GainPilot: Steering Vision-Language-Action Policies with Gain-Triggered Replanning and Bounded Action Correction

Anonymous authors
Paper under double-blind review

Abstract

Action-chunked vision-language-action (VLA) policies can accumulate execution errors, but detecting deviations alone does not determine whether replanning helps. We introduce GainPilot, which combines gain-triggered replanning with bounded post-sampling correction of a base policy. A dynamics-informed trigger predicts the terminal-success gain of replacing the active queue with a subsequently sampled and corrected plan. An anchored critic jointly learns absolute success, anchor-relative gains, and outcome differences between paired local perturbations. One action-gradient computation then supplies a coordinate-aware correction direction, bounded separately for translation and gripper commands. Across 13 backbone–horizon configurations on Meta-World MT50, spanning $\pi_{0.5}$, SmolVLA, and X-VLA, GainPilot improves success by 10.6–26.0 percentage points over the base policies. For $\pi_{0.5}$ at $H = 10$, success rises from 61.6% to 74.2%, compared with 69.4% for gain-triggered replanning alone. On LIBERO with 10-shot fine-tuning, mean success rises from 79.00% to 90.75%; across nine real-robot tasks, it increases from 53.3% to 73.9%.

1 Introduction

Vision-language-action (VLA) models control diverse manipulation tasks from visual observations and language instructions (Kim et al., 2025; Physical Intelligence et al., 2025). Action-chunked execution reduces policy-call frequency by executing a prefix before replanning (Chi et al., 2023), but longer prefixes delay feedback. An unfinished plan may therefore persist after interaction conditions change. We study when to replace such a plan and how to improve its replacement.

VLA-Corrector triggers corrective replanning from visual-dynamics discrepancies (Pan et al., 2026), while Bernoulli-Continuation Policy (BCP) learns chunk horizons from success and policy-call rewards (Xu et al., 2026). A deviation, however, does not establish whether intervention will help. We directly learn the terminal-success gain of a corrected replacement over continuing the queue. For correction, Guided Action Flow applies gradients within sampling (Yang et al., 2026), and Coda predicts a residual afterward (Tang et al., 2026). We instead learn local outcome differences around the sampled plan.

We introduce GainPilot, combining gain-triggered replanning with anchored, single-step post-sampling correction. Offline, independently trained dynamics and critic models are frozen to provide predicted effects and actual corrected branches for gain supervision. Online, the trigger combines observed history, predicted dynamics, horizon, and queue phase before replacement sampling. A horizon-conditioned critic compares prefixes around the new plan using absolute-success, anchor-relative gain, and paired local-difference supervision. One action gradient is converted through the action-coordinate metric and separate translation/gripper RMS and coordinate-wise bounds into a corrected prefix; execution then resumes normal planning and monitoring.

Across 13 Meta-World MT50 backbone–horizon configurations spanning $\pi_{0.5}$, SmolVLA, and X-VLA, GainPilot improves success by 10.6–26.0 percentage points over base policies. For $\pi_{0.5}$ at $H = 10$, success rises from 61.6% to 69.4% with replanning and 74.2% with GainPilot. Across nine real-robot tasks, average success increases from 53.3% to 73.9%.

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.

Our contributions are:

- **Gain-based selective replanning.** A dynamics-informed trigger predicts corrected-replacement gain before sampling, relative to preserving the queue under a common ordinary H-step controller and remaining budget.
- **Anchored local-difference supervision.** Absolute outcomes, anchor-relative gains, and paired perturbation differences supervise local action comparisons around a fixed, newly sampled plan.
- **Bounded single-step post-sampling correction.** Each intervention converts one critic-gradient computation into a coordinate-aware, magnitude-constrained action prefix, executes it, and resumes normal planning.

2 Related Work

2.1 Action-Chunked Visuomotor and VLA Policies

Action Chunking with Transformers (ACT; Zhao et al., 2023) learns action sequences and temporally ensembles overlapping predictions; Diffusion Policy (Chi et al., 2023) combines diffusion-based prediction with prefix execution. Octo (Ghosh et al., 2024) uses diffusion decoding for action chunks conditioned on language or goal images, while π_0 (Black et al., 2025a) combines a pretrained VLM with a flow-matching action expert. In language-conditioned manipulation, $\pi_{0.5}$ (Physical Intelligence et al., 2025), SmolVLA (Shukor et al., 2025), and X-VLA (Zheng et al., 2025) generate chunks from multimodal context. Real-Time Chunking (RTC; Black et al., 2025b) overlaps inference and execution, using guided inpainting to respect actions committed during inference delay. GainPilot studies selective interruption and replacement correction, complementing these action-generation and continuity mechanisms.

2.2 Execution Monitoring and Selective Replanning

Sentinel (Agia et al., 2025) combines temporal action-consistency statistics with vision-language assessments of progress. SAFE (Gu et al., 2025) learns failure scores from VLA features and calibrates alarms with functional conformal prediction. FAIL-Detect (Xu et al., 2025) calibrates sequential out-of-distribution alarms using successful rollouts. VLA-Corrector (Pan et al., 2026) uses persistent visual-prediction errors to trigger queue truncation and gradient-guided replanning. Adaptive-horizon methods include PACE (Nie et al., 2026), which selects low-speed valleys in smoothed motion profiles using demonstration-calibrated thresholds, and BCP (Xu et al., 2026), whose ordered continuation model learns from trajectory-level success and replanning-efficiency rewards. GainPilot instead learns the terminal-success difference between preserving the queue and executing a replacement corrected by the final frozen critic, from the same execution cut. It reevaluates this gain during execution, before sampling a replacement.

2.3 Critic-Guided and Post-Sampling Action Correction

Residual reinforcement learning (Johannink et al., 2019) adds a return-maximizing policy to a fixed, hand-designed controller. Diffusion-QL (Wang et al., 2023) combines behavior cloning and Q-value maximization during policy training, propagating value gradients through the diffusion chain. Diffuser (Janner et al., 2022) guides trajectory sampling with gradients from a learned return predictor. Guided Action Flow (Yang et al., 2026) trains an action-chunk critic with offline Implicit Q-Learning (Kostrikov et al., 2021) and uses critic gradients at estimated clean actions to guide flow velocities. Coda (Tang et al., 2026) predicts a demonstration-supervised endpoint residual from the sampled candidate, source noise, and observation features, reallocating part of the integration budget. GainPilot applies one bounded critic-gradient correction after sampling, when requested by its gain trigger. Absolute success, anchor-relative gains, and paired perturbation differences reuse terminal outcomes to supervise local comparisons, without action-residual targets or temporal-difference backups.

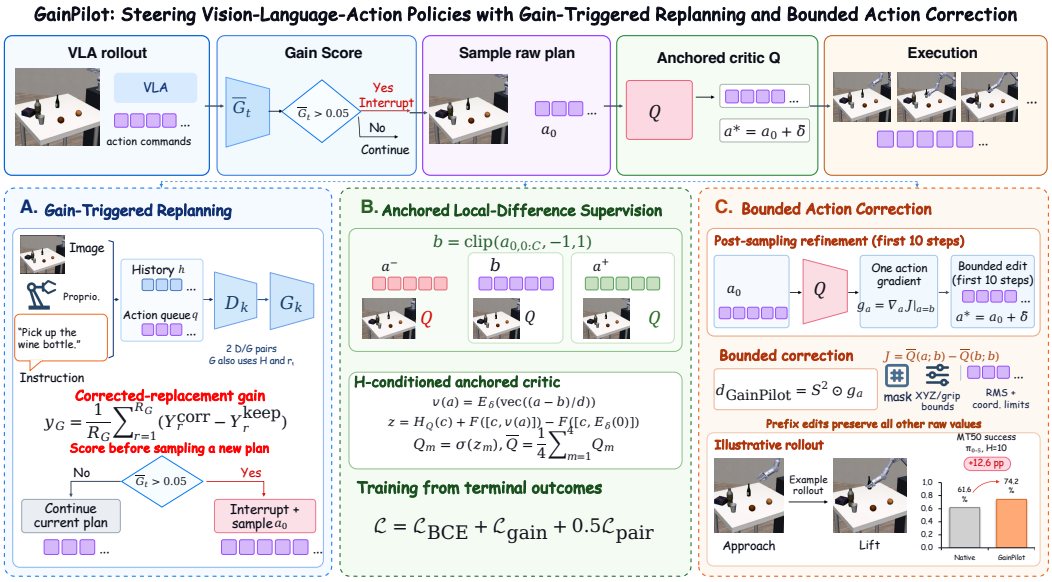

Figure 1: **GainPilot's execution loop. (A)** Dynamics-informed gain scoring precedes replacement sampling. **(B)** The new plan anchors a critic trained on absolute success and local outcome differences. **(C)** One coordinate-scaled action gradient yields a bounded prefix; execution then resumes normal planning. Scene images use external visualization views.

3 Method

GainPilot selects interventions before replacement sampling, then corrects the sampled prefix using an anchored critic (Figure 1). Offline, D and Q train independently and freeze before G learns the resulting correction operation's terminal-success gain.

3.1 Problem Setup

The policy π generates a raw K-step chunk $A^{\text{raw}} \in \mathbb{R}^{K \times d_a}$. Ordinary execution uses its first H actions, discards the tail, and replans. At control step t, the decision context is $x_t = (h_t, q_t, H, r_t)$: h_t contains visual/proprioceptive history, executed commands, the instruction, and remaining-budget fraction $(T-t)/T$; q_t is the unexecuted committed queue, with r_t steps remaining. An intervention executes a corrected replacement prefix for $C = 10$ steps without further intervention, then resumes ordinary planning under the original budget T. The reference controller μ_H preserves the original queue and natural replanning clock on a keep branch; after a candidate's C steps, it discards the unused tail and samples a fresh H-step commitment.

3.2 Gain-Triggered Replanning

Two dynamics models D_k predict visual changes $\Delta\widehat{v}_t^{(k)}$ from the next $\ell = 5$ queued commands. Paired gain models combine these predictions with history and fixed-scale normalized controller inputs $\widetilde{H}, \widetilde{r}_t$:

$$\overline{G}_t = \frac{1}{2}\sum_{k=1}^{2} G_k\left(h_t,\, q_{t,0:\ell},\, \Delta\widehat{v}_t^{(k)},\, \widetilde{H},\, \widetilde{r}_t\right). \tag{1}$$

Let $q_H(x, a)$ be terminal success probability after candidate a executes for C steps followed by μ_H, and $V_{\mu_H}(x)$ that of retaining the queue. The frozen final critic defines $\mathcal{T}_Q(x, A_0)$, the corrected raw C-step prefix of a fresh plan. The replacement advantage and its empirical label are

$$A_H(x) = \mathbb{E}_{A_0 \sim \pi(\cdot|\text{obs}(x))}\left[q_H(x, \mathcal{T}_Q(x, A_0))\right] - V_{\mu_H}(x),$$
$$y_G = \frac{1}{R_G}\sum_{r=1}^{R_G}\left(Y_r^{\text{corr}} - Y_r^{\text{keep}}\right). \tag{2}$$

The Y values are actual terminal outcomes under the original budget. Paired repetitions average over replacement and continuation randomness. Each G_k regresses y_G by mean squared error using frozen D_k predictions, and outputs a real-valued score before replacement sampling. At eligible ℓ-step monitoring times, $\overline{G}_t > 0.05$ triggers intervention; natural replanning takes precedence (Algorithm 1, Appendix C).

3.3 Anchored Critic with Local-Difference Supervision

A newly sampled replacement A_0^{raw} supplies the bounded anchor

$$b = \text{clip}\left(A_{0,0:C}^{\text{raw}}, -1, 1\right). \tag{3}$$

Each critic learns q_H from candidate terminal outcomes. Its state feature f_m combines causal history and $\widetilde{H}$. Omitting the member index, state, anchor, and difference encoders E_s, E_a, E_δ define

$$\begin{aligned} c &= [E_s(f),\ E_a(\text{vec}\, b)], \\ v(a) &= E_\delta\left(\text{vec}\,\frac{a-b}{d}\right), \\ z(f,a;b) &= H_Q(c) + F([c, v(a)]) - F([c, E_\delta(0)]), \\ Q(f,a;b) &= \sigma(z(f,a;b)). \end{aligned} \tag{4}$$

Here d contains fixed critic-input scales. At $a = b$, relative terms cancel, giving $z(f, b; b) = H_Q(c)$. Four members contribute their mean sigmoid probability.

Each training source i contains an anchor ($j = 0$) and local candidates sharing H and paired continuation seeds. For binary terminal outcomes y_{ijr}, let $\widehat{p}_{ij} = R^{-1}\sum_r y_{ijr}$ and $p_{ij} = \sigma(z_{ij})$. The same outcomes supervise three source-wise losses:

$$\begin{aligned} \mathcal{L}_{\text{BCE},i} &= \text{Avg}_{w_i^{\text{BCE}}}\left[\text{BCEWithLogits}\left(z_{ij}, \widehat{p}_{ij}\right)\right], \\ \mathcal{L}_{\text{gain},i} &= \text{Avg}_{w_i^{\text{gain}}}\left[\left((p_{ij} - p_{i0}) - (\widehat{p}_{ij} - \widehat{p}_{i0})\right)^2\right], \\ \mathcal{L}_{\text{pair},i} &= \text{Avg}_{w_i^{\text{pair}}}\left[\left((p_{ij} - p_{ik}) - (\widehat{p}_{ij} - \widehat{p}_{ik})\right)^2\right]_{(j,k)\in\mathcal{P}}, \end{aligned} \tag{5}$$

Here $\mathcal{P}$ contains recorded positive/negative perturbation pairs, and Avg_w accounts for duplicate raw commands (Appendix B). The objective is

$$\mathcal{L} = \frac{1}{B}\sum_{i=1}^{B}\left(\mathcal{L}_{\text{BCE},i} + \mathcal{L}_{\text{gain},i} + 0.5\,\mathcal{L}_{\text{pair},i}\right). \tag{6}$$

BCE learns absolute success; the gain and pair terms learn anchor-relative and local outcome differences in probability space.

3.4 Single-Step Correction and Execution

With anchor, context, and horizon fixed, one candidate-action gradient gives

$$J(a;b) = \frac{1}{M}\sum_{m=1}^{M}\left[Q_m(f_m, a; b) - Q_m(f_m, b; b)\right], \qquad M = 4,$$
$$g_a = \nabla_a J(a;b)\big|_{a=b}, \qquad d_{\text{GainPilot}} = S^2 \odot g_a. \tag{7}$$

The affine denormalization scale S maps the direction into raw-command coordinates through S^2, separately from critic-input scales d. XYZ and gripper corrections each use their own boundary mask, RMS normalization, and feasible radius, enforcing both RMS and coordinate-wise limits. The radius depends on that group's direction and available command margin; full bounds and controller pseudocode appear in Appendix C.

GainPilot adds δ_j only to raw-prefix coordinates with nonzero correction, preserving all other commands. It executes the corrected prefix for C steps or until termination, discards the unused tail, and resumes ordinary H-step planning and monitoring. "Single-step" denotes one action-gradient computation per intervention; interventions may recur within an episode.

4 Experiments

4.1 Experimental Setup

We evaluate $\pi_{0.5}$, SmolVLA, and X-VLA on Meta-World MT50 (Yu et al., 2020), with 500 episodes per configuration (ten fixed initial conditions per task). All checkpoints use 10 MT50 training epochs. Within each backbone, methods and horizons share evaluation initial conditions. Table 1 covers $H = 10$–50 for $\pi_{0.5}$/SmolVLA and $H = 10$–30 for X-VLA; generated chunks contain 50/50/30 actions, respectively, and $C = 10$. We also evaluate LIBERO (Liu et al., 2023) and the AgileX PiPER robot.

Compared configurations and metrics. Native executes normal H-step prefixes. Replan uses GainPilot's trigger with raw replacements; GainPilot corrects them. Matched Replan trains a dedicated trigger on raw-replacement gain. Interventions repeat by default. GainPilot uses L11 and four critics; each backbone trains its auxiliary models separately from its own checkpoint's data with matched collection/training budgets. Each $D/G/Q$ member has approximately 10/3.23/3.78M parameters for $\pi_{0.5}$ (Appendices A–B). We report terminal success and mean calls/interventions over all episodes; a call generates a complete chunk.

4.2 Main Results and Baseline Comparisons

GainPilot improves over Native by 10.6–26.0 percentage points (pp) and Replan by 4.8–8.4 pp in all 13 configurations (Table 1). For $\pi_{0.5}$ at $H = 10$, success progresses from 61.6% to 69.4% to 74.2%. At $H = 50$, GainPilot retains 65.2% success versus Native's 39.2%; the gains extend to every backbone and horizon.

Inference cost. Replanning alone gains 7.8/17.6 pp over Native for $\pi_{0.5}$ at $H = 10/50$. Correction adds 4.8–8.4 pp across configurations while changing mean calls by only -0.310 to $+0.314$ per episode. At $H = 10$, its additional query latency is 15.067–20.484 ms across backbones (Appendix C). Figure 2 shows a coffee-pulling example.

Baseline comparisons. With the same $\pi_{0.5}$ checkpoint and 500 MT50 episodes at $H = 10$, GainPilot reaches 74.2%, versus 69.8% for VLA-Corrector, 68.0% for GAF, and 68.8% for Matched Replan (Table 2). Matched Replan shares dynamics features, gain architecture, training sources, and optimization budget, but supervises raw-replacement gain (Appendix F); GainPilot exceeds it by 5.4 pp.

4.3 Ablations and Intervention Analysis

Local-difference supervision. We retrain all critic parameters for four objectives, fixing architecture, data, and update budget and matching initialization and sampling streams. Three independent

Table 1: **MT50 across backbones and execution horizons.**

H	Success (%) / policy calls			GainPilot Int.	GainPilot gain (pp)	
	Native	Replan	**GainPilot**		vs. Native	vs. Replan
$\pi_{0.5}$						
10	61.6 / 15.130	69.4 / 14.452	**74.2** / 14.248	2.050	**+12.6**	**+4.8**
20	48.0 / 9.372	59.2 / 10.158	**67.2** / 9.968	2.390	**+19.2**	**+8.0**
30	41.0 / 6.964	58.4 / 8.650	**66.2** / 8.678	2.616	**+25.2**	**+7.8**
40	41.0 / 5.602	58.6 / 8.276	**64.8** / 7.966	2.594	**+23.8**	**+6.2**
50	39.2 / 4.312	56.8 / 7.584	**65.2** / 7.422	2.624	**+26.0**	**+8.4**
SmolVLA						
10	47.2 / 19.532	54.0 / 20.550	**60.8** / 20.382	2.700	**+13.6**	**+6.8**
20	45.2 / 9.996	56.2 / 13.328	**63.2** / 13.338	3.096	**+18.0**	**+7.0**
30	41.0 / 7.036	54.8 / 11.212	**61.6** / 11.098	3.194	**+20.6**	**+6.8**
40	38.2 / 5.780	50.4 / 10.302	**58.6** / 10.134	3.200	**+20.4**	**+8.2**
50	33.8 / 4.576	48.2 / 9.462	**55.6** / 9.280	3.172	**+21.8**	**+7.4**
X-VLA						
10	78.6 / 12.264	84.4 / 13.402	**89.2** / 13.412	2.884	**+10.6**	**+4.8**
20	75.6 / 6.612	82.8 / 10.524	**89.0** / 10.352	3.396	**+13.4**	**+6.2**
30	74.0 / 4.572	80.6 / 9.584	**87.0** / 9.898	3.848	**+13.0**	**+6.4**

Entries: success (%) / mean policy calls; Int.: mean interventions. Gains are percentage points (pp). Each row uses 500 episodes; Replan and GainPilot permit repeated interventions.

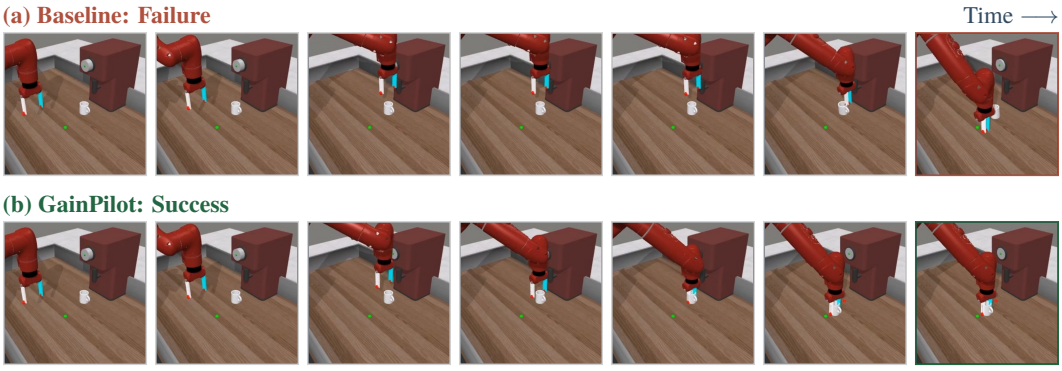

Figure 2: **Coffee pulling in simulation.** Frames progress from left to right. The baseline fails to move the cup to the green target marker, whereas GainPilot pulls the cup toward the target and completes the task.

Table 2: **MT50 comparison** ($\pi_{0.5}$, $H = 10$).

Method	**Success (%)** ↑	**Gain (pp)**	**Calls**	**Interventions**
Native $\pi_{0.5}$	61.6	—	15.130	0.000
Replan	69.4	**+7.8**	14.452	2.034
Matched Replan	68.8	**+7.2**	14.540	—
VLA-Corrector	69.8	**+8.2**	19.962	3.200
GAF	68.0	**+6.4**	14.864	N/A
GainPilot	**74.2**	**+12.6**	14.248	2.050

Gains are relative to Native; counts average all 500 episodes. GAF applies continuous guidance. Native/Replan/GainPilot reuse Table 1. Matched Replan's intervention count is unreported (—).

fits per objective are evaluated on full MT50. L11 achieves 73.33 ± 1.15%, improving over BCE

by 6.00 pp and BCE+gain by 4.00 pp (Table 3); both difference terms contribute to the combined objective.

Table 3: **Full MT50 loss ablation** ($\pi_{0.5}$, $H = 10$).

Objective	Success (%) ↑	Gain (pp)	Calls
L00: BCE	67.33 ± 1.15	—	15.813
L10: BCE + gain	69.33 ± 1.15	+2.00	16.253
L01: BCE + 0.5 pair	68.00 ± 2.00	+0.67	16.260
L11: BCE + gain + 0.5 pair	$\mathbf{73.33 \pm 1.15}$	**+6.00**	15.513

Mean ± sample SD over three independent fits on full MT50; gains are vs. BCE, computed before rounding. Calls average all episodes; interventions are repeated.

Intervention count. At $H = 10$, one GainPilot intervention raises success from 61.6% to 71.0%; repeated interventions reach 74.2%. Correction exceeds Replan by 3.8/4.8 pp under single/repeated limits, with normal replanning retained. Appendix D reports all five horizons and dynamics capacity.

4.4 Evaluation on LIBERO and Real-World Tasks

LIBERO. Both $\pi_{0.5}$ checkpoints use 20 fine-tuning epochs: full fine-tuning uses the official public dataset, and 10-shot uses ten fixed complete trajectories per task. With 100 evaluation episodes per task, GainPilot raises 10-shot mean success from 79.00% to 90.75% (+11.75 pp), with the largest gain on Long (+14.0 pp; Table 4).

Table 4: **LIBERO success (%).**

Suite	Full FT	10-shot FT	10-shot + GainPilot	Gain (pp)
Spatial	**98.00**	84.00	95.00	**+11.00**
Object	**100.00**	88.00	98.00	**+10.00**
Goal	**99.00**	74.00	86.00	**+12.00**
Long	**91.00**	70.00	84.00	**+14.00**
Avg.	**97.00**	79.00	90.75	**+11.75**

FT: fine-tuning; all configurations use $\pi_{0.5}$. Each task uses 100 evaluation episodes. Avg. equally weights suites; gains are vs. 10-shot FT.

Real-world manipulation. We use the same $\pi_{0.5}$ checkpoint on a six-axis PiPER with a two-finger gripper at 30 Hz. Nine tasks cover Pick-place, Alignment, and Disturbance Recovery, with 20 trials per task and method. GainPilot improves all nine tasks, raising average success from 53.3% to 73.9% (37 more successes in 180 trials; Table 5). Figure 3 shows cube stacking; Appendix E details the tasks.

Table 5: **Real-world success on AgileX PiPER (%).**

Method	Pick-place	Alignment	Disturbance Recovery	Avg.
$\pi_{0.5}$	68.3 ± 11.8	48.3 ± 12.6	43.3 ± 12.5	53.3 ± 7.1
GainPilot	$\mathbf{83.3 \pm 9.4}$	$\mathbf{71.7 \pm 11.4}$	$\mathbf{66.7 \pm 11.9}$	$\mathbf{73.9 \pm 6.3}$
Gain (pp)	**+15.0**	**+23.3**	**+23.3**	**+20.6**

Each category contains three tasks, with 20 trials per task and method. ±: 95% CI half-width (Wald by category, stratified for Avg.; Appendix E). Gains use counts before rounding.

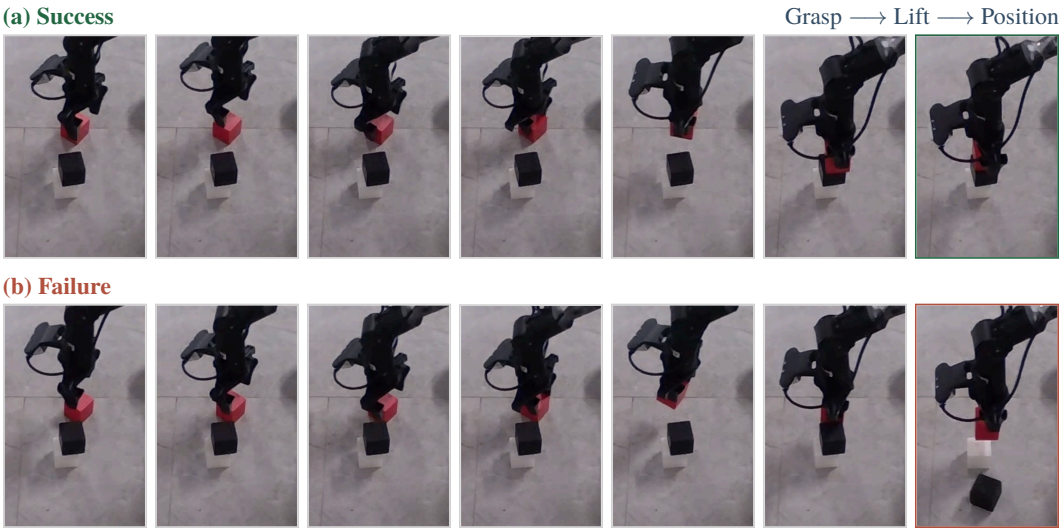

Figure 3: **Real-robot cube stacking: success and failure.** Frames progress left to right through grasping, lifting, and positioning. In **(a)**, the red cube aligns above the stack; in **(b)**, the black supporting cube is displaced in the final frame.

5 Discussion and Conclusion

GainPilot learns when to replace an active plan and how to correct its replacement. Dynamics-informed gain supervision selects interventions, while absolute-success, anchor-relative gain, and paired local-difference losses guide bounded single-gradient correction. The Native–Replan–GainPilot comparisons and loss ablation demonstrate the complementary benefits of these decisions and supervision signals. GainPilot improves success across three MT50 backbones and execution horizons, extends 10-shot LIBERO performance, and raises real-robot success from 53.3% to 73.9%. Together, these results establish an effective execution framework for steering frozen VLA policies.

AI use statement. Generative AI assisted editing, translation, literature retrieval, and LaTeX preparation; authors supplied experimental observations and performance numbers.

Reproducibility statement. Sections 3–4 specify the method and evaluation; the appendices detail architectures, collection, training, execution, and supplementary results.

References

Christopher Agia, Rohan Sinha, Jingyun Yang, Ziang Cao, Rika Antonova, Marco Pavone, and Jeannette Bohg. Unpacking failure modes of generative policies: Runtime monitoring of consistency and progress. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 689–723. PMLR, 2025. URL https://proceedings.mlr.press/v270/agia25a.html.

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

Kevin Black, Manuel Y. Galliker, and Sergey Levine. Real-time execution of action chunking flow policies. In *Advances in Neural Information Processing Systems*, volume 38, 2025b. doi: 10.52202/085713-1122. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/300ccb2187dedd4edcc07f7e76d8e553-Abstract-Conference.html.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin CM Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, 2023. doi: 10.15607/RSS.2023.XIX.026. URL https://www.roboticsproceedings.org/rss19/p026.html.

Dibya Ghosh, Homer Rich Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Quan Vuong, Ted Xiao, Pannag R Sanketi, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, 2024. doi: 10.15607/RSS.2024.XX.090. URL https://www.roboticsproceedings.org/rss20/p090.html.

Qiao Gu, Yuanliang Ju, Shengxiang Sun, Igor Gilitschenski, Haruki Nishimura, Masha Itkina, and Florian Shkurti. SAFE: Multitask failure detection for vision-language-action models. In *Advances in Neural Information Processing Systems*, volume 38, pp. 40041–40076. Curran Associates, Inc., 2025. doi: 10.52202/085713-1337. URL https://papers.nips.cc/paper_files/paper/2025/hash/392d0d05e2f514063e6ce6f8b370834c-Abstract-Conference.html.

Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 9902–9915. PMLR, 2022. URL https://proceedings.mlr.press/v162/janner22a.html.

Tobias Johannink, Shikhar Bahl, Ashvin Nair, Jianlan Luo, Avinash Kumar, Matthias Loskyll, Juan Aparicio Ojea, Eugen Solowjow, and Sergey Levine. Residual reinforcement learning for robot control. In *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019. doi: 10.1109/ICRA.2019.8794127. URL https://ieeexplore.ieee.org/document/8794127/.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, 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. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025. URL https://proceedings.mlr.press/v270/kim25c.html.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit Q-learning. *arXiv preprint arXiv:2110.06169*, 2021. URL https://arxiv.org/abs/2110.06169.

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

Junnan Nie, Jiayi Li, Jiachen Zhang, Junyi Lao, Chenghao Liu, Tianle Zhang, Liang Lin, and Songfang Huang. PACE: Phase-aware chunk execution for robot policies with action chunking. *arXiv preprint arXiv:2606.00537*, 2026. URL https://arxiv.org/abs/2606.00537v2.

Yi Pan, Miao Pan, Qi Lu, Jiaming Huang, Man Zhang, Siteng Huang, Xin Li, Jie Zhang, Yongliang Shen, Xuhong Zhang, and Wenqi Zhang. VLA-Corrector: Lightweight detect-and-correct inference for adaptive action horizon. *arXiv preprint arXiv:2607.01804*, 2026. URL https://arxiv.org/abs/2607.01804.

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 Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025. URL https://arxiv.org/abs/2504.16054.

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

Zhipeng Tang, Xinda Chen, Weining Rao, Xiao Li, Wenting Tan, Yuning Wang, Xiao Shi, and Xiaofang Zhao. Fewer steps, better actions: Rethinking flow-matching inference for VLA policies. *arXiv preprint arXiv:2609.21216*, 2026. URL https://arxiv.org/abs/2609.21216.

Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=AHvFDPi-FA.

Chen Xu, Tony Khuong Nguyen, Emma Dixon, Christopher Rodriguez, Patrick Miller, Robert Lee, Paarth Shah, Rares Andrei Ambrus, Haruki Nishimura, and Masha Itkina. Can we detect failures without failure data? uncertainty-aware runtime failure detection for imitation learning policies. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, 2025. doi: 10.15607/RSS.2025.XXI.073. URL https://www.roboticsproceedings.org/rss21/p073.html.

Weichen Xu, Zhenhua Liu, Lin Luo, Yaobo Liang, Chengtang Yao, Qingyu Mei, Jian Cao, Xixin Cao, Xing Zhang, Jiaolong Yang, and Baining Guo. Continue or replan? Bernoulli-continuation policy learning for adaptive horizon execution. *arXiv preprint arXiv:2608.03483*, 2026. URL https://arxiv.org/abs/2608.03483v1.

Liuhaichen Yang, Zhuang Jiang, Chenchao Sheng, and Zezhi Tang. Guided action flow: Q-guided inference for flow-matching vision-language-action policies. *arXiv preprint arXiv:2607.02092*, 2026. URL https://arxiv.org/abs/2607.02092v3. Version 3, revised August 23, 2026.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-World: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 1094–1100. PMLR, 2020. URL https://proceedings.mlr.press/v100/yu20a.html.

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

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, Ya-Qin Zhang, Jiangmiao Pang, Jingjing Liu, Tai Wang, and Xianyuan Zhan. X-VLA: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. *arXiv preprint arXiv:2510.10274*, 2025. URL https://arxiv.org/abs/2510.10274.

A Auxiliary Models and Training Data

The following details describe the $\pi_{0.5}$–MT50 configuration used in the main results and loss ablation. The base policy, fine-tuned on MT50 for 10 epochs, uses ten flow-sampling steps to generate a 50-step chunk of three translational commands and one gripper command. Auxiliary parameters and feature preprocessing remain fixed online.

A.1 Context and Architecture

The observation history contains five 4,096-dimensional visual features, five four-dimensional policy-visible proprioceptive observations, four executed commands, a 2,048-dimensional instruction embedding, and the remaining-budget fraction. Incomplete histories are left-padded with validity masks. The fixed visual encoder uses 16×16-to-4×4 average pooling and a fixed orthogonal $2{,}048 \to 256$ channel projection, yielding 4,096 features. Masked averaging of native language tokens gives the instruction embedding. The gain model additionally receives H and the remaining committed queue length, while the critic receives H. The horizon and remaining queue length are divided by 50, and the remaining episode budget by 300.

Dynamics and gain models. The dynamics model D uses a history GRU and an action-conditioned rollout GRU to predict residual visual and proprioceptive changes over five steps. Both GRUs use hidden width $h_D = 512$ by default. The architecture supports widths $h_D \in \{256, 512, 768, 1024, 1536\}$. The gain model G_0 (member G_k in Eq. (1)) combines three 256-dimensional vectors: a history-GRU encoding, an encoding of the next five queued commands, and an encoding of the time-averaged normalized visual changes predicted by D. The remaining-budget fraction, normalized H, and normalized remaining commitment length complete the input to a $771 \to 256 \to 1$ head. Hidden activations use SiLU and training dropout is 0.1; the output is a real-valued regression score. Two dynamics–gain pairs are used. Each D member has approximately 10M trainable parameters, and each G_0 member has 3,227,393 (approximately 3.23M).

Anchored critic. Each critic receives a 62-dimensional state feature: 16 visual PCA components, eight language PCA components, 20 proprioceptive-history values, 16 executed-action values, the remaining-budget fraction, and normalized H. Visual PCA uses the 8,192-dimensional concatenation of the latest feature and masked history mean; language PCA follows parameter-free LayerNorm. Centered SVD fits each projection on training contexts; projected components use training standard deviations floored at 10^{-3}. Preprocessing is fixed per member, with standardized visual features and normalized PCA outputs clipped to $[-8, 8]$.

The critic has base width $w = 512$, with the MLPs in Table 6. Hidden layers use SiLU without dropout or batch normalization. Anchor and action-difference inputs each contain 40 coordinates; the difference scales are $d = (0.15, 0.15, 0.15, 0.40)$, broadcast over ten steps. Four members, each with 3,782,146 parameters (approximately 3.78M), contribute their mean sigmoid probability to the correction objective. Auxiliary parameter counts exclude the base policy, visual encoder, and preprocessing buffers.

Table 6: **Critic architecture per member ($w = 512$).**

Module	Layer widths
State encoder E_s	$62 \to 1024 \to 512$
Anchor encoder E_a	$40 \to 512 \to 512$
Difference encoder E_δ	$40 \to 512 \to 512$
Baseline head H_Q	$1024 \to 1024 \to 1$
Advantage head F	$1536 \to 1024 \to 1$
Reported size per member	**3,782,146 parameters (≈3.78M)**

A.2 Branch Collection and Candidate Outcomes

Collection uses 20 dynamics/gain sources and 50 critic sources per task, sharing observations, commands, and trajectory records across the networks. A source is specified by its task, initial arrangement, execution prefix, cut, and normal horizon H. Branches restart from that arrangement and replay the prefix with closed-loop tracking. Pose, grasp, and contact information identify state-consistent replay groups. Each branch records its actual cut observation, committed queue, horizon H, remaining commitment, and remaining episode budget.

Ordinary trajectories and random-candidate branches train D to predict five-step visual and proprioceptive changes from actual commands. After the final four-member Q ensemble and correction configuration are frozen, gain-data collection compares its actual corrected $C = 10$ replacement with preserving the active queue. The corrected branch then resumes μ_H from the latest observation; the keep branch follows the original queue and its natural replanning clock. Both use the source horizon and original episode budget. Each gain source contributes one corrected-versus-keep pair ($R_G = 1$), whose terminal-outcome difference forms the target in Eq. (2).

Critic candidates use the bounded prefix b of a new plan as anchor. Independent standard-Gaussian knots at positions 0, 4.5, and 9 are linearly interpolated into ten-step XYZ, gripper, and joint directions, then bounded using Appendix C's mask and groupwise limits. Opposite signs share one bounded increment; the half-amplitude joint pair halves the full increment. The anchor, XYZ pair, gripper pair, and half/full joint pairs give nine candidates (0–8) and four pairs:

$$\mathcal{P} = \{(1, 2), (3, 4), (5, 6), (7, 8)\}. \tag{A1}$$

Each candidate executes ten steps, discards its unused tail, and resumes ordinary planning with the source horizon H from the latest observation under the original episode budget. Each fixed candidate has $R = 4$ continuations, with random streams matched across candidates within a repetition and varied across repetitions. Mean terminal outcomes supervise absolute success, anchor-relative gains, and local pair differences.

A.3 Data Splits and Training

Data are split into training and test sets by initial scene, keeping its prefixes, cuts, candidates, repetitions, and H variants together. With the base policy and features fixed, D and Q members are independently trained offline from random initialization, then frozen. The final Q ensemble generates the corrected branches used to collect gain labels. Each G_0 member is then trained by mean squared error on paired terminal-outcome differences, using predictions from its paired frozen D member. This $D/Q \to G_0$ sequence produces two D, two G_0, and four Q members for the final repeated-intervention controller.

Training batches and resources. Training uses FP32 AdamW on eight NVIDIA H100 80 GB GPUs, one per member. Dynamics batches contain 64 ordinary windows and 16 four-candidate local groups; critic/gain batches contain 32/16 source groups. Dynamics/gain sampling is stratified by task, horizon, and source; critics bootstrap sources within task–horizon strata. Members use independent initialization and sampling streams.

B Loss Weights and Optimization

Dynamics supervision. Dynamics targets cover five future steps with validity masks. Local groups draw four action-distinct candidates from a source's first repetition, yielding six pairs. Let Avg_m average feature coordinates, then valid sample–time entries; v, p denote visual/proprioceptive targets, and s_v the coordinate-wise RMS of training-set five-step visual changes. The losses are

$$\begin{aligned}
\mathcal{E}_v &= \mathrm{Avg}_{m_v}\left[\left((\widehat{v} - v)/s_v\right)^2\right], \\
\mathcal{E}_p &= \mathrm{Avg}_{m_p}\left[\left((\widehat{p} - p)/0.1\right)^2\right], \\
\mathcal{E}_{\Delta v} &= \mathrm{Avg}_{m_i \wedge m_j}\left[\left(\frac{(\widehat{v}_i - \widehat{v}_j) - (v_i - v_j)}{s_v}\right)^2\right].
\end{aligned} \tag{A2}$$

The dynamics objective balances ordinary and local supervision:

$$\mathcal{L}_D = \tfrac{1}{2}\left(\mathcal{E}_v^{\text{ord}} + 0.1\mathcal{E}_p^{\text{ord}}\right) + \tfrac{1}{2}\left(\mathcal{E}_v^{\text{local}} + 0.1\mathcal{E}_p^{\text{local}} + \mathcal{E}_{\Delta v}^{\text{local}}\right). \tag{A3}$$

Here $\mathcal{E}_{\Delta v}$ supervises visual-effect differences between candidate actions. Visual standard deviations are floored at 0.01, and s_v at 0.01 times those deviations.

Anchored critic supervision. To avoid overcounting identical raw commands, let γ_{ij} denote candidate j's alias group in source i, with $n_i(\gamma)$ recorded slots. The weights in Eq. (5) are

$$w_{ij}^{\text{BCE}} = \frac{1}{n_i(\gamma_{ij})}, \qquad w_{ij}^{\text{gain}} = \frac{\mathbf{1}[\gamma_{ij} \neq \gamma_{i0}]}{n_i(\gamma_{ij})}. \tag{A4}$$

Pairs within one alias group receive zero weight; repeated unordered pairs of distinct groups count once. Each term averages within a source before averaging over the mini-batch; zero-weight terms contribute zero. The BCE, gain, and pair weights are 1, 1, and 0.5, respectively. Difference losses operate in probability space, without division by perturbation radius.

All five critic subnetworks are optimized jointly, with feature preprocessing fixed.

The loss ablation retrains a four-member ensemble for each objective and fit, matching initialization, bootstrap samples, and batch order across objectives. Three fits provide the mean and sample standard deviation in Table 3. The base policy, dynamics/gain models, and feature encoder remain fixed throughout this comparison.

C Execution Details and Computation

Figure 4 summarizes the control flow and offline training order.

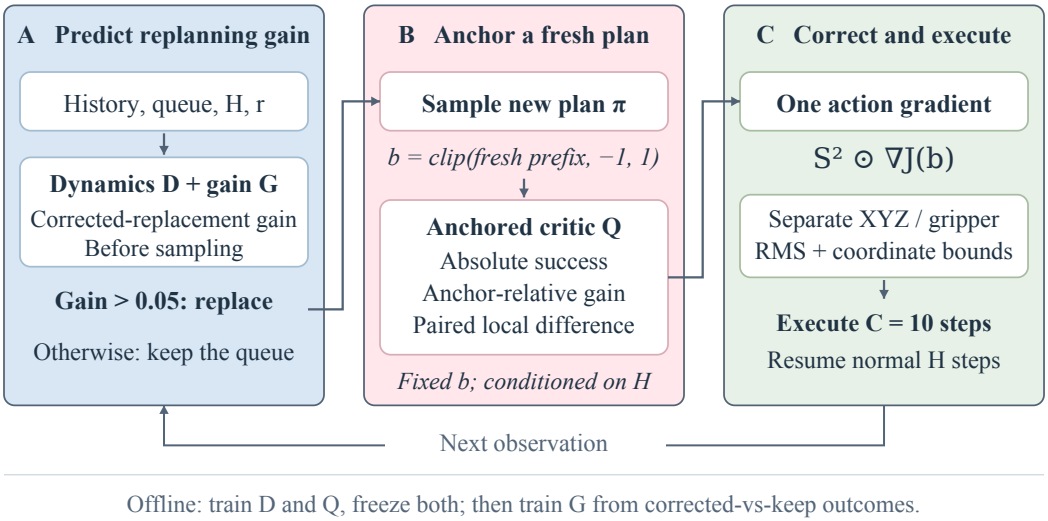

Figure 4: **Controller and training overview.** GainPilot scores corrected-replacement gain before sampling, anchors the new prefix, and applies bounded correction. Offline, independently trained D and Q are frozen before fitting G to corrected-versus-keep outcomes.

Groupwise bounds and online controller. The following bounds convert Eq. (7)'s direction into the executable prefix correction. The coordinate derivation and numerical limits are given in Appendix C.

Independent groupwise bounds. XYZ and gripper groups independently convert this direction into bounded command changes. For coordinates j across the C-step prefix, let g denote a group, L_g its coordinate-wise limit, ρ_g its RMS limit, and η a boundary margin:

$$M_j = \mathbf{1}[1 - |b_j| \geq \eta], \quad B_j = \min(1 - |b_j|, L_g),$$
$$\widetilde{d}_j = M_j d_{\text{GainPilot},j}, \qquad n_g = \sqrt{\frac{1}{|g|}\sum_{j \in g} \widetilde{d}_j^2}. \tag{A5}$$

If $n_g \leq \epsilon$, the group receives zero correction. Otherwise, its normalized direction and feasible amplitude are

$$e_j = \frac{\widetilde{d}_j}{\max(n_g, \epsilon)},$$
$$c_g = \min_{\substack{j \in g \\ |e_j| > \epsilon}} \frac{B_j}{|e_j|}, \tag{A6}$$
$$r_g = (1 - \xi)\max(0, \min(\rho_g, c_g)), \qquad \delta_j = r_g e_j,$$

where ξ is a numerical contraction factor; all limits and tolerances are given in Appendix C.

RMS normalization includes masked coordinates in the denominator. Each group's radius depends only on its direction and command-space margin. The operation sets a feasible amplitude and can enlarge a small nonzero direction, unlike shrink-only clipping.

Monitoring and eligibility. Monitoring runs every ℓ control steps, provided the queue and remaining budget each contain at least ℓ steps. Intervention requires complete history, a positive plan age, at least C steps remaining in the episode budget, and $\overline{G}_t > \tau$, with $\tau = 0.05$. Natural replanning takes precedence, preventing immediate replacement of a fresh normal plan. Algorithm 1 gives the complete execution sequence.

Coordinate transformation. For the four active command dimensions, affine denormalization is $a^{\text{raw}} = \mu + S \odot u$, with $\mu = (q_{99} + q_{01})/2$ and $S = (q_{99} - q_{01})/2$. Holding context and anchor fixed gives

$$\nabla_u J = S \odot \nabla_a J, \qquad S \odot \nabla_u J = S^2 \odot \nabla_a J. \tag{A7}$$

Thus $S^2 \odot \nabla_a J$ expresses an internal-coordinate Euclidean gradient direction in raw-command coordinates. Scales are broadcast across the prefix, zero quantile ranges give zero correction, and padded action dimensions are excluded. This transformation is separate from critic-input normalization by d.

Numerical bounds. For the ten-step prefix, XYZ and gripper groups contain 30 and ten coordinates, respectively. Their RMS limits are $(0.05, 0.10)$ and coordinate-wise limits are $(0.15, 0.40)$. We use boundary margin $\eta = 0.05$, tolerance $\epsilon = 10^{-10}$, and contraction $\xi = 10^{-6}$. Radius calculations use float64 and corrections use float32. Group RMS includes masked coordinates; the groups are normalized independently. Coordinates with zero correction retain their original raw commands.

Controller and sampling. The $\pi_{0.5}$ controller uses $T = 300$, $\ell = 5$, $C = 10$, and strict threshold $\tau = 0.05$. The single-intervention ablation adds a limit of one intervention per episode. Natural-plan and replacement-plan sampling use separate random streams; replacement noise depends on source, policy seed, and intervention time, independently of whether single or repeated interventions are permitted.

Computation. Each intervention generates one replacement chunk and computes one action gradient. Including prefix scores recorded before and after correction for diagnostics, the implementation performs three ensemble forward evaluations (twelve member evaluations). The corrected prefix is then executed under the commitment rule. Computation also includes feature extraction and dynamics/gain monitoring. During online execution, all model parameters remain fixed and differentiation is confined to the candidate actions.

Algorithm 1: GainPilot Online Execution

Inputs:
Policy π; dynamics–gain pairs $\{D_k, G_k\}$; critics $\{Q_m\}$
Normal horizon H; intervention commitment C; episode budget T
Monitoring interval ℓ; gain threshold τ
Initialize:
Empty causal history and active plan; *locked_until* $\leftarrow 0$
for $t = 0, \ldots, T-1$:
Stop if the task has succeeded or the environment has terminated.
Append the current real observation to the causal history.
if no active plan exists or its committed prefix is exhausted:
Discard its unused tail and sample a normal plan from π.
age $\leftarrow 0$; *active_horizon* $\leftarrow H$
if $t \geq$ *locked_until* and $t \bmod \ell = 0$
and $\min(\textit{active_horizon} - \textit{age}, T - t) \geq \ell$:
Set $r_t \leftarrow$ *active_horizon* $-$ *age*; evaluate $\overline{G}_t$ using Eq. (1).
if history is complete and *age* > 0
and $t + C \leq T$ and $\overline{G}_t > \tau$:
Sample one replacement plan from π.
Construct anchor b and critic features f_m conditioned on H.
Compute one action gradient and $d_{\text{GainPilot}}$ using Eq. (7).
Obtain δ using Eqs. (A5)–(A6).
Replace the active plan, preserving unchanged raw coordinates.
age $\leftarrow 0$; *active_horizon* $\leftarrow C$; *locked_until* $\leftarrow t + C$
Execute the active plan's next command and record the actual command.
age $\leftarrow$ *age* $+ 1$
Obtain the next observation and terminal outcome.

Query latency. Table 7 reports mean query times at $H = 10$, measured on NVIDIA H100-HBM3-80G hardware (80 GB HBM3 memory). Timing includes complete action-chunk generation, concurrent dynamics–gain feedback when enabled, and critic-based action correction for GainPilot. GainPilot adds 15.067–20.484 ms per query relative to replanning alone across the three backbones. This small additional latency has a negligible impact on the system's real-time performance.

Table 7: **Mean query time and policy calls ($H = 10$).**

Backbone	Controller	Policy calls per episode	Mean query time (ms) ↓
$\pi_{0.5}$	Native	15.130	189.076
	Replan only	14.452	214.938
	GainPilot	14.248	230.005
SmolVLA	Native	19.532	151.583
	Replan only	20.550	171.168
	GainPilot	20.382	190.567
X-VLA	Native	12.264	86.871
	Replan only	13.402	128.126
	GainPilot	13.412	148.610

Native omits dynamics–gain feedback. Replan only uses this feedback to trigger replanning; GainPilot additionally corrects replacement actions. Policy calls count complete action-chunk generations per episode and match the $H = 10$ results in Table 1.

D Supplementary Experiments

Table 8: **Intervention-count comparison on MT50 ($\pi_{0.5}$, $H = 10$).**

Intervention limit	Method	Success (%) ↑	Gain (pp)	
			over Native	over Replan
0	Native	61.6	—	—
Single (max. 1)	Replan	67.2	+5.6	—
	GainPilot	71.0	+9.4	+3.8
Repeated (no cap)	Replan	69.4	+7.8	—
	GainPilot	**74.2**	**+12.6**	**+4.8**

Each configuration uses 500 episodes. Limits apply to event-triggered interventions per episode; normal replanning remains active. Gains over Replan compare GainPilot with Replan under the same intervention limit.

D.1 Intervention Limits across Execution Horizons

Table 9 compares single- and repeated-intervention settings across all five $\pi_{0.5}$ horizons, using 500 MT50 episodes per configuration and the same initial conditions. Normal replanning remains active under both settings.

Repeated GainPilot interventions improve on single-intervention GainPilot by 2.8–5.2 percentage points across horizons. GainPilot also exceeds Replan under both limits, by 3.8–9.6 points with one intervention and 4.8–8.4 points with repeated interventions.

Table 9: **Intervention limits across MT50 execution horizons ($\pi_{0.5}$).**

H	Native	Replan		GainPilot		GainPilot gain (pp) Repeated minus single
		Single	Repeated	Single	Repeated	
10	61.6	67.2	69.4	71.0	**74.2**	**+3.2**
20	48.0	58.4	59.2	64.4	**67.2**	**+2.8**
30	41.0	55.4	58.4	63.2	**66.2**	**+3.0**
40	41.0	53.0	58.6	61.2	**64.8**	**+3.6**
50	39.2	50.4	56.8	60.0	**65.2**	**+5.2**

Success rates are percentages; gains are percentage points (pp). Every method–horizon entry uses 500 episodes. Native and repeated-intervention results reproduce the corresponding entries in Table 1.

D.2 Dynamics-Model Capacity

We vary the dynamics hidden width h_D while fixing the remaining architecture, ensemble sizes, training data, source splits, update budget, and evaluation initial conditions. The gain and critic widths remain $h_G = 256$ and $w = 512$.

The default width 512 achieves the highest success in every difficulty group and 74.20% overall, compared with 69.00% at width 256 and 71.80% at width 1024 (Figure 5). Avg. weights the 50 tasks equally.

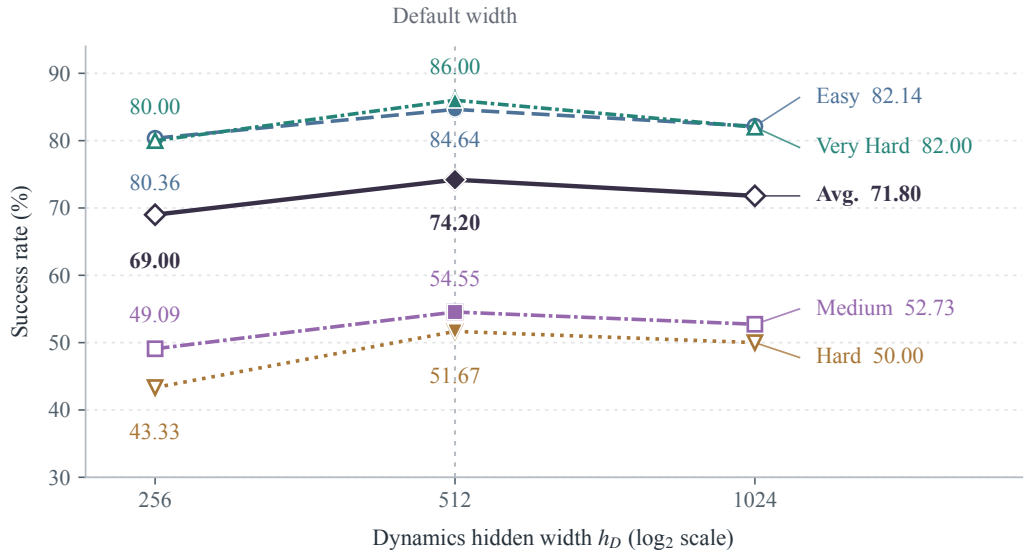

Figure 5: **The default dynamics width 512 achieves the highest success overall and in every difficulty group.** Success rates use the same MT50 evaluation initial conditions; Avg. weights all 50 tasks equally. Widths 256, 512, and 1024 have approximately 5, 10, and 27M trainable parameters per dynamics member, respectively.

E Real-World Task Suite and Detailed Outcomes

E.1 Task Definitions and Trial Counts

The platform is a six-axis AgileX PiPER arm with a two-finger gripper, operated at 30 Hz. Baseline and GainPilot share the same $\pi_{0.5}$ checkpoint. The suite has three tasks per category and 20 trials per task and method, giving 60 trials per category and 180 in total.

Pick-place. *Cube-to-Position* requires grasping a cube and placing it in a fixed target region on the table. *Cube Stacking 1* requires grasping a cube and placing it stably on another cube. *Cube Stacking 2* requires grasping a cube and placing it on two other cubes.

Alignment. *Corner Alignment* requires placing a cube at a designated corner of a platform. *Square Recess Insertion* requires aligning a small cube with a larger square recess and inserting it. *Shallow Slot Placement* requires aligning a cube with a slightly larger shallow slot and placing it inside.

Disturbance Recovery. In *Moving-object Grasp*, the cube is moved when the gripper is about to grasp it. In *Moving Target Region*, the placement target region is moved after the cube has been grasped. In *Moving Tray*, the shallow tray used as the placement target is moved after the cube has been grasped. Success is assessed by completion of the corresponding manipulation task.

E.2 Task-Level Outcomes and Aggregation

Table 10 gives the nine task-level results underlying Table 5. The baseline and GainPilot obtain 96 and 133 successes, respectively, with gains on every task.

E.3 Confidence Intervals

For success proportion $\widehat{p} = s/n$, the reported 95% Wald half-width is $100 \times 1.96\sqrt{\widehat{p}(1-\widehat{p})/n}$, with $n = 20$ per task and $n = 60$ per category. Avg. is the equal-weighted category mean, with

Table 10: Task-level real-world outcomes on AgileX PiPER. Each task uses 20 trials per method.

	Baseline		GainPilot		Gain
Task	Successes	SR ± CI	Successes	SR ± CI	(pp)
Pick-place					
Cube-to-Position	17/20	85.0 ± 15.6	18/20	**90.0 ± 13.1**	**+5.0**
Cube Stacking 1	14/20	70.0 ± 20.1	17/20	**85.0 ± 15.6**	**+15.0**
Cube Stacking 2	10/20	50.0 ± 21.9	15/20	**75.0 ± 19.0**	**+25.0**
Alignment					
Corner Alignment	12/20	60.0 ± 21.5	16/20	**80.0 ± 17.5**	**+20.0**
Square Recess Insertion	9/20	45.0 ± 21.8	13/20	**65.0 ± 20.9**	**+20.0**
Shallow Slot Placement	8/20	40.0 ± 21.5	14/20	**70.0 ± 20.1**	**+30.0**
Disturbance Recovery					
Moving-object Grasp	10/20	50.0 ± 21.9	16/20	**80.0 ± 17.5**	**+30.0**
Moving Target Region	8/20	40.0 ± 21.5	13/20	**65.0 ± 20.9**	**+25.0**
Moving Tray	8/20	40.0 ± 21.5	11/20	**55.0 ± 21.8**	**+15.0**

SR and CI half-widths are percentages. The ± values give 95% Wald confidence-interval half-widths. Gains are GainPilot minus baseline in percentage points.

half-width computed from independent category variances:

$$\overline{p} = \frac{1}{3}\sum_{c=1}^{3}\widehat{p}_c, \qquad \mathrm{HW}_{\mathrm{Avg}} = 100 \times \frac{1.96}{3}\sqrt{\sum_{c=1}^{3}\frac{\widehat{p}_c(1-\widehat{p}_c)}{60}}. \tag{A8}$$

F Evaluation Protocols

Backbone interfaces. Each backbone uses ten fixed sources per MT50 task, reused across its execution horizons. The $\pi_{0.5}$ configuration uses NVIDIA EGL, 30 Hz control, and action scale 0.015. SmolVLA and X-VLA use Mesa rendering, 80 Hz control, and action scale 0.01. Generated chunk lengths are 50 for $\pi_{0.5}$ and SmolVLA and 30 for X-VLA, with $C = 10$ throughout.

External methods. All methods in Table 2 use the same $\pi_{0.5}$ checkpoint, $H = 10$, and 500 evaluation sources. GAF uses training data at the same scale as GainPilot. Its guidance acts continuously during flow sampling, with the intervention-count entry marked N/A.

Matched Replan protocol. The two gain targets share a paired collection protocol. At execution cut x, the keep branch follows the active queue to its original natural replanning boundary and then continues with μ_H. The fresh branch samples a raw replacement, executes its first $C = 10$ commands, discards the remaining tail, and resumes μ_H from the latest observation. Each pair uses the source horizon, matched continuation seeds, and the original episode budget. The raw-replacement target is

$$y_{\mathrm{replan}}(x) = \frac{1}{R_{\mathrm{replan}}}\sum_{r=1}^{R_{\mathrm{replan}}}\left(Y_r^{\mathrm{fresh}} - Y_r^{\mathrm{keep}}\right). \tag{A9}$$

Here R_{replan} is the number of paired repetitions, each Y is a terminal success indicator, and the average covers replacement sampling and continuation randomness. G_{replan} regresses this target with mean squared error. The comparison fixes the base policy, two frozen dynamics members, causal inputs, source splits, two-member gain architecture, optimization settings, and update budget. Each gain ensemble is fitted to its corresponding terminal-outcome target. Online, G_{replan} scores the current queue before replacement sampling and applies the same eligibility conditions, threshold $\tau = 0.05$, and $C = 10$ commitment as GainPilot. Evaluation uses the common 500 MT50 initial conditions at $H = 10$; policy calls count complete chunk generations over all episodes.