$), the effective decay rate vanishes and L_t holds near its current value. Under occasional triggers ($s_t \to 0$), L_t decays at the standard rate β_{in}.

Latch output. $m_t = \mathbf{1}[L_t > L_{\text{th}}]$. When $m_t = 1$, the execution horizon is capped: $K_{\text{exec}} \leq T/4$.

Derived constants. All three internal thresholds derive from β_{in} and (K, T):

(a) *Outer EMA rate.* Match the outer half-life to one chunk-budget cycle (T/K chunks at $\alpha = 1$). Solving $(1 - \beta_{\text{out}})^{T/K} = 1/2$:

$$\beta_{\text{out}} = 1 - 2^{-K/T}. \tag{28}$$

For $K = 2, T = 16$: $\beta_{\text{out}} \approx 0.083$.

(b) *Active threshold L_{th}.* Under standard decay ($s_t = 0$), a single trigger at $t = 0$ followed by n non-trigger steps gives $L_n = \beta_{\text{in}}(1 - \beta_{\text{in}})^n$. Setting $L_{\text{th}} = L_2$ calibrates a two-chunk active period under this standard-decay reference (including the triggering chunk):

$$L_{\text{th}} = \beta_{\text{in}}(1 - \beta_{\text{in}})^2. \tag{29}$$

For $\beta_{\text{in}} = 0.3$: $L_{\text{th}} \approx 0.147$. In the coupled latch, a trigger also raises C_t, giving $s_t > 0$ and slower subsequent decay. Thus, two chunks specify the threshold calibration, while actual persistence depends on the outer state.

(c) *Sticky reference.* s_t reaches $1/2$ at $C_t = R_{\text{TH}}$. The natural reference is the same scale as the active threshold:

$$R_{\text{TH}} = L_{\text{th}}. \tag{30}$$

Regime behavior summary.

Regime	L_t behavior	m_t	K_{exec}
Stable (uniform/accel.)	$\to 0$	0	full chunk budget
Single isolated event	spike, sticky-modulated decay	brief 1	brief $T/4$ cap
Chronic (random walk)	sticks high ($s_t \to 1$)	persistent 1	$T/4$
Periodic (stop-and-go)	bursty with slow decay	intermittent 1	intermittent cap

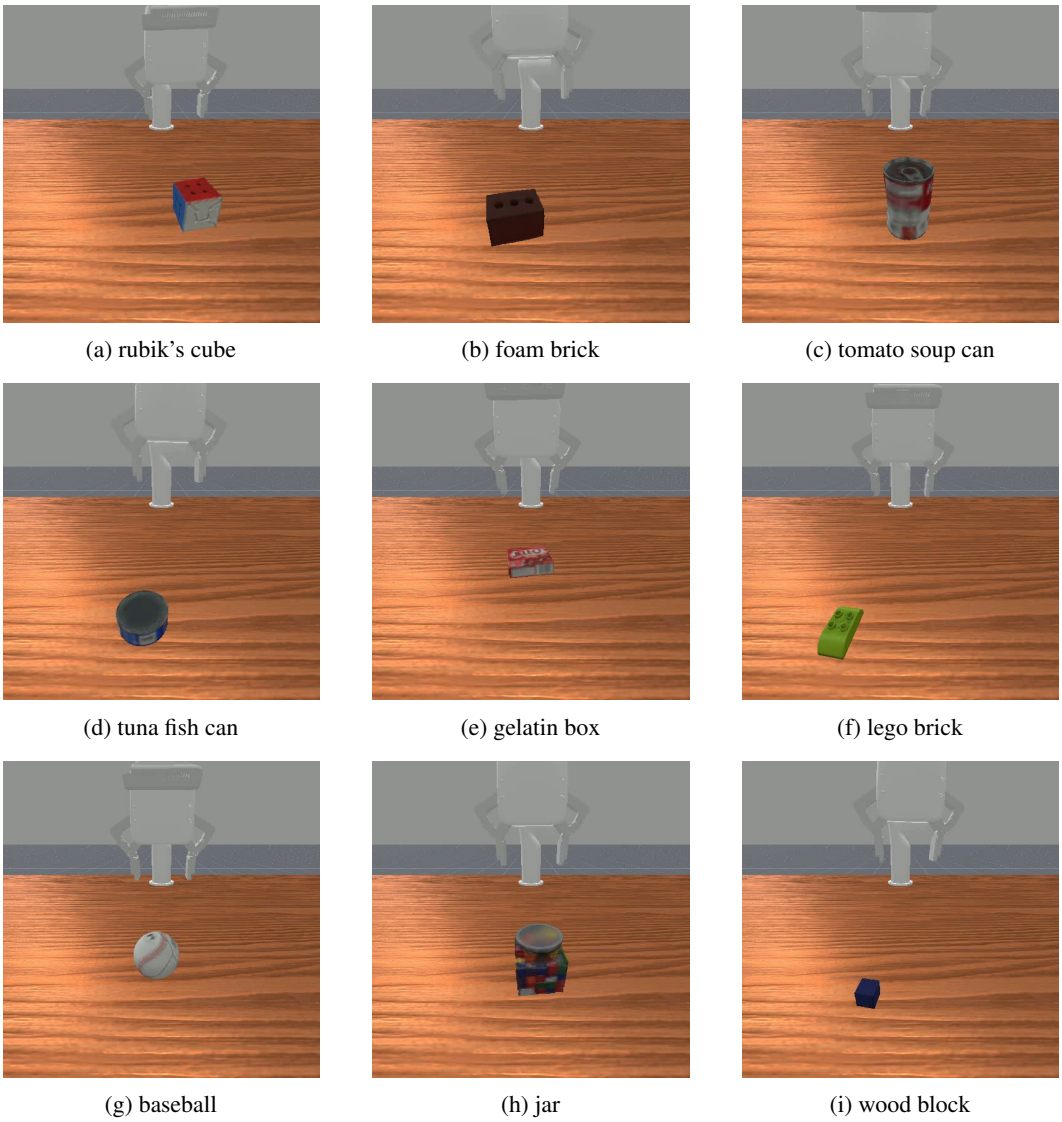
Grasp reset. When the TCP-to-object distance falls below the gripper half-span ($\|p_{\text{tcp}} - p_{\text{obj}}\| < r_{\text{grip}}$), the object transitions from external dynamics to internal state of the manipulator. The latch state is reset: $(L_t, C_t) \leftarrow (0, 0)$.

B MoveBench Details

This section expands MoveBench's implementation details summarized in the main text. We control simulation at 20 Hz via ManiSkill's SAPIEN backend, with 512×512 RGB streams from a fixed overhead and a wrist-mounted camera, 7-DoF proprioception, and a language-instruction channel. Each episode admits at most 200 environment steps (10 s wall-clock) and is judged successful when the target is grasped and lifted by ≥ 3 cm, matching the protocol established by the foundational VLA suites we benchmark against. The target is sampled uniformly from a pool of 9 YCB household objects covering a broad range of geometries (cube, brick, cylinder, sphere, flat box, jar) and physical sizes (25–103 mm along the dominant axis), exposing the policy to grasp-pose ambiguity orthogonal to the motion variable. The following subsections detail the object pool, motion-regime parametrization, demonstration generation pipeline, and reported diagnostic statistics.

B.1 Object Pool

The grasp target in every MoveBench environment is sampled uniformly at episode reset from a fixed pool of nine YCB household objects, shown in Fig. 7. The pool is intentionally small but geometrically heterogeneous, spanning the cube, rectangular brick, tall cylinder, flat cylinder, flat box, elongated lego, sphere, jar, and tiny block primitives, with object scales adjusted so that all targets fit within a 25–103 mm dominant-axis range and remain reachable by the xArm6 gripper. This diversity ensures that the policy must commit to an object-specific grasp pose at the moment of contact, which interacts non-trivially with the motion regime: a motion handler that succeeds by approaching from a fixed direction collapses once the pool forces approach-direction variation. The object identity is exposed only through the natural-language instruction "`Pick up the {name}.`", so the policy must ground each name to its visual appearance under the episode's sampled motion regime.



(a) rubik's cube (b) foam brick (c) tomato soup can

(d) tuna fish can (e) gelatin box (f) lego brick

(g) baseball (h) jar (i) wood block

Figure 7: The nine YCB objects sampled in MOVEBENCH. Each panel is the base-camera frame at $t{=}0$ from a demonstration episode of the corresponding task.

B.2 MOTION REGIMES

A deterministic motion-update rule, sampled per episode and applied to the target each simulation tick (20 Hz), fully specifies each environment. The nine dynamic environments form three families.

Uniform translation. The object is initialized with a single uniformly-sampled direction in the table plane and a constant speed drawn from a regime-specific range, $[1, 2]$, $[2, 4]$, and $[4, 8]$ cm/s for the easy, medium, and hard tier respectively. No further randomization is applied during the episode, so the regime is fully characterized by the initial speed magnitude.

Accelerated motion. The object is initialized with a low base speed $v_0 \in [2, 3]$ cm/s, common to all three tiers, and a per-episode acceleration vector whose magnitude is drawn from $[2, 3]$, $[3, 5]$, and $[5, 9]$ cm/s^2 for easy, medium, and hard. With v_0 decoupled from the acceleration tier, any cross-tier performance gap reflects only the second-order signal, without an initial-speed confound.

Irregular motion. Three regime-change patterns probe non-stationary behavior. *Random Walk* maintains a constant 5 cm/s speed but resamples a fresh planar direction every 5–12 ticks, simulating

continuous reactive disturbances at the timescale of a single chunk. *Stop-and-Go* alternates between 7 cm/s uniform motion for 3–7 ticks and full pauses for 3–6 ticks, presenting a binary on/off velocity signal that rewards opportunistic grasping during pause windows. *Teleport* keeps the object stationary except for two mid-episode discontinuities, the first scheduled within ticks 3–10 and each requiring a minimum displacement of 8 cm from the current position, an event whose magnitude exceeds any plausible per-chunk plan correction. Each pattern has a single difficulty level because it probes regime-change response rather than continuously tunable intensity; introducing a difficulty axis would conflate the regime's qualitative novelty with its quantitative magnitude.

B.3 Episode and Workspace Configuration

All environments share an identical scaffolding: a 6-DoF xArm6 with a gripper mounted at a fixed table-side base pose, a planar tabletop with the target object initialized within a square reachability region centered at the robot's neutral grasp height, and two RGB cameras (a static overhead view and a wrist-mounted view) streaming at 20 Hz. The observation passed to the policy at each tick is the pair $(I_t^{\text{base}}, I_t^{\text{wrist}})$ together with a 7-D proprioceptive state (end-effector position, axis-angle orientation, gripper width). Every episode is capped at 200 environment steps (10 s wall-clock) and is judged successful only if the gripper both contacts the target and lifts it by at least 3 cm above its resting height before the cap. Lift-only and approach-only events are explicitly counted as failures, removing any pseudo-success that could arise from a policy that brushes against the object without committing to a grasp. Beyond the motion-regime parameters, episode-level randomization covers only the object identity, the in-plane initial position, and the motion seed; the table, lighting, and camera intrinsics are fixed across all 10,000 episodes, so any cross-environment gap reflects motion regime alone.

B.4 Demonstration Generation

Every episode is collected by an oracle motion-planner solution that has full access to the simulator state, including the object's true pose and velocity at every tick. The planner emits a smooth end-effector trajectory at the control rate of 20 Hz, with each waypoint expressed as a 6-D end-effector delta (XYZ translation plus axis-angle rotation) plus a 1-D gripper command, exactly the action space exposed to the learned policy. We discard any episode whose oracle rollout fails the lift criterion in the first attempt, ensuring that the released dataset contains only successful demonstrations and that any failure observed during evaluation is attributable to the policy rather than to an unsolvable initial condition. The full 1,000-episode pool per environment supports downstream finetuning, but the main benchmark evaluates each method on 100 held-out seed indices outside this pool, drawn from a deterministic seed offset to compare identical trial realisations.

B.5 Diagnostic Statistics

In addition to the per-environment success rate reported in the main text, we release for each policy run the per-episode trajectory of $(\alpha, K_{\text{exec}}, v_t, \hat{d}_t)$ when the policy is wrapped by PPC, the chunk-boundary timestamps and re-inference cadence, and the minimum gripper–object distance attained within the 200-step budget. These finer-grained signals do not enter the headline numbers but support the analysis in Appendix E; in particular the empirical α distribution drives our worst-case inference cost discussion, and the per-object success breakdown verifies that the gain from PPC is not concentrated on a single target geometry. All statistics will be released with the benchmark code.

C Supplementary Experiments and Analysis

C.1 Cross-Backbone Comparison of Inference-Time Wrappers

In the main results (Table 1), the comparison wrappers ACT (Zhao et al., 2023) and BID (Liu et al., 2025c) are evaluated on the strongest foundational backbone $\pi_{0.5}$ for fairness. We supplement this with a parallel evaluation on a second backbone, SmolVLA (Shukor et al., 2025), to verify that PPC's advantage over chunk-boundary smoothing approaches generalizes across backbones rather than being specific to a single VLA. ACT applies temporal ensembling over overlapping chunks, and BID applies guided rejection sampling at chunk boundaries; both operate without an external dynamics

signal. Following the same protocol as Table 1 (100 trials per environment, default deployment configurations), we report per-environment success rate in Table 4.

Table 4: **Inference-time wrappers on SmolVLA.** Dynamic-only success changes by -7.0, -0.8, and $+16.4$ points with ACT, BID, and PPC, respectively. $*$: PPC defaults to the baseline at $v = 0$.

Method	Static	Uniform Motion			Accelerated Motion			Irregular Motion			Average	
		Moving			Accelerating			Rand. Walk	Stop & Go	Teleport	Dyn. Only	All
		Easy	Med.	Hard	Easy	Med.	Hard					
SmolVLA (foundational)	81	76	57	27	41	33	13	53	40	44	42.7	46.5
SmolVLA + ACT (Zhao et al., 2023)	73 (−8%)	60 (−16%)	49 (−8%)	14 (−13%)	52 (+11%)	43 (+10%)	19 (+6%)	43 (−10%)	36 (−4%)	5 (−39%)	35.7 (−7.0%)	39.4 (−7.1%)
SmolVLA + BID (Liu et al., 2025c)	77 (−4%)	73 (−3%)	57 (±0%)	18 (−9%)	50 (+9%)	25 (−8%)	18 (+5%)	50 (−3%)	38 (−2%)	48 (+4%)	41.9 (−0.8%)	45.4 (−1.1%)
SmolVLA + PPC (ours)	81*	69 (−7%)	69 (+12%)	58 (+31%)	58 (+17%)	59 (+26%)	35 (+22%)	60 (+7%)	71 (+31%)	53 (+9%)	**59.1** (+16.4%)	**61.3** (+14.8%)

C.2 Additional Analysis on Experimental Results

Per-backbone gain decomposition. Among the four foundational VLAs in Table 1, PPC's largest absolute gain is on GR00T N1.6 ($+28.8$ dynamic-only), followed by π_0 ($+21.1$), $\pi_{0.5}$ ($+18.6$), and SmolVLA ($+16.4$). This ordering inversely correlates with the backbone's baseline dynamic performance: weaker dynamic baselines leave more room for the closed-form correction to recover. Notably, GR00T N1.6 has the strongest static score (88%) but the weakest dynamic score (37.3%), exhibiting the most severe dynamics blindness and consequently benefiting the most from PPC.

ACT's teleport collapse. ACT on $\pi_{0.5}$ scores 1% on Teleport, a 59-point drop from the bare backbone (60%). Temporal ensembling maintains a sliding buffer of overlapping chunks and averages them into a single action. When the object teleports, old chunks in the buffer still point toward the pre-teleport position, and averaging actively drags the end-effector away from the new target. The longer the buffer, the longer the stale signal persists. BID avoids this collapse (48%) because rejection sampling discards rather than averages stale chunks. PPC bypasses the issue entirely: the velocity signal detects the teleport via the ν_t sim-consistency gate, and the wrapper defaults to the unmodified baseline until the next valid reading.

Static performance. DynamicVLA scores 70% on the static environment, compared with 81% for SmolVLA. PPC preserves each backbone's static result through its bypass, with $\alpha^\star = 1$ and $\delta_k^\star = 0$ at zero motion input.

Irregular motion: regime-specific behavior. Stop & Go yields the largest PPC gain for GR00T, SmolVLA, and π_0 ($+19$, $+31$, and $+24$ points), while Random Walk yields the largest gain for $\pi_{0.5}$ ($+20$ points). Direction shifts and motion–pause transitions engage the latch-regulated cadence, complementing within-chunk correction. Teleport yields the smallest gain for each backbone, ranging from -7 to $+9$ points: instantaneous relocation leaves no continuous trajectory to track during the discontinuity. Thus, the positive irregular-family averages coexist with limited gains on teleportation.

Failure mode: SmolVLA+PPC on Uniform-Easy (-7%). SmolVLA already reaches 76% on the easiest uniform tier, where object speeds (1–2 cm/s) are close to the velocity estimator's noise floor. At these speeds, the cosine-projected $\alpha^\star$ oscillates near 1.0 with sign noise, occasionally triggering unnecessary compression that shortens the execution window without meaningful tracking benefit. This case marks PPC's lower effective operating boundary rather than a systematic failure.

D Supplementary Experimental Details

This appendix expands the implementation details summarized in Section 4. All numerical values were extracted directly from the canonical scripts cited per subsection.

Hardware. All foundational VLAs are fine-tuned on a single NVIDIA H200 (141 GB HBM3e). Original simulation evaluations (foundational baselines, comparison wrappers, PPC, and ablations) use one NVIDIA RTX A6000 (48 GB GDDR6). Each backbone requires at most 48 H200-hours of fine-tuning; evaluation per method (1,000 trials across 10 environments) takes about 2 A6000-hours.

Foundational VLA Fine-tuning. Each foundational backbone is fine-tuned from its publicly released checkpoint on the MoveBench demonstration set following the official recipe of the corresponding policy. Table 5 summarizes the per-backbone configuration. All backbones share the same dataset

(∼10K demonstrations across the ten environments), the same 7-D action space (6-D arm delta in physical space + 1-D gripper command), and the same control loop (20 Hz, `pd_ee_delta_pose` mode, image resolution and augmentations as in each backbone's released training config). The action chunk length T_{policy} varies by backbone: GR00T N1.5/N1.6 emits a 16-step chunk, while π_0, $\pi_{0.5}$, and DynamicVLA emit 20-step chunks; SmolVLA and Diffusion Policy use their default chunk size from the released configs. The wrapper retains the action representation and operates on the configured chunk window; the $T = 16$ window and its relation to native chunk lengths are specified below.

Table 5: **Per-backbone fine-tuning configuration on MOVEBENCH.** BS = batch size; T_{policy} = native chunk length emitted by the policy. All backbones except Diffusion Policy initialize from the official released pretrained checkpoint; optimizer, learning rate, and augmentation schedules follow each backbone's officially released training recipe and are not modified.

Backbone	Steps	BS	T_{policy}
GR00T N1.5/N1.6	20,000	32	16
π_0	30,000	32	20
$\pi_{0.5}$	30,000	32	20
SmolVLA	30,000	32	50
DynamicVLA	30,000	32	20
Diffusion Policy	30,000	64	16

Comparison Baseline Deployment. The comparison wrappers ACT and BID are deployed on top of the released $\pi_{0.5}$ checkpoint (and additionally on SmolVLA in Appendix C) using their original parameters, with one consistency adjustment for fair reactivity: *the execution horizon of every chunk-boundary baseline is capped at 10 env-steps*. Concretely, if a baseline's default execution horizon exceeds 10 it is set to 10, and if its default is below 10 we keep the default. This holds the inter-chunk re-observation latency comparable across methods and prevents any baseline from being penalized by an artificially long open-loop window. The full per-method configurations follow.

- **ACT (temporal ensembling)** on $\pi_{0.5}$: `decay_m=0.01`, `buffer_size=20`, `exec_horizon=1` (ensemble emits one action per env-step, well below the 10-step cap).
- **BID (guided rejection sampling)** on $\pi_{0.5}$: `n_samples=4`, `exec_horizon=10`, `overlap_window=4`.
- **DynamicVLA** uses its released 0.4B SmolVLA-based checkpoint fine-tuned on MOVEBENCH. Under the shared synchronous simulation protocol, it re-infers every 2 env-steps, matching PPC's minimum execution window. This setting controls inference-delay effects and achieves 44.9% overall success, compared with 31.9% for the conventional 10-step setting (51/46/44/21/38/37/12/37/28/5 across Static/Mov. E/M/H/Acc. E/M/H/RW/SnG/Tele). We therefore report the better-performing two-step setting.
- **VLASH** uses the $\pi_{0.5}$ backbone fine-tuned through its official pipeline on the same MOVEBENCH demonstrations. It follows the same synchronous simulation protocol as the other methods, with zero added inference delay to control latency effects when evaluating motion handling.

PPC Configuration. The following wrapper configuration is fixed across the original simulation runs; the real-robot configuration is specified in Appendix G.4.

- **Chunk geometry**: $T = 16$, $K = 2$ (K_{exec} floor), $H_{\text{eff}} = 10$ (K_{exec} ceiling). For backbones whose native chunk length T_{policy} exceeds 16, only the first 16 model-steps enter the wrapper; remaining model-steps are unused, matching the de-facto execution length set on every comparison baseline (≤ 10 env-steps consumed per chunk).
- **Latch** (single free knob): $\beta_{\text{in}} = 0.3$. The remaining latch constants $\beta_{\text{out}}, R_{\text{TH}}, L_{\text{th}}$ are derived from $(\beta_{\text{in}}, K, T)$ per Section 3.
- **Bayesian** α: $Q = 1.8$, $\beta_{\text{revert}} = 0.0$ (random-walk prior, no drift). Under the oracle velocity signal used in simulation, the perception-noise envelope is set to a large sentinel

value, forcing $K_{\text{kal}} \to 1$ and bypassing Bayesian shrinkage in the original oracle-signal experiments. The non-oracle simulation experiments and real-robot deployment retain the same sentinel; the hardware tracker gates immature estimates instead of supplying a confidence weight (Appendix G.4).

- **Velocity bound**: $V_{\max} = 1$ m/s, matching the xArm6 published maximum TCP velocity. See Appendix E for the robustness check.
- **Grasp gate**: $\|p_{\text{tcp}} - p_{\text{obj}}\| < 30$ mm (the xArm6 finger half-span); within this radius the wrapper bypasses as the object becomes internal state per the contact predicate in Section 3.1.
- **Workspace**: a world-anchored $80 \times 80 \times 30$ cm box, $[-0.4, 0.4] \times [-0.4, 0.4] \times [0.0, 0.3]$ m.
- **Lift threshold**: 30 mm above the object's initial height (paper success criterion).

Three implementation specifics warrant explicit mention:

- Δp *source*. The representative per-step motion estimate used by the correction is computed from the realized TCP trajectory rather than from raw chunk outputs. Specifically, $\Delta p_t = (p_t^{\text{tcp}} - p_{t-K}^{\text{tcp}})/K$ at every chunk-reset. The first chunk of each episode lacks K prior TCP samples and falls back to $\Delta p_0 = \frac{1}{K}\sum_{i=0}^{K-1} \text{chunk}[i, :3] \cdot c_{\text{pd}}$, where $c_{\text{pd}} = 0.04$ is the simulator's controller response factor that maps action-space delta to world-space delta.
- *Oracle velocity source*. The dynamics signal $\hat{vd}$ is obtained from the one-step finite difference of the simulator's ground-truth object position, $\hat{vd} = \text{cube}_t - \text{cube}_{t-1}$, capped at $V_{\max}$. The simulator's reported velocity field is used *only* as the input to the ν_t sim-consistency gate (Section 3) because that field reads bit-zero on kinematic teleports; the position finite difference is the actual signal driving $\alpha^\star$ and δ_k. For the non-oracle experiments this signal is replaced by the RGB-D mask-centroid and Kalman tracking pipeline described in Appendix G.3; CoTracker3 is evaluated as a separate online tracking baseline.
- *Channels modified*. The wrapper modifies only each chunk action's xyz translation, retaining the 3-D rotation deltas and 1-D gripper command from the corresponding chunk window. The correction therefore retains the policy's rotation and gripper values within that window.

Evaluation Protocol. For every method (foundational, comparison wrapper, PPC, ablation) we run 100 trials per environment across the 10 MOVEBENCH environments, totaling 1,000 trials per method. Trial seeds are $\{0, \ldots, 99\}$ and are disjoint from the demonstration pool. Each trial runs for at most 200 env-steps (10 s of simulator time at 20 Hz). Success is defined as a successful grasp followed by lifting the object by at least 30 mm above its initial height (`FORMAL_LIFT_THRESH` $= 0.03$ m). Reported success rates are point estimates; at $50/100$ successes, the 95% Clopper–Pearson interval is approximately $[39.8, 60.2]\%$, with a half-width of 10.2 pp.

E SUPPLEMENTARY ABLATION STUDIES

This appendix expands the ablations in Section 4.4 with computational overhead, adaptive engagement statistics of $\alpha^\star$, sim-consistency gate activation, velocity-bound robustness, detailed component ablations, and synthetic perception-noise experiments. Unless otherwise noted, evaluations use 100 rollouts per environment on MOVEBENCH.

Wrapper Computational Overhead. We instrument `compute_unified_correction` on MOVEBENCH-MovingMedium for 5 episodes (42 chunk-resets) and report per-call latency in Table 6. The closed-form operator runs in 0.07 ms (mean) on a single CPU thread, with $P99$ latency under 0.12 ms. Against typical GR00T N1.6 chunk inference of ~ 64 ms on an L40 GPU, the wrapper adds $< 0.2\%$ overhead, confirming real-time suitability within the inference budget.

Adaptive $\alpha^\star$ Engagement. Here $\alpha^\star$ denotes the execution compression recorded in the reported simulation configuration, bounded by the chunk-budget cap $T/K = 8$, rather than the unbounded analytical pace proposal. It reduces to 1 when no environmental motion is present. Table 7 reports the per-env distribution of $\alpha^\star$ across all chunk-resets on MOVEBENCH. The distribution is heavy-tailed and regime-adaptive: median $\alpha^\star$ remains near 1 across all environments while $P90$ scales monotonically with difficulty, reaching the saturation cap of 8 on AccelHard. Compression engages aggressively only when needed: $\alpha^\star \geq 2$ in 30.2% of AccelHard chunks (vs 5% on Moving) and

Table 6: **Per-chunk PPC computational overhead.** Single-thread CPU latency of `compute_unified_correction` on MOVEBENCH-MovingMedium ($n = 42$ chunk-resets).

Statistic	Latency (ms)
Mean	0.069
Median	0.080
P90	0.090
P99	0.115
Max	0.115

saturates at $\alpha^\star = 8$ in 10.5% of AccelHard chunks. On Teleporting, $\alpha^\star \equiv 1$ throughout because the ν_t gate clamps the wrapper to identity on A_1-violating chunks.

Table 7: **Per-environment distribution of $\alpha^\star$.** Median $\alpha^\star$ stays near 1 across all regimes; the upper tail scales with difficulty and saturates at $T/K = 8$ in 10.5% of AccelHard chunks.

Environment	Chunks	Median	P90	Max	$\alpha^\star \geq 2$	$\alpha^\star = 8$
Teleporting	1166	1.00	1.00	1.00	0.0%	0.0%
Moving Easy	1014	1.00	1.64	8.00	4.7%	0.3%
Moving Medium	987	1.01	1.60	8.00	5.2%	0.5%
Moving Hard	1312	1.09	2.04	8.00	11.3%	1.0%
Random Walk	1846	1.00	1.74	8.00	5.5%	0.2%
Stop and Go	1547	1.00	1.67	8.00	5.2%	0.3%
Accelerating Easy	1510	1.06	3.05	8.00	19.1%	3.6%
Accelerating Med.	1822	1.11	4.86	8.00	27.4%	7.5%
Accelerating Hard	2311	1.09	**8.00**	8.00	**30.2%**	**10.5%**

Sim-Consistency Bypass Rate. The ν_t gate detects A_1 violations from sim-internal contradiction: simulator-reported velocity strictly zero while observed displacement is non-zero. Table 8 reports per-environment activation. The gate fires on 82.7% of Teleporting chunks and 0% in other regimes, exactly matching its regime-specific discrete-bypass design. The remaining 17.3% occur between teleports, in stationary intervals with zero reported velocity and observed displacement.

Table 8: **Sim-consistency gate ν_t.** Active only on Teleporting; dormant under continuous dynamics.

Environment	$\nu_t = 0$ rate
Teleporting	**82.7%**
Random Walk	0.0%
Stop and Go	0.0%
Moving (Easy/Med./Hard)	0.0%
Accel. (Easy/Med./Hard)	0.0%

$V_{\max}$ Robustness. The velocity bound $V_{\max} = 1$ m/s corresponds to xArm6's official maximum TCP velocity. Doubling $V_{\max}$ to 2 m/s and re-running AccelHard drops success rate from 33% to 27% (Table 9), confirming that exceeding the hardware ceiling makes the wrapper overshoot the physical envelope. Thus, $V_{\max}$ is a physical specification rather than a hyperparameter.

E.1 ADDITIONAL COMPONENT ABLATION DETAILS

EMA Stabilizer Ablation. Disabling the latch costs 6.1 points overall (Uniform $74.0_{\downarrow 2.7}$, Accel $45.3_{\downarrow 7.7}$, Irregular $60.7_{\downarrow 8.0}$, Avg $60.0_{\downarrow 6.1}$), with the loss concentrated on irregular regimes and minimal on uniform motion. This asymmetry matches the latch's role as a regime-instability detector, activating only when the closed-form's quasi-stationarity assumption breaks down.

2nd-Order Channel Ablation. Removing the Lucas branch costs 3.4 points overall (Uniform $76.3_{\downarrow 0.4}$, Accel $45.7_{\downarrow 7.3}$, Irregular $66.0_{\downarrow 2.7}$, Avg $62.7_{\downarrow 3.4}$), with losses concentrated on accelerated motion and negligible on uniform regimes, matching the Lucas branch's 2nd-order correction role.

Table 9: $V_{\max}$ **robustness.** Exceeding the hardware specification degrades performance, confirming $V_{\max}$ is a physical constraint.

$V_{\max}$	AccelHard SR
1 m/s (xArm6 spec)	**33%**
2 m/s (beyond spec)	27 (−6%)

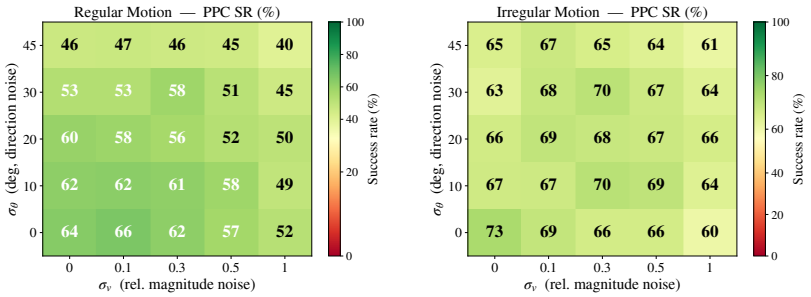


Figure 8: **Robustness to perception noise.** Success rate (%) under varying magnitude noise σ_v and directional noise σ_θ on the velocity signal. PPC exceeds the bare baseline under all conditions.

β_{outer} calibration. As illustrated in Fig.5 (a), sweeping β_{out} on irregular regimes (rand. walk and stop & go) yields a peak success rate of 68.5% at $\beta_{\text{out}} \approx 0.08$, which closely matches the theoretical value $1 - 2^{-K/T} \approx 0.083$ derived in Section 3. This agreement supports the derived outer-rate calibration. The latch construction computes the outer rate and threshold from chunk geometry and β_{in}, leaving the latter as its single free parameter.

Dynamic vs. fix α. As illustrated in Fig.5 (b), the dynamic α derived from the closed-form cost reaches 66.1% overall, surpassing every tested fixed setting. Among the static alternatives, $\alpha = 4$ and $\alpha = 6$ peak at around 58% and degrade on either side. The drop at $\alpha = 8$ shows that stronger fixed compression does not consistently improve success. The adaptive rule instead adjusts compression to each chunk's motion geometry.

E.2 Robustness to Perception Noise

Robustness to perception noise. In Fig. 8, we inject synthetic magnitude noise σ_v and directional noise σ_θ to simulate the error of real-world tracking hardware. Across all noise conditions, averaged over the six regular-motion environments and the three irregular-motion environments, PPC never falls below the corresponding bare-backbone baseline. Under noise levels typical of depth-camera and visual-tracker pipelines ($\sigma_v \leq 0.3$, $\sigma_\theta \leq 20°$), the correction operator retains the majority of its oracle-signal gains, confirming sufficient robustness for potential physical deployment.

F Qualitative Visualization

Trajectory comparison: baseline VLA vs. PPC. Figure 9 overlays end-effector (TCP) trajectories from the bare baseline and PPC-equipped runs on identical seeds, alongside the corresponding object trajectory, across four representative dynamic environments. In every panel, the baseline TCP veers into the static target's expected position and stops short of the moving object, terminating without grasp at distances of 30–100 mm. The PPC-equipped TCP, sharing the same backbone and seed, instead curves toward the object's current position at grasp closure, demonstrating active redirection of the executed path within the chunk window without changing the underlying policy.

Adaptive $\alpha^\star$ engagement across motion families. Figure 10 plots per-chunk $\alpha^\star$ and the disturbance magnitudes ($\|v\|$, $\|A^\star\|$) for one successful episode per family (uniform/accelerated/irregular). The three regimes produce qualitatively distinct wrapper signatures. On uniform motion, the observed velocity is small and roughly constant, producing $\alpha^\star \approx 1$ throughout with occasional mild excursions.

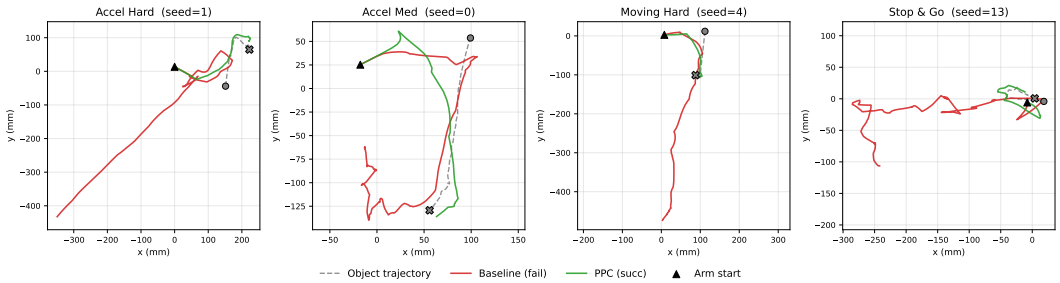


Figure 9: **Top-down (x–y) end-effector trajectories on identical seeds.** Gray dashed: object trajectory (● start, × end). Red: bare baseline TCP (terminates without grasp). Green: PPC-equipped TCP (terminates at grasp). Black triangle: arm start. PPC redirects the chunk-interior path to track the moving target across all four motion regimes.

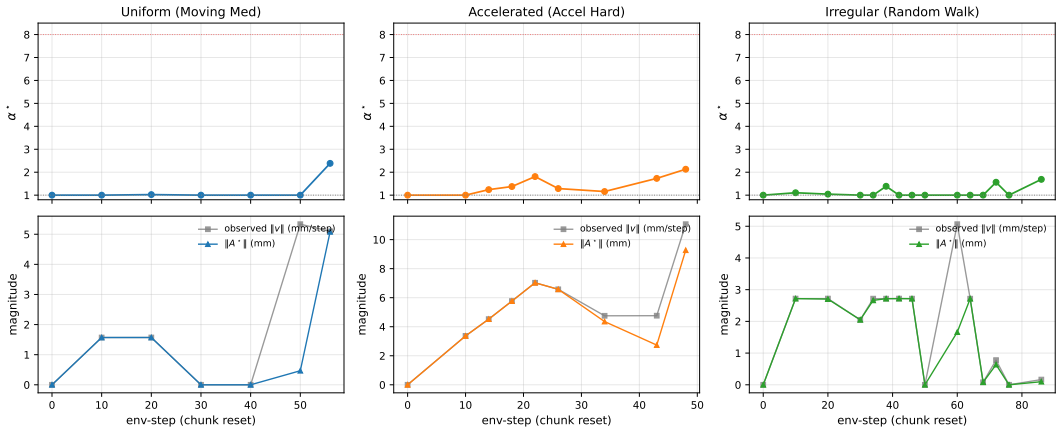


Figure 10: **Wrapper internals across motion families.** Top row: $\alpha^\star$ per chunk-reset; gray dotted line marks $\alpha = 1$ (no compression), red dotted line marks the chunk-budget cap $T/K = 8$. Bottom row: observed velocity $\|v\|$ (gray) and disturbance magnitude $\|A^\star\|$ (colored) per chunk. The three regimes yield distinct $\alpha^\star$ profiles: flat near 1 under uniform motion, monotonically rising under acceleration, and transiently spiking under irregular motion.

On accelerated motion, $\alpha^\star$ rises monotonically with the cube's accumulating velocity, saturating at the chunk-budget cap $T/K = 8$ in the late chunks where compression matters most. On irregular motion (random walk), $\alpha^\star$ produces transient spikes correlated with direction-shift events, returning to baseline once the regime stabilizes. These traces concretize Section 3's claim that the closed-form $\alpha^\star$ is regime-adaptive: dormant when motion is mild, aggressive when motion is fast, and reactive (rather than chronically engaged) when motion is irregular.

G Additional Controls and Deployment Experiments

This section details the controls and deployment experiments accompanying Sections 4.4 and 4.5. The matched simulation controls use a separate evaluation run from the main benchmark in Table 1. Their pooled success rates differ slightly (e.g., 54.8 vs. 54.3 and 73.9 vs. 72.9 for $\pi_{0.5}$), within the reported run-to-run variation of about two percentage points. Unless otherwise specified, the simulation controls cover nine dynamic environments, with 100 trials each for $\pi_{0.5}$ and GR00T.

G.1 Equal-Access Velocity Baseline

The naive feedforward control shifts the commanded trajectory by the predicted target displacement, adding $v\hat{d}$ per executed step. Summed over the execution window, this is the full first-order target

Table 10: **Equal-access velocity control.** Dynamic pooled success rate (%), $n = 900$ per arm. PPC retains a $+7.2$ / $+11.1$ percentage-point advantage over naive feedforward with the same signal.

Arm	$\pi_{0.5}$ SR [95% CI]	GR00T SR [95% CI]
Bare backbone (no v)	54.8 [51.5, 58.1]	36.7 [33.5, 39.9]
Same v, naive feedforward	66.7 [63.5, 69.7]	54.6 [51.2, 57.8]
Same v, PPC	**73.9 [70.9, 76.7]**	**65.7 [62.5, 68.8]**

Table 11: **Equal-access controls by motion family.** SR (%); each family pools three environments ($n = 300$), and All pools nine ($n = 900$). All-result confidence intervals are in Table 10.

Backbone	Arm	Uniform	Accel.	Irregular	All
$\pi_{0.5}$	Bare (no v)	62.7	44.3	57.3	54.8
	Same v, feedforward	78.3	66.3	55.3	66.7
	Same v, PPC	79.7	71.0	71.0	73.9
GR00T	Bare (no v)	49.0	5.3	55.7	36.7
	Same v, feedforward	68.7	39.3	55.7	54.6
	Same v, PPC	74.3	51.7	71.0	65.7

extrapolation under a constant-velocity model. It therefore gives the baseline the same dynamics information available to PPC without PPC's structured pace/path correction.

On uniform and accelerated motion, the feedforward baseline is competitive because it implements a constant-velocity prediction. Its improvement is limited on the irregular family, where extrapolation can misfire as the motion model breaks. PPC's irregular-family margins over feedforward are 15.7 and 15.3 points on $\pi_{0.5}$ and GR00T, respectively. This separates the benefit of the external signal from the additional benefit of the correction structure.

G.2 Fixed Horizons and Correction-Deleted Schedule

Control (a) runs the bare backbone at fixed short horizons, including PPC's floor. Control (b) uses PPC's adaptive execution-horizon rule with correction zeroed at execution. The control therefore shares PPC's scheduling rule.

More frequent re-observation helps $\pi_{0.5}$ by up to 13.6 points but still leaves a 5.5-point gap to PPC. On GR00T, the two-step setting reduces success to 1.7%, consistent with truncation of late grasp commands given that 84% of closure commands lie in the chunk's back half. PPC's adaptive rule with correction deleted recovers at most a third of the gain on $\pi_{0.5}$ and none on GR00T. These results support correction beyond the tested cadence-only controls and show that horizon shortening alone is not uniformly effective across backbones.

G.3 Closed-Loop Perception and Tracker Error

The deployed-class tracker fuses RGB-D segmentation-mask centroids across cameras with a constant-velocity Kalman filter, densely estimating one designated target's metric velocity per tick. We evaluate it and CoTracker3 online in closed loop in all nine dynamic environments.

Table 12: **Execution-horizon controls.** Dynamic pooled SR (%).

Arm	$\pi_{0.5}$	GR00T
Bare, $K_{\text{exec}} = 10$	54.8	36.7
Bare, $K_{\text{exec}} = 5$	66.6	35.0
Bare, $K_{\text{exec}} = 2$ (PPC floor)	68.4	1.7
PPC schedule, correction deleted	61.0	35.1
Full PPC	73.9	65.7

Table 13: **Oracle-to-tracker gap.** SR (%), $n = 900$ per arm.

Arm	$\pi_{0.5}$ SR [95% CI]	GR00T SR [95% CI]
Bare backbone	54.8 [51.5, 58.1]	36.7 [33.5, 39.9]
PPC, CoTracker3 online	57.2 [53.9, 60.5]	38.2 [35.0, 41.5]
PPC, RGB-D mask + Kalman	68.7 [65.5, 71.7]	57.1 [53.8, 60.4]
PPC, oracle velocity	73.9 [70.9, 76.7]	65.7 [62.5, 68.8]

Table 14: **Perception stress grid.** SR changes (pp) from clean-oracle PPC on three environments.

Degradation	$\pi_{0.5}$ Δ	GR00T Δ
Tracker latency 2 / 4 env-steps (100 / 200 ms)	$+0.0/-6.0$	$+1.3/-4.0$
Tracking failure, 10% / 20% of chunks	$-1.7/-5.7$	$-1.0/-11.0$
Occlusion, 5 of every 20 env-steps	-2.3	-16.3
Extrinsic calibration error, $5^\circ/10^\circ$	$-2.3/-1.0$	$+2.7/-1.3$
Combined: latency 2 + failure 10% + calibration 5°	-4.0	-8.0

The oracle-to-RGB-D gap is 5.2 / 8.6 points, retaining approximately 73% / 70% of the oracle gain over bare. Its measured signal error is $\sigma_v = 0.21$, $\sigma_\theta = 10.9^\circ$. These values lie within the synthetic noise envelope studied in Appendix E.2. CoTracker3 barely exceeds bare in this configuration: eight-frame strides yield 2.5 Hz updates for a 20 Hz control loop, limiting temporal coverage and leaving stale estimates at many chunk boundaries.

Hardware tracker measurements. The physical deployment measures the same noise axes using 12 recorded RGB-D belt traverses. Appendix G.4.3 reports the six object-by-speed operating points, bootstrap confidence intervals, and coverage in Table 15, together with the deployed sensing pipeline.

Individual and combined perception degradation. A three-environment subset isolates tracker latency, intermittent failure, occlusion, and extrinsic calibration errors. Table 14 reports percentage-point changes from clean-oracle PPC, with pooled SR 68.0 / 56.7 on this subset versus bare 37.3 / 24.7.

Every degraded condition stays above its corresponding bare baseline on this subset. When the signal is lost or gated, PPC receives $v = 0$ and becomes the identity wrapper for the affected chunk, withholding motion correction. Noisy depth is exercised end to end through rendered depth, segmentation, centroids, and Kalman filtering rather than by an additional depth-noise injection; on hardware, RealSense depth noise contributes to the measured noise envelope. Target identity ambiguity is further tested by the moving-distractor experiment below.

G.4 Real-Robot Deployment

We evaluate whether PPC can extend a policy fine-tuned only on static demonstrations to dynamic grasping through the same closed-form correction family. This complements the simulation experiments, whose backbones receive dynamic demonstrations during fine-tuning. On hardware, PPC uses the full-compensation limit of the general objective and requires no additional policy training; the deployment combines real RGB-D sensing, asynchronous inference, and constrained robot actuation.

G.4.1 Platform and Backbone

Platform and task. A UFACTORY xArm6 with a parallel-jaw gripper picks objects from a conveyor aligned with the robot's y axis. Two Intel RealSense cameras provide aligned RGB-D at 30 fps with 480×480 crops: a fixed eye-to-hand view and a wrist-mounted eye-in-hand view. Control runs at 10 Hz and inference uses one RTX 4090. The objects are a plastic medicine bottle (94 mm tall) and a blue toy car (47 mm tall); the IR-dark car is the more difficult depth-tracking target. The instruction is "pick up the {medicine bottle, blue toy car}". Each regime includes 10 trials per object per arm.

Static-only fine-tuning. We fine-tune $\pi_{0.5}$ from its base checkpoint on 100 scripted static pick demonstrations (50 per object, $9{,}117$ frames at 10 Hz), with batch size 128 for $2{,}000$ steps. The

belt is off during collection and objects are placed at random belt positions. The policy consumes two camera crops and a 7-D joint state, and emits a 10-step action chunk with translation deltas in metres, rotation deltas in radians, and an absolute gripper command at 0.1 s per tick. Inference is asynchronous, with the next chunk requested two ticks before the current execution window ends. This provides a nominal two-tick inference lead; measured latency is reported below. The bare policy executes eight steps per chunk.

Execution and actuation limits. Translation commands are accumulated into Cartesian targets, converted by damped least-squares IK, and streamed as joint targets. Both arms use the same limits in every trial: per-tick joint step ≤ 0.30 rad, joint speed ≤ 2.9 rad/s, Cartesian command step ≤ 22 mm, and a tool floor 5 mm below the belt surface. Oversized Cartesian rows are scaled without changing direction. The step or speed limiter engages on 0.05% of PPC control cycles versus 0.2% for bare. Measured TCP speed at the 99th percentile is 0.24 m/s for PPC and 0.22 m/s for bare; maximum measured joint speed is 1.4 versus 1.9 rad/s. These measurements distinguish realized motion from command-step bounds and show that the gains do not require relaxed actuation limits.

G.4.2 PPC Configuration and the Full-Compensation Limit

Chunk geometry and stabilizer. The robot uses $T = 10$, execution floor $K = 2$, and ceiling $H_{\text{eff}} = 8$, equal to the bare execution horizon. With $\beta_{\text{in}} = 0.3$, the derived constants are $\beta_{\text{out}} = 1 - 2^{-K/T} \approx 0.129$, $L_{\text{th}} = R_{\text{TH}} = 0.147$, and latch cap $\lfloor T/4 \rfloor = 2$. The velocity bound $V_{\max} = 1$ m/s and 30 mm contact gate are unchanged. Velocity is converted to displacement per control tick. Hardware uses $\nu_t \equiv 1$ and retains the oracle-parity sentinel $K_{\text{kal}} \to 1$; the tracker gates immature estimates instead of supplying a confidence weight.

Path channel at $\lambda \to 0$. All reported simulation results use $\lambda = 1$, giving the Fibonacci profile. Hardware uses the $\lambda \to 0$ limit of the same objective (Appendix A.7). For the first-order interior solution,

$$\delta_k^\star \longrightarrow A^\star = v\hat{d} - (\alpha^\star - 1)\Delta p = v\hat{d}_\perp, \qquad k < K. \tag{31}$$

This is the full-compensation endpoint of the regularized solution family, retaining its pace solution and orthogonal decomposition. The effort term trades tracking accuracy against spatial correction magnitude. In simulation, the dynamically fine-tuned backbones already exhibit some pursuit, motivating a regularized correction. In the hardware pilot, the static-only fine-tuned backbone exhibited limited lateral pursuit: pre-closure TCP velocity was approximately -0.4 cm/s along the belt axis, compared with belt velocities of -1.5 to -2.2 cm/s. Persistent lag motivated selecting the full-compensation endpoint before the formal evaluation.

The real-robot deployment uses a uniform full-compensation offset at each executed step, while retaining the pace and latch mechanisms. It applies the uniform offset $A^\star$ to each executed row. Apart from the robot's chunk geometry and interfaces, the path profile is the correction-operator change from simulation: the pace computation, adaptive execution-horizon rule, and latch mechanism are retained. The PPC correction configuration is fixed across both objects and all formal moving-belt regimes. Thus, the two deployments instantiate one analytic family for different backbone responses without additional backbone training or per-regime correction tuning.

Comparison arms and static control. Bare and PPC share the backbone and execution stack. The bare arm sets $\alpha = 1$, $\delta_k = 0$, and $K_{\text{exec}} = 8$. In the static control, an explicit interlock pins PPC to identity; this checks performance with the wrapper present under its static bypass configuration.

G.4.3 Conveyor Dynamics Sensor

The perception chain receives neither commanded belt speed nor regime labels. An open-vocabulary Grounding-DINO detector seeds the target in the fixed RGB-D view, re-seeding once per second or upon loss. Depth points above the belt plane within the detected region yield a 3-D centroid for a constant-velocity Kalman filter. Velocity is projected onto the known belt axis. The served speed is a running median of clean-zone samples collected while the tool is at least 15 cm from the track, held through occlusion. Estimates with fewer than eight consecutive updates are withheld to suppress re-acquisition transients. The latch reads the raw per-chunk velocity stream. When no valid served estimate is available, PPC receives $v = 0$ and withholds motion correction for that chunk.

Table 15: **Measured hardware tracker noise.** Point estimates and bootstrap 95% CIs over belt traverses. Slow, medium, and fast calibration speeds are approximately 2.4, 3.6, and 5.7 cm/s. Simulation reference points are RGB-D mask + Kalman $(0.21, 10.9^\circ)$ and CoTracker3 $(0.39, 6.4^\circ)$.

Object	Speed	σ_v [95% CI]	σ_θ (°) [95% CI]	Coverage
Bottle	Slow	0.208 [0.168, 0.242]	10.0 [5.4, 11.2]	0.98
Bottle	Medium	0.145 [0.126, 0.165]	9.8 [9.3, 10.4]	0.98
Bottle	Fast	0.140 [0.132, 0.146]	7.8 [7.3, 8.1]	0.98
Toy car	Slow	0.287 [0.216, 0.347]	15.6 [11.9, 17.7]	0.99
Toy car	Medium	0.256 [0.199, 0.294]	14.4 [13.3, 15.0]	0.98
Toy car	Fast	0.120 [0.101, 0.135]	12.2 [10.5, 13.0]	0.97

For the speed-varying trials, the sensing pipeline additionally uses a change-detection reset of the median pool and admits near-zone wrist-camera velocity samples when the arm blocks the fixed view. The share of control cycles with a velocity sample rises from 16.5% with the base view to 79.4% with the wrist channel. This extends sensing coverage during terminal approach; the PPC correction parameters remain fixed.

Tracker calibration and noise. We replay 12 raw RGB-D belt-traverse recordings through the deployed binary, covering two objects and three speeds (15,414 frames in total). Error is scored against a per-traverse displacement self-reference using the same σ_v and σ_θ definitions as in simulation; intervals are obtained by bootstrapping belt traverses. The six point estimates fall within $\sigma_v \leq 0.3$ and $\sigma_\theta \leq 20^\circ$ (Table 15, Fig. 11b). The slow-car magnitude interval extends to 0.347. Free-flight magnitude ratios are 1.03–1.08, corresponding to a 3–8% positive offset relative to the reference. An arm-sweep null test gives a residual of $+0.07$ cm/s. Cycles withheld by the maturity gate count against coverage rather than entering the error estimate. The 0.97–0.99 replay coverage describes belt-traverse calibration and is distinct from the occluded closed-loop coverage above. These measurements locate hardware sensing on the simulation noise axes; end-to-end grasping trials directly establish performance of the deployed $\lambda \to 0$ instance.

G.4.4 Evaluation Protocol

Each arm has 20 trials in each of four regimes: static (belt off), constant slow (1–4 cm/s), constant fast (4–6 cm/s), and speed-varying (2–6 cm/s). In the last regime, the operator changes the analog belt-speed dial one or more times within the episode. The static and constant-speed comparisons are paired: bare and PPC run interleaved on the same condition sequence, bare first. Since the belt is analog-controlled, constant-speed bands are assigned from measured object velocity using a pre-registered 4.0 cm/s threshold; pairs falling into different bands are re-collected.

An episode ends upon success, at 25 s, when the object exits the working zone, or at a safety stop; the latter three are failures. Success requires gripper readback within the hold band, a lift of at least 3 cm sustained across consecutive ticks, and confirmation that the object is held from the camera or, when occluded by the gripper, the operator. We report 95% Clopper–Pearson intervals. All rows use two-sided Fisher's exact tests.

G.4.5 Results and Timing

Dynamic grasping. Table 3 reports PPC success of 16/20 (80%), 12/20 (60%), and 13/20 (65%) on constant-slow, constant-fast, and speed-varying belts, versus 0/20 for bare in each regime. Across moving belts, this is 41/60 (68.3%, 95% CI $[55.0, 79.7]\%$) versus 0/60 (0%, CI $[0, 6.0]\%$). All three moving-belt comparisons satisfy $p < 10^{-3}$. The static control remains high at 19/20 for bare and 20/20 for the identity-pinned wrapper. The moving-belt gains demonstrate dynamic grasping without dynamic fine-tuning of the deployed backbone.

Latency. Across 1,758 constant-speed chunk boundaries, capture-to-command latency, including tracking, policy inference, and dispatch, is 169 ms at the median and 342 ms at the 90th percentile; policy inference accounts for 114 ms at the median. For speed-varying trials, end-to-end median and 90th-percentile latency are 187 and 314 ms. Both medians fit the nominal 200 ms window

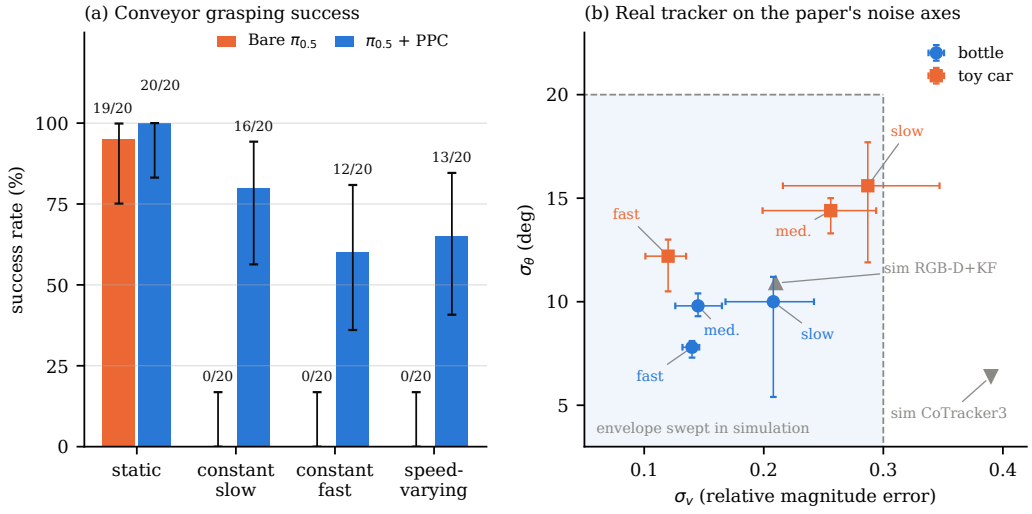


Figure 11: **Real-robot deployment.** (a) Success with 95% Clopper–Pearson intervals. PPC achieves 60–80% moving-belt success with a backbone fine-tuned only on static demonstrations. PPC is pinned to identity for the static control. (b) Hardware tracker point estimates and bootstrap 95% CIs on the simulation noise axes. All six point estimates lie in the shaded region; the slow-car magnitude CI extends beyond it.

Table 16: **Competing moving object.** SR (%) in the moving-distractor scene.

Arm	$\pi_{0.5}$ SR [95% CI]	GR00T SR [95% CI]
Bare backbone	18.0 [11.0, 26.9]	7.0 [2.9, 13.9]
PPC, non-oracle RGB-D tracker	42.0 [32.2, 52.3]	34.0 [24.8, 44.2]
PPC, oracle velocity	45.0 [35.0, 55.3]	49.0 [38.9, 59.2]

at the two-tick execution floor; the upper tail exceeds that window, and asynchronous execution accommodates variable inference timing.

Deployment significance. The hardware evaluation complements the simulation study along the backbone's training background as well as the sensing and execution pipeline. With the correction configuration fixed across formal moving-belt tests, the full-compensation instance enables a static-only fine-tuned policy to grasp under real depth noise, occlusion, and latency. The simulation ablations characterize the regularized correction structure, while these hardware trials establish the full-compensation deployment's effectiveness. Together they support one closed-form family across backbones with different dynamic responses.

G.5 Moving-Distractor Scene

A second, task-irrelevant object crosses the workspace to test whether perception tracks the designated target. Evaluation is zero-shot, without re-tuning, using both oracle and non-oracle signals.

The gains with both oracle and non-oracle signals show that PPC can compensate for a designated target's motion in the presence of another moving object.

G.6 Hardware-Clamped Correction

A hardware-clamped arm projects each corrected per-step delta onto a conservative actuation envelope with Cartesian-step and acceleration bounds before execution.

The small performance reduction shows that PPC's gains persist under conservative actuation limits. The projection is applied after computing the closed-form correction, retaining the inexpensive

Table 17: **Conservative actuation constraints.** Dynamic pooled SR (%). Clamping preserves gains of $+17.3$ / $+25.5$ points over bare.

Arm	$\pi_{0.5}$ SR [95% CI]	GR00T SR [95% CI]
PPC, unclamped	73.9 [70.9, 76.7]	65.7 [62.5, 68.8]
PPC, hardware-clamped	72.1 [69.1, 75.0]	62.2 [59.0, 65.4]
Change (percentage points)	-1.8	-3.5

analytic proposal while constraining the executed command. Commands then pass through the robot's existing IK, servo, and safety constraints.

H Limitations and Broader Impact

H.1 Limitations

Quasi-stationarity assumption. The closed-form derivation assumes the disturbance velocity and direction are approximately constant within each executed chunk (A2). While the hierarchical latch mitigates chronic violations by capping K_{exec}, the correction within any single chunk remains suboptimal when the disturbance changes rapidly during execution. Teleportation, the most extreme violation, relocates the object instantaneously and leaves no continuous trajectory for the operator to track, explaining the limited gains observed in that regime.

Hardware validation scope. Beyond ManiSkill/SAPIEN experiments, we validate PPC on a physical xArm6 conveyor rig with two RGB-D views and a non-oracle tracker. This establishes transfer for one hardware configuration, two objects, and three moving-belt regimes, rather than arbitrary scenes or robots. Depth noise, occlusion, tracker latency, and calibration errors remain deployment factors. The hardware results validate the $\lambda \to 0$ instance with a static-only fine-tuned backbone; simulation controls and ablations characterize the regularized $\lambda = 1$ instance. Appendices G.3 and G.4 report the corresponding perception tests and hardware measurements.

Task coverage. MoveBench isolates motion regime through a single task family (pick), and PPC tracks one designated target per chunk. The moving-distractor experiment validates target selection and correction with a competing moving object. Other target-relative primitives, such as pushing or placing into a moving receptacle, fit the correction interface but have not been evaluated; neither have sequential multi-object manipulation or multi-step assembly. Contact-rich tasks such as pouring that cannot be represented by a moving target point require a broader formulation.

External velocity signal. PPC decouples perception from correction by reading velocity from an external source rather than from the VLA backbone. This design sidesteps the ego-motion confound but introduces a dependency on a reliable tracking or depth-sensing pipeline. In cluttered or heavily occluded scenes where object tracking fails, PPC degenerates to the baseline VLA.

Robot feasibility. PPC is a high-level Cartesian action wrapper, not a whole-body controller. Our generality claim concerns chunked VLA backbones sharing this interface. Corrected commands must still satisfy the robot's IK, servo, and safety constraints. Conservative per-step Cartesian and acceleration clamps preserve most gains; all real-robot trials use a hard per-tick step clamp. This does not establish feasibility near joint or torque limits, singularities, or for underactuated and mobile morphologies; a robot-specific constrained QP or optimal-control formulation is complementary.

H.2 Broader Impact

This work improves the robustness of robot manipulation policies in dynamic environments without requiring retraining or additional data collection. The primary positive impact is enabling safer and more capable deployment of general-purpose robots in settings where objects or people move during task execution, such as manufacturing lines, household assistance, and human-robot collaboration.

Since PPC is a training-free wrapper with no learnable parameters, it does not introduce new data privacy or bias concerns beyond those of the underlying VLA.

On the negative side, more capable manipulation in dynamic settings could lower the barrier for autonomous systems operating near humans, where safety-critical failure modes require careful handling before deployment. Although PPC is evaluated in simulation and on a physical conveyor rig, these experiments do not constitute safety validation for real-world human-facing applications. Any deployment should include appropriate safety mechanisms independent of the policy layer.