SELF-SUPERVISED VLA PREFERENCE OPTIMISATION VIA TEMPORALLY SHIFTED GUIDANCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision–language–action (VLA) models learn diverse robotic behaviors from demonstrations, but improving them beyond imitation often requires costly sources of feedback. Preference optimization offers a direct and effective post-training strategy, yet obtaining useful action comparisons can still depend on human intervention, additional interactions with environments, or external evaluators. We introduce *temporally shifted classifier-free guidance* (TS-CFG) for flow-matching action policies, using their learned action structure to construct hard negatives from demonstrations. The central idea is to steer a demonstration's reconstruction through the model's conditional predictions, targeting hard negatives with subtle deviations from demonstrated behavior. TS-CFG adapts classifier-free guidance by combining predictions under the original context and a nearby temporal context from the same episode. Temporal proximity provides related conditioning, intended to avoid mixing markedly different action distributions while introducing a structured perturbation to generation. Treating each demonstration as preferred over its temporally shifted alternative provides a preference-learning signal using only existing demonstrations and the pretrained policy. This construction supports direct preference post-training without new environment interaction, human preference annotation, or auxiliary world-model or reward evaluation. The additional computation is confined to training, leaving the policy's deployment procedure unchanged. Extensive experiments on RoboTwin and RoboCasa show improvements in task success over the starting policies.

1 INTRODUCTION

We study how to improve a robot policy using only existing demonstrations, removing the need to collect new interaction data or rely on an external evaluator during post-training. This is investigated in the context of Vision–Language–Action (VLA) models, that typically learn control through imitation learning: predicting recorded actions from visual observations and language instructions (Brohan et al., 2023; Kim et al., 2025; Black et al., 2025b;a). Fitting demonstrations teaches a policy which actions to reproduce, but does not explicitly compare them with alternative actions under the same observation. Such comparisons offer a rich additional form of supervision for training stronger policies than imitation learning alone.

Performance may be improved beyond behavioral cloning (BC) by pursuing richer supervision through reinforcement learning (RL) or direct preference optimization (DPO). However, in existing works, both of these introduce additional requirements or challenges: Online RL requires new physical or simulated interactions to improve a policy (Lu et al., 2025; Li et al., 2026). On physical robots, collecting this feedback requires execution time, resets, and often human oversight (Luo et al., 2025; Chen et al., 2025; Amin et al., 2026). Learned world models can supply simulated experience, but introduce an additional dynamics model whose prediction errors can affect policy updates (Zhu et al., 2026; Jiang et al., 2026). DPO provides a simple and robust way to improve a policy by updating it using preference pairs, without separately fitting a reward model or running online RL (Rafailov et al., 2023). However, obtaining ranked preference pairs remains an open challenge. Existing methods use human corrections or intervention labels (Wu et al., 2026; Xia et al., 2025) or use external models to help evaluate candidate behavior (Zhang et al., 2026b; Hung et al., 2025). These external feedback sources impose additional dependencies and DPO efficacy depends on the quality of the external feedback.

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

A few prior works have aimed to solely use existing demonstrations to construct preference supervision. The key question here is how to construct and label preferred and dispreferred action pairs. Shaikh et al. (2025) preferred demonstrated responses to model samples in language modeling, and Zhang et al. (2026a) ranked robot action predictions by their distance to recorded actions. However, choice of alternatives and their ordering remains challenging. Robot behavior can have multiple valid modes (Chi et al., 2023), so distance from a single demonstration need not indicate action quality. We retain demonstration data as the preferred example, and focus on how to construct informative negatives, using the policy's learned action structure to shape the difference between them.

We introduce a self-supervised approach to DPO by steering the denoising process to reconstruct existing actions with subtly different context, aiming to construct *hard negatives*. Here, self-supervision refers to constructing preference pairs from existing demonstrations and the policy, without new preference annotations. Unlike directly adding action noise, this context-shift conditional generation uses the policy's learned dependencies to shape the negatives. Exploiting the generator in this way aims to produce plausible alternatives that differ subtly from the paired demonstration.

The specific context-shift used is the key design parameter in our paradigm. Using an arbitrarily different context may describe a different task, object, or scene, leading to an action distribution poorly matched to the original – and thus an uninformative negative. We instead select a *nearby temporal context from the same demonstration episode* for context-shift. Using the same task and scene encourages relatedness between positive and negative samples, with temporal proximity providing a practical way to obtain subtle differences in action generation.

We instantiate this idea as *temporally shifted classifier-free guidance* (TS-CFG) for VLAs with flow-matching action heads (Black et al., 2025b;a). TS-CFG first inverts the demonstrated action under its original context using a frozen copy of the starting policy. In the exact-flow limit, reconstructing from this latent with unchanged conditioning would recover the demonstration. TS-CFG changes that reconstruction by combining the policy's velocity predictions under the temporal and original contexts, adapting the field-combination principle of classifier-free guidance (Ho & Salimans, 2022). The mixture starts with temporal-context guidance and gradually returns to the original context through convex interpolation. Inversion links the alternative to a specific demonstration, while context guidance changes its reconstruction along a direction supplied by the learned model. Under the ideal flow–score relation, this guidance adds a structured score perturbation determined by the temporal-to-original conditional density ratio. It also admits a *local context smoothing* interpretation: pooling predictions from related contexts reduces reliance on the original context alone.

In summary, we make the following contributions: (1) Introducing context-shift as a self-supervised strategy for DPO post-training using only demonstration data without additional rollouts or external supervision. (2) Instantiating this paradigm through TS-CFG - temporal context shift for generating high-quality synthetic preference pairs. (3) Extensive experiments showing that TS-CFG consistently improves baseline policies. Furthermore this is achieved without modifying the starting policy's action sampler, or requiring test-time scaling through verifiers or candidate selection strategies.

2 Related Work

Vision–language–action models. VLA models combine pretrained visual and linguistic representations with robot demonstrations to learn language-conditioned control. RT-2 and OpenVLA express actions as discrete tokens (Brohan et al., 2023; Kim et al., 2025), while FAST compresses action chunks for efficient autoregressive prediction (Pertsch et al., 2025). Continuous generative policies model action chunks through iterative denoising, as in Diffusion Policy (Chi et al., 2023), or flow integration, as in π_0 and $\pi_{0.5}$ (Black et al., 2025b;a). The latter couple a vision–language backbone with a flow-matching action expert, connecting semantic representations to continuous robot control. We use the resulting conditional velocity fields to construct preference comparisons for post-training, while retaining the policy's architecture and deployment sampler.

RL-based post-training. Reward-based adaptation draws on simulated experience (Lu et al., 2025; Li et al., 2026), robot interaction with human assistance (Luo et al., 2025; Chen et al., 2025; Amin et al., 2026), or learned-world-model rollouts (Zhu et al., 2026; Jiang et al., 2026). For generative policies, DPPO applies policy gradients to diffusion policies (Ren et al., 2025), whereas DSRL learns

a latent-noise policy to steer a frozen action generator (Wagenmaker et al., 2025). TS-CFG instead uses steering during preference-data construction, with demonstrations supplying the supervision.

Preference-based post-training. DPO and Diffusion-DPO learn from preference pairs relative to a reference model (Rafailov et al., 2023; Wallace et al., 2024), but leave the source of the ordering to the application. FlowPRO prefers human corrections to interrupted erroneous policy segments (Wu et al., 2026); APO derives binary action-desirability labels from intervention timing, without requiring matched pairs (Xia et al., 2025). GRAPE ranks trajectories using rewards derived from GPT-4o-generated cost functions for individual task stages, together with policy log-likelihood and task-success indicators (Zhang et al., 2026b). NORA-1.5 scores action candidates by combining the distance between world-model-predicted and demonstrated goal representations with distance to recorded actions (Hung et al., 2025). Other constructions reuse existing data: DITTO prefers demonstrated language responses to model samples (Shaikh et al., 2025), Self-DPO contrasts caption-matched and mismatched images (Peng et al., 2025), and VTLA prefers action predictions closer to recorded actions (Zhang et al., 2026a). TS-CFG also obtains its ordering from demonstrations, but shapes each negative through inversion and temporal-context guidance instead of ranking independent candidates. The existing policy supplies the perturbation direction, requiring neither new human feedback nor an auxiliary model to judge the generated action.

3 Temporally Shifted Classifier-Free Guidance

TS-CFG constructs preference pairs from existing demonstration episodes using a pretrained VLA with a flow-matching action head. We first reverse the flow from a demonstrated action to obtain its latent. We then reconstruct from that latent while mixing velocity predictions under the original context and a temporally nearby context from the same episode. The aim is to construct a closely related hard negative, paired with the demonstration as the preferred action.

For each pair, c_i denotes the original context at demonstration frame i, and $a_i^+ \in \mathbb{R}^{H\times D}$ its demonstrated action chunk. The context contains visual observations, proprioception, and the associated task instruction. The action chunk uses the policy's normalized representation, with horizon H and action width D. We initialize the trainable parameters θ from the starting policy and freeze a reference $\bar{\theta}$ for both negative generation and preference optimization.

3.1 Preliminaries: conditional flows and guidance

Generative action policies represent action chunks through conditional denoising or flow matching (Chi et al., 2023; Black et al., 2025b; Lipman et al., 2023). We use flow time $\tau \in [0, 1]$, with the noise endpoint at $\tau = 0$ and the action endpoint at $\tau = 1$. For an action a and Gaussian noise ϵ, the training path and its velocity target are

$$x_\tau(a, \epsilon) = (1 - \tau)\epsilon + \tau a, \qquad u(a, \epsilon) = a - \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I). \tag{1}$$

A network $v_\phi(x, \tau, c)$ predicts this velocity conditioned on c (Lipman et al., 2023). Generation integrates this field from $\tau = 0$ to $\tau = 1$, while inversion runs the same flow backward from an action to its latent.

Classifier-free guidance controls generation by combining conditional and unconditional predictions (Ho & Salimans, 2022). For velocity predictions, with unconditional input $\emptyset$ and guidance strength w, the combination is

$$v_\phi^{\mathrm{CFG}}(x, \tau, c; w) = v_\phi(x, \tau, \emptyset) + w\big[v_\phi(x, \tau, c) - v_\phi(x, \tau, \emptyset)\big], \tag{2}$$

Setting $w = 1$ recovers the conditional field. TS-CFG adapts this combination to two observed contexts, using convex interpolation to perturb a demonstration's reconstruction.

3.2 Generating negatives from demonstrations

We target hard negatives by steering a demonstration's reconstruction with the policy's learned predictions from a temporally nearby context, rather than adding noise directly to the action. Figure 1b illustrates the intended effect: TS-CFG preserves the broad pattern of the demonstrated motion while introducing a subtle, structured deviation. This gives preference learning a focused comparison between closely related behaviors.

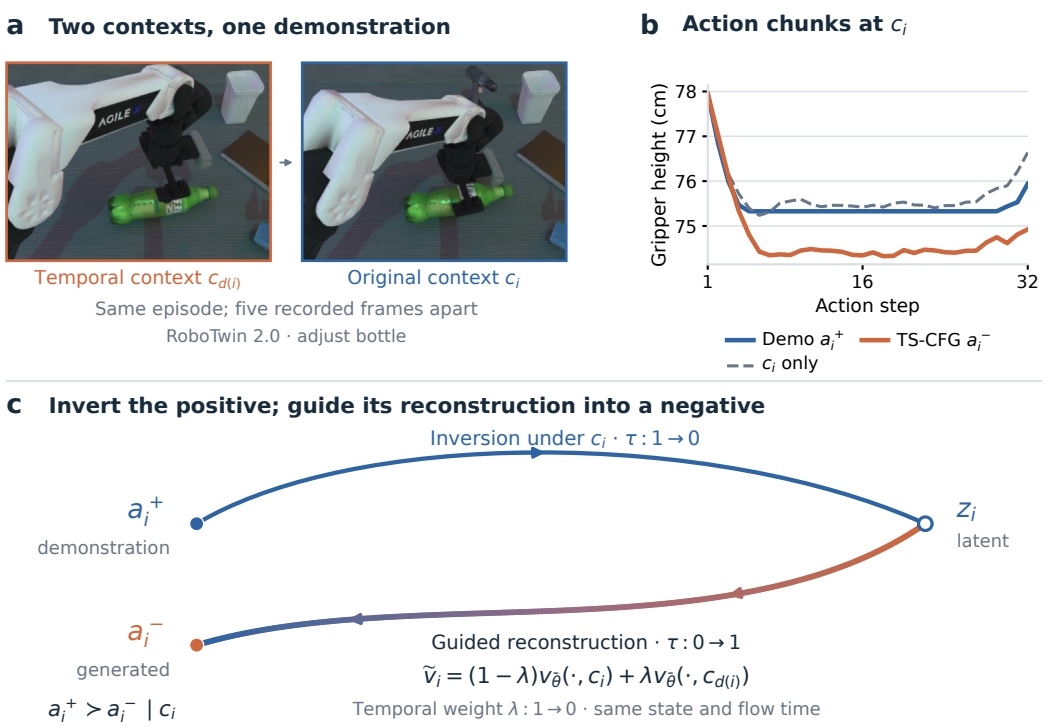

Figure 1: **TS-CFG perturbs reconstruction using the policy's learned conditional fields. (a)** The original RoboTwin 2.0 observation (Chen et al., 2026) c_i is paired with the demonstrated positive a_i^+; nearby context $c_{d(i)}$ supplies guidance. **(b)** Recorded and generated action chunks for this example, visualized as left-gripper height targets through forward kinematics. The gray curve reconstructs under c_i alone; the orange curve uses temporal guidance. These are commanded targets, not executed rollouts. **(c)** Inversion under c_i links a_i^+ to z_i; reconstruction mixes the two velocity predictions at the same state and flow time, with temporal weight decreasing from one to zero. The schematic flow paths illustrate generation, whereas panel (b) shows actual action outputs. The demonstration and generated negative are compared under the same original context c_i.

Original and temporal contexts. For each demonstration pair (c_i, a_i^+), we select a distinct *temporal context* $c_{d(i)}$: the context at a nearby frame in the same recorded episode. Evaluating the frozen policy under this additional context supplies a learned velocity field for perturbing reconstruction under the original context. An arbitrary context may correspond to a different task or scene, so its predictions can redirect reconstruction toward unrelated behavior. Nearby frames retain the task and scene while reflecting changes in the robot and object states. We use this local variation to seek related conditional action distributions and, in turn, subtle changes to the demonstration. The temporal offset $\delta > 0$ controls its separation from the original context; a preceding frame gives $d(i) = i - \delta$. The selected context stays fixed throughout reconstruction and is used only during training.

Demonstration inversion. A random latent may decode to a different action even under the original context, so it is not tied to the paired demonstration. We instead invert a_i^+ under c_i by integrating the flow backward to obtain its latent. With exact integration, decoding this latent under c_i recovers a_i^+. Keeping this latent fixed lets context guidance modify the reconstruction of that specific demonstration. Let $N \geq 2$ be the number of integration steps and $h = 1/N$. Starting from $z_i^{(0)} = a_i^+$, we obtain the latent $z_i = z_i^{(N)}$ by explicit Euler inversion under the original context for $j = 0, \ldots, N-1$:

$$z_i^{(j+1)} = z_i^{(j)} - h v_{\bar{\theta}}(z_i^{(j)}, 1 - jh, c_i). \tag{3}$$

Temporal-context decoding. To generate an alternative from this latent, we steer reconstruction using the difference between the temporal- and original-context velocity predictions. We implement this steering through the mixed field

$$\widetilde{v}_i(x, \tau; \lambda) = (1 - \lambda) v_{\bar{\theta}}(x, \tau, c_i) + \lambda v_{\bar{\theta}}(x, \tau, c_{d(i)}). \tag{4}$$

Both velocity predictions use the same action state x and flow time τ, with $\lambda \in [0, 1]$ controlling a tradeoff between velocities from the original context and the temporal context. We decrease λ during decoding so that the temporal context redirects the initial reconstruction, while the original context guides its completion. Initialize $x_i^{(0)} = z_i$ and, for $k = 0, \ldots, N - 1$, set

$$\tau_k = \frac{k}{N}, \qquad \lambda_k = 1 - \frac{k}{N - 1}. \tag{5}$$

The guided reconstruction follows

$$x_i^{(k+1)} = x_i^{(k)} + h\widetilde{v}_i(x_i^{(k)}, \tau_k; \lambda_k). \tag{6}$$

Its endpoint $a_i^- = x_i^{(N)}$ is the generated alternative.

We generate one alternative per demonstration pair and assign the synthetic preference $a_i^+ \succ a_i^-$ under the original context c_i. The temporal frame supplies conditioning information; its recorded action is not substituted for the negative. No candidate selection or outcome-based filtering is applied.

3.3 Structured score perturbation and context smoothing

To interpret the field in Equation 4, consider an ideal velocity model for the linear path in Equation 1. Let $p_\tau(x \mid c)$ denote its noised action density and $s(x, \tau, c) = \nabla_x \log p_\tau(x \mid c)$ its score. The ideal velocity and score satisfy

$$v^*(x, \tau, c) = \frac{x + (1 - \tau) s(x, \tau, c)}{\tau}, \qquad 0 < \tau < 1. \tag{7}$$

Because the context-independent term is shared, applying the same interpolation to the ideal velocities gives the combined score

$$\begin{aligned} \widetilde{s}_i(x, \tau; \lambda) &= (1 - \lambda) s(x, \tau, c_i) + \lambda s(x, \tau, c_{d(i)}) \\ &= \nabla_x \log p_\tau(x \mid c_i) + \lambda \nabla_x \log \frac{p_\tau(x \mid c_{d(i)})}{p_\tau(x \mid c_i)}. \end{aligned} \tag{8}$$

The guidance therefore adds λ times the difference between the temporal and original scores to the original score. This changes the reconstruction direction using the model's conditional structure; the temporal context supplies the perturbation, while the demonstration remains the preferred action. For fixed τ and λ, the combined score is the gradient of the log density

$$\widetilde{p}_i(x, \tau; \lambda) \propto p_\tau(x \mid c_i) \left[\frac{p_\tau(x \mid c_{d(i)})}{p_\tau(x \mid c_i)} \right]^\lambda. \tag{9}$$

This provides a *local context smoothing* interpretation of the reconstruction field: predictions from a nearby context modify those associated with the original context. Equation 8 connects the choice of context to the perturbation direction, while λ determines how much of that difference is applied.

3.4 Preference post-training with flow errors

For each generated pair, we sample a flow time τ and a Gaussian tensor ϵ for the flow-error objective. The same (τ, ϵ) is shared across the positive and negative actions and across the trainable and reference policies. All four evaluations use the original context c_i. Define the mean squared velocity error

$$\ell_\phi(a, c; \tau, \epsilon) = \frac{1}{HD} \left\| v_\phi(x_\tau(a, \epsilon), \tau, c) - (a - \epsilon) \right\|_F^2. \tag{10}$$

Let $m_{\phi,i} = \ell_\phi(a_i^-, c_i; \tau, \epsilon) - \ell_\phi(a_i^+, c_i; \tau, \epsilon)$. We use a flow-error surrogate for reference-relative preference optimization (Rafailov et al., 2023; Wallace et al., 2024):

$$\mathcal{L}_{\text{pref}}(\theta) = -\mathbb{E}_{i,\tau,\epsilon}\left[\log \text{sigmoid}\left(\beta\left[m_{\theta,i} - m_{\bar{\theta},i}\right]\right)\right], \tag{11}$$

where $\beta > 0$ controls preference strength. Minimizing this loss favors a larger negative-minus-positive error gap than the frozen reference assigns to the same pair. We optimize this preference objective without an additional behavior-cloning term.

4 Experiments

Our experiments examine whether temporally guided reconstruction supplies useful preference supervision beyond the demonstrations used to train a VLA. We evaluate policy improvement across two manipulation benchmarks and use controlled comparisons to examine how negative construction affects post-training.

4.1 Experimental setting

Benchmarks and policies. We evaluate $\pi_{0.5}$ (Black et al., 2025a) on RoboTwin 2.0 (Chen et al., 2026) and RoboCasa365 (Nasiriany et al., 2026). For RoboTwin, we use the checkpoint released by Motus Team (2025); for RoboCasa, we use the benchmark authors' human300 checkpoint at 75,000 steps (RoboCasa, 2026). RoboTwin covers 50 bimanual manipulation tasks; we reuse the release's demonstration mixture of 50 clean and 500 randomized episodes per task, totaling 27,500 episodes. The resulting policy is evaluated under both clean (Easy) and randomized (Hard) conditions with unseen language instructions. For RoboCasa, the released policy is trained on human demonstrations of 300 pretraining tasks. Our post-training uses the 34 pretraining tasks that overlap with its 50-task target benchmark: 18 atomic and 16 composite tasks. We report these two groups separately from the remaining 16 unseen composite tasks, following the benchmark's official task split. All RoboCasa evaluations use target-split kitchens; target-split demonstrations are not used for post-training.

Post-training and evaluation. We train LoRA adapters (Hu et al., 2022) for 5,000 steps with a global batch size of eight and a learning rate of 5×10^{-7}. On RoboTwin, adapters update only the action expert; on RoboCasa, they update both the language backbone and the action expert. Within each benchmark, all post-training methods use the same adapter scope, demonstrations, and update budget. TS-CFG constructs one negative per demonstration using the frozen reference policy, ten integration steps in each direction, and a temporal context five recorded frames away. We use the linearly decreasing guidance schedule from Section 3.2 and set $\beta = 2$ for the DPO loss. We evaluate the final checkpoint on 50 rollouts per task and report average success, with 2,500 rollouts per complete benchmark setting. The deployment sampler is unchanged. Appendix A provides the full optimization, preprocessing, and evaluation settings.

Comparison methods. Continued behavior cloning (BC) applies additional supervised updates with the original flow-matching objective. We also adapt the preference-construction ideas of DITTO (Shaikh et al., 2025) and VTLA (Zhang et al., 2026a) to the same $\pi_{0.5}$ backbone and flow-based preference objective, denoting these variants DITTO* and VTLA*. DITTO* pairs each demonstration with one action sampled from the frozen starting policy under the original context using its ordinary ten-step sampler. It transfers demonstration-over-sample supervision from DITTO's language setting, while omitting its initial SFT stage, iterative generator updates, replay, and inter-model comparisons. VTLA* samples 16 actions from the same frozen policy and selects the candidate with the largest mean squared distance to the demonstrated chunk, measured over all normalized action coordinates including padding. It uses the demonstration itself as the preferred action, whereas VTLA ranks generated predictions by their proximity to recorded actions. Both adaptations retain the demonstrated positive and the shared preference loss to compare alternative ways of constructing negatives.

Latent perturbation. We additionally evaluate TS-CFG + LN, which adds 0.05ξ, with $\xi \sim \mathcal{N}(0, I)$, once to the inverted demonstration latent before reconstruction. Temporal guidance changes the reconstruction field, whereas LN varies its starting point around the inverted demonstration latent. The temporal guidance schedule and preference objective remain unchanged. We report this variant separately to distinguish its empirical contribution from the base TS-CFG construction.

Table 1: **Policy improvement across benchmark settings and task groups.** Success rates are percentages; parentheses give gains in percentage points over $\pi_{0.5}$. RoboTwin Easy and Hard each cover 50 tasks. RoboCasa uses Atomic-S (18 seen atomic tasks), Comp.-S (16 seen composite tasks), and Comp.-U (16 unseen composite tasks); All averages all 50 tasks. *Adapted preference constructions described in Section 4.1. LN adds latent noise with $\sigma_z = 0.05$.

Method	RoboTwin 2.0		RoboCasa365			
	Easy	Hard	Atomic-S	Comp.-S	Comp.-U	All
$\pi_{0.5}$	56.24	58.36	44.56	5.13	1.00	18.00
+ BC	56.16 (−0.08)	58.56 (+0.20)	44.00 (−0.56)	6.63 (+1.50)	0.63 (−0.38)	18.16 (+0.16)
+ DITTO*	58.96 (+2.72)	62.04 (+3.68)	46.33 (+1.78)	7.25 (+2.13)	1.38 (+0.38)	19.44 (+1.44)
+ VTLA*	58.40 (+2.16)	62.20 (+3.84)	43.89 (−0.67)	6.25 (+1.13)	1.00 (+0.00)	18.12 (+0.12)
+ TS-CFG	60.84 (+4.60)	64.08 (+5.72)	49.22 (+4.67)	8.63 (+3.50)	1.25 (+0.25)	20.88 (+2.88)
+ TS-CFG + LN	**61.32** (+5.08)	**65.52** (+7.16)	**49.67** (+5.11)	**10.63** (+5.50)	**1.63** (+0.63)	**21.80** (+3.80)

4.2 Policy improvement across tasks and environments

Table 1 shows that TS-CFG improves RoboTwin success by 4.60 percentage points in Easy and 5.72 points in Hard using the same post-trained checkpoint. On RoboCasa, overall success increases from 18.00% to 20.88%, with gains of 4.67 and 3.50 points on seen atomic and composite tasks, respectively. The improvements extend across RoboTwin's clean and randomized conditions and RoboCasa's household manipulation setting. These results show that comparisons constructed from an existing policy and its demonstrations can supply useful post-training supervision, without additional interaction data or changes to the deployment sampler.

Comparison with BC and adapted preference methods. Continued BC remains close to the starting policy on RoboTwin Easy and Hard, whereas TS-CFG improves over BC by 4.68 and 5.52 points, respectively, with the same data and update budget. TS-CFG exceeds DITTO* by 1.88 and 2.04 points on Easy and Hard, respectively, and VTLA* by 2.44 and 1.88 points. These RoboTwin comparisons support constructing negatives through demonstration-linked reconstruction rather than relying only on further imitation or selecting among ordinary policy samples. On RoboCasa, BC, DITTO*, and VTLA* achieve 18.16%, 19.44%, and 18.12% success, respectively; TS-CFG exceeds them by 2.72, 1.44, and 2.76 points. On this benchmark, selecting the most distant candidate among 16 policy samples yields a smaller improvement than using a single sample, reinforcing that action distance alone does not determine the usefulness of a negative.

Adding a small latent deviation. TS-CFG + LN reaches 61.32% on RoboTwin Easy, 65.52% on Hard, and 21.80% on RoboCasa. These are gains of 0.48, 1.44, and 0.92 points over TS-CFG, respectively. These gains suggest that small deviations around the inverted demonstration latent can produce more useful negatives for preference learning.

4.3 Which properties of the negatives matter?

We study negative construction on RoboTwin Hard, holding the preferred demonstrations, DPO objective, adapter configuration, and training budget fixed. All comparisons cover the same 50 tasks and 2,500 evaluation rollouts. The controls examine whether the gains can be reproduced by directly perturbing demonstrated actions, replacing the inverted latent with random noise, or substituting randomly selected contexts or corrupted inputs for nearby temporal contexts. Baseline for this section is fixed as TS-CFG.

Table 2: **Negative construction.** RoboTwin Hard success (%); gains over $\pi_{0.5}$ in parentheses.

Variant	Success (%)
ActNoise	54.28 (−4.08)
ActMix	47.32 (−11.04)
RandInit	58.16 (−0.20)
RandCtx	58.32 (−0.04)
TS-CFG	**64.08** (+5.72)

Demonstration anchoring and local conditioning play complementary roles. RandInit retains temporal guidance but starts from a fresh Gaussian latent, removing the connection established by demonstration inversion. Its

58.16% success shows that the same guided field is not sufficient when its starting point is unrelated to the paired demonstration. Conversely, RANDCTX preserves inversion but chooses a uniformly sampled distinct frame from the same episode, yielding 58.32%. Thus, sharing the task and scene is also insufficient when temporal proximity is discarded. Together, these controls support the combination used in TS-CFG: inversion determines *which demonstrated behavior* is modified, while nearby conditioning supplies a related field with which to modify it.

Perturbation direction matters. ACTNOISE adds isotropic Gaussian noise to the demonstrated action, whereas ACTMIX linearly interpolates it with the action recorded at the temporal context. We calibrate both on 500 training examples to match TS-CFG's mean squared displacement from the demonstration (Appendix A.3). Despite this matched average magnitude, Table 2 shows that they yield 54.28% and 47.32% success, both below the original policy's 58.36%. A negative's distance from the demonstration is therefore insufficient to characterize its value for preference learning. TS-CFG instead obtains the perturbation direction from the learned conditional fields, achieving 64.08% without directly modifying or interpolating recorded action coordinates. In particular, the poor performance of ACTMIX distinguishes temporal *conditioning* from directly interpolating recorded actions across frames.

Temporal context versus input corruption. We replace temporal conditioning with instruction-token masking (TEXTMASK), scattered image-patch masking (PATCHMASK), a contiguous image cutout (CUTOUT), or Gaussian image blur (BLUR). Inversion still uses the original context, and the corrupted branch enters the same reconstruction schedule; deployment inputs remain uncorrupted. In Table 3, text masking yields 58.96–59.48%, while the strongest tested image corruption, PATCHMASK, reaches 60.52%. All remain below temporal conditioning at 64.08%. For these controls, removing information from an observation supplies less useful preference supervision than conditioning on a nearby observed situation. The comparison favors the proposed source of the field perturbation, rather than attributing the gain to any change in conditioning.

Table 3: **Conditioning source.** RoboTwin Hard success (%); gains over $\pi_{0.5}$ in parentheses.

Variant	Success (%)
TextMask 25%	58.96 (+0.60)
TextMask 50%	59.04 (+0.68)
TextMask 75%	59.48 (+1.12)
PatchMask 10%	60.52 (+2.16)
Cutout 25%	59.16 (+0.80)
Blur $\sigma = 2$	58.96 (+0.60)
TS-CFG	**64.08** (+5.72)

4.4 HOW MUCH SHOULD RECONSTRUCTION BE PERTURBED?

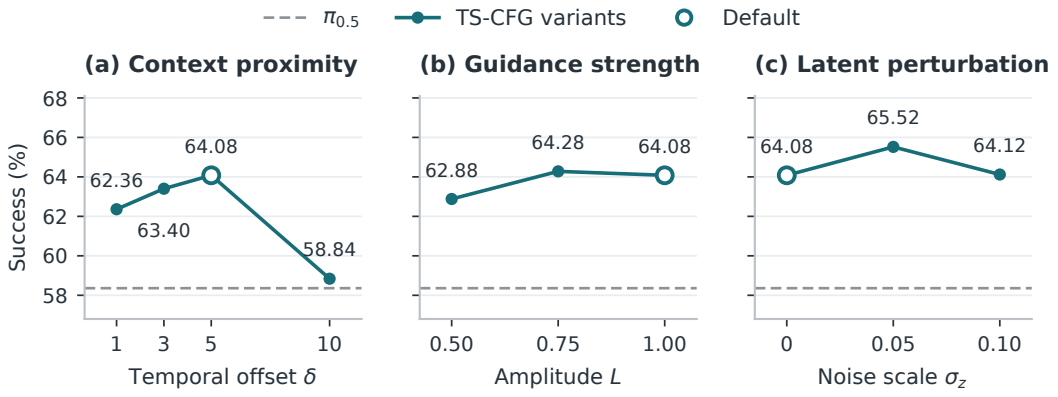

Figure 2: **Sensitivity to context proximity and perturbation strength.** Each panel changes one factor on RoboTwin Hard: (a) temporal offset, (b) the amplitude of the decreasing guidance schedule, and (c) Gaussian noise added once to the inverted latent. Open markers denote the default TS-CFG setting; the dashed line denotes $\pi_{0.5}$. Values are success rates over 2,500 rollouts.

A useful context is nearby, but need not be the nearest. Offsets of one, three, and five recorded frames yield 62.36%, 63.40%, and 64.08%, respectively; increasing the offset to ten reduces success to 58.84% (Figure 2a). Together with RANDCTX, this trend motivates a balance between changing the conditioning and retaining its relation to the original observation. Very close contexts may supply little change, whereas more distant contexts can alter the learned reconstruction field substantially. The results are consistent with this motivation: the best tested offset is intermediate, and increasing temporal separation does not monotonically strengthen the supervision.

The benefit persists across moderate guidance strengths. Scaling the schedule to $\lambda_k = L(1 - k/9)$ gives 62.88%, 64.28%, and 64.08% at $L = 0.50$, 0.75, and 1.00 (Figure 2b). All three settings improve over $\pi_{0.5}$, with similar success at $L = 0.75$ and $L = 1.00$. Thus, useful preference supervision is obtained across the tested range of guidance strengths.

Table 4: **Guidance schedule.** RoboTwin Hard success (%).

Variant	λ_k	Success
CFGI 0.10	0.10	63.28
CFGI 0.25	0.25	62.88
CFGI 0.50	0.50	56.56
TS-CFG	$1 - k/9$	64.08

Table 5: **Latent noise and guidance.** RoboTwin Hard success (%).

Variant	λ_k	σ_z	Success
LN-only	0	0.05	59.60
TS-CFG	$1 \to 0$	0	64.08
TS-CFG + LN	$1 \to 0$	0.05	65.52

Constant versus decreasing guidance. We also compare TS-CFG's decreasing schedule with constant-weight context interpolation (CFGI), using $\lambda_k = \lambda$ throughout reconstruction for $\lambda \in \{0.10, 0.25, 0.50\}$ (Table 4). These variants retain original-context inversion and the five-frame temporal context. The constant 0.50 schedule has the same mean coefficient as TS-CFG's ten-step schedule, but distributes temporal conditioning uniformly instead of concentrating it near the start of reconstruction. Constant weights of 0.10 and 0.25 achieve 63.28% and 62.88% success, respectively, compared with 64.08% for TS-CFG, whereas a constant weight of 0.50 reduces success to 56.56%. Despite having the same mean guidance coefficient, the constant 0.50 and decreasing schedules produce markedly different outcomes, indicating that when temporal guidance is applied matters beyond its average strength. These results favor stronger temporal guidance early in reconstruction followed by a gradual return to the original-context field.

Latent perturbation complements temporal guidance. LN-ONLY applies the same $\sigma_z = 0.05$ perturbation after demonstration inversion but reconstructs using only the original-context field, setting $\lambda_k = 0$ throughout. It retains the demonstrated positive and the same preference objective and training budget. On RoboTwin Hard, it reaches 59.60% success, compared with 65.52% for TS-CFG + LN at the same noise scale (Table 5). Temporal guidance therefore provides gains that latent noise alone does not recover, while LN further improves TS-CFG's 64.08% success.

Sensitivity to latent perturbation. We then vary σ_z in the reconstruction starting point $z_i + \sigma_z \xi$ while keeping temporal guidance fixed. The smaller perturbation, $\sigma_z = 0.05$, achieves 65.52%, whereas doubling the scale to $\sigma_z = 0.10$ yields 64.12% (Figure 2c). Thus, increasing the noise scale removes most of the additional gain over TS-CFG.

5 CONCLUSION

We introduced TS-CFG, a preference post-training method for flow-matching VLA policies that constructs action comparisons from an existing policy and demonstration dataset. By steering demonstration reconstruction with a nearby temporal context, TS-CFG uses the policy's learned conditional fields to generate structured alternatives for preference learning. Experiments on RoboTwin and RoboCasa show improved task success while retaining the ordinary deployment sampler. We also find that a small perturbation of the inverted latent provides an additional empirical gain across the three evaluation settings. The ablations indicate that the source and direction of the perturbation matter: matching action displacement, removing demonstration inversion, or replacing nearby conditioning with arbitrary context changes does not recover the same benefit. Together, these findings support using learned generative structure to turn existing demonstrations into useful comparative supervision, without collecting new human preferences or relying on an auxiliary evaluator.

AI USE STATEMENT

We used generative AI tools to assist with manuscript drafting and editing, literature search, experimental implementation and debugging, and figure preparation. These tools also provided feedback on experimental design and supported discussion of methodological interpretations and results. The authors take responsibility for the final text, code, analyses, and claims.

ETHICS STATEMENT

This work studies robot policy post-training using existing demonstration datasets and simulation benchmarks. It does not involve new human-subject studies or physical-robot experiments. Improved benchmark success does not establish safe operation around people or in uncontrolled environments; physical deployment would require application-specific safety evaluation and appropriate safeguards.

REPRODUCIBILITY STATEMENT

Section 3 specifies negative construction and the preference objective, and Section 4 identifies the policy releases, benchmarks, and evaluation setting. Appendix A documents data preprocessing, optimization settings, numerical integration, ablation definitions, and the rollout protocol, with numerical settings collected in Table 6.

REFERENCES

Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, Danny Driess, Michael Equi, Adnan Esmail, Yunhao Fang, Chelsea Finn, Catherine Glossop, Thomas Godden, Ivan Goryachev, Lachy Groom, Hunter Hancock, Karol Hausman, Gashon Hussein, Brian Ichter, Szymon Jakubczak, Rowan Jen, Tim Jones, Ben Katz, Liyiming Ke, Chandra Kuchi, Marinda Lamb, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Yao Lu, Vishnu Mano, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Charvi Sharma, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, Will Stoeckle, Alexander Swerdlow, James Tanner, Marcel Torne, Quan Vuong, Anna Walling, Haohuan Wang, Blake Williams, Sukwon Yoo, Lili Yu, Ury Zhilinsky, and Zhiyuan Zhou. $\pi^*_{0.6}$: a VLA that learns from experience. In *Proceedings of Robotics: Science and Systems*, 2026. doi: 10.15607/RSS.2026.XXII.087. URL https://www.roboticsproceedings.org/rss22/p087.html.

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. In *Proceedings of The 9th Conference on Robot Learning*, volume 305, pp. 17–40, 2025a. URL https://proceedings.mlr.press/v305/black25a.html.

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. π_0: A vision-language-action flow model for general robot control. In *Robotics: Science and Systems*, 2025b. doi: 10.15607/RSS.2025.XXI.010. URL https://roboticsproceedings.org/rss21/p010.html.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu,

Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control. In *Proceedings of The 7th Conference on Robot Learning*, volume 229, pp. 2165–2183, 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, Weiliang Deng, Yubin Guo, Tian Nian, Xuanbing Xie, Qiangyu Chen, Kailun Su, Tianling Xu, Guodong Liu, Mengkang Hu, Huan ang Gao, Kaixuan Wang, Zhixuan Liang, Yusen Qin, Xiaokang Yang, Ping Luo, and Yao Mu. RoboTwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. In *International Conference on Machine Learning*, 2026. URL https://icml.cc/virtual/2026/poster/62192.

Yuhui Chen, Shuai Tian, Shugao Liu, Yingting Zhou, Haoran Li, and Dongbin Zhao. ConRFT: A reinforced fine-tuning method for VLA models via consistency policy. In *Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.019. URL https://www.roboticsproceedings.org/rss21/p019.html.

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

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2022. URL https://arxiv.org/abs/2207.12598.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Chia-Yu Hung, Navonil Majumder, Haoyuan Deng, Liu Renhang, Yankang Ang, Amir Zadeh, Chuan Li, Dorien Herremans, Ziwei Wang, and Soujanya Poria. NORA-1.5: A vision-language-action model trained using world model- and action-based preference rewards, 2025. URL https://arxiv.org/abs/2511.14659.

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.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Pannag 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, pp. 2679–2713, 2025. URL https://proceedings.mlr.press/v270/kim25c.html.

Haozhan Li, Yuxin Zuo, Jiale Yu, Yuhao Zhang, Zhaohui Yang, Kaiyan Zhang, Xuekai Zhu, Yuchen Zhang, Tianxing Chen, Ganqu Cui, Dehui Wang, Dingxiang Luo, Yuchen Fan, Youbang Sun, Jia Zeng, Jiangmiao Pang, Shanghang Zhang, Yu Wang, Yao Mu, Bowen Zhou, and Ning Ding. SimpleVLA-RL: Scaling VLA Training via Reinforcement Learning. In *International Conference on Learning Representations*, pp. 125846–125869, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/cbfbcb4da14235bd69b134070898ae9d-Abstract-Conference.html.

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

Guanxing Lu, Wenkai Guo, Chubin Zhang, Yuheng Zhou, Haonan Jiang, Zifeng Gao, Yansong Tang, and Ziwei Wang. VLA-RL: Towards masterful and general robotic manipulation with scalable reinforcement learning, 2025. URL https://arxiv.org/abs/2505.18719. To appear in the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2026.

Jianlan Luo, Charles Xu, Jeffrey Wu, and Sergey Levine. Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning. *Science Robotics*, 10(105):eads5033, 2025. doi: 10.1126/scirobotics.ads5033. URL https://doi.org/10.1126/scirobotics.ads5033.

Motus Team. pi0.5_robotwin2. Hugging Face model release, 2025. URL https://huggingface.co/motus-robotics/pi0.5_robotwin2.

Soroush Nasiriany, Sepehr Nasiriany, Abhiram Maddukuri, and Yuke Zhu. RoboCasa365: A large-scale simulation framework for training and benchmarking generalist robots. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/a05003fdb1e9562ab0c0a9719ea4de10-Abstract-Conference.html.

Liang Peng, Boxi Wu, Haoran Cheng, Yibo Zhao, and Xiaofei He. Self-supervised direct preference optimization for text-to-image diffusion models. In *Advances in Neural Information Processing Systems*, volume 38, pp. 2332–2346, 2025. doi: 10.52202/085713-0083. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/03600ae6c3392fd65ad7c3a90c6f7ce8-Abstract-Conference.html.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. FAST: Efficient action tokenization for vision-language-action models. In *Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.012. URL https://www.roboticsproceedings.org/rss21/p012.html.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, volume 36, pp. 53728–53741, 2023. doi: 10.52202/075280-2338. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/a85b405ed65c6477a4fe8302b5e06ce7-Abstract-Conference.html.

Allen Z. Ren, Justin Lidard, Lars L. Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion policy policy optimization. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/c0749c39aaff9e9e4c91f7118bf21b1e-Abstract-Conference.html.

RoboCasa. RoboCasa365 checkpoints. Hugging Face model release, 2026. URL https://huggingface.co/robocasa/robocasa365_checkpoints/tree/main/pi05_pretrain_human300/multitask_learning/75000. Human300 multitask checkpoint at step 75,000.

Omar Shaikh, Michelle S. Lam, Joey Hejna, Yijia Shao, Hyundong Cho, Michael S. Bernstein, and Diyi Yang. Aligning language models with demonstrated feedback. In *International Conference on Learning Representations*, pp. 20498–20525, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/349a45f211fb1b3850da1ccd829e869e-Abstract-Conference.html.

Andrew Wagenmaker, Mitsuhiko Nakamoto, Yunchu Zhang, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering Your Diffusion Policy with Latent Space Reinforcement Learning. In *Proceedings of The 9th Conference on Robot Learning*, volume 305, pp. 258–282, 2025. URL https://proceedings.mlr.press/v305/wagenmaker25a.html.

Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam, Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion model alignment using

direct preference optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8228–8238, 2024. doi: 10.1109/CVPR52733.2024.00786. URL https://openaccess.thecvf.com/content/CVPR2024/html/Wallace_Diffusion_Model_Alignment_Using_Direct_Preference_Optimization_CVPR_2024_paper.html.

Yihao Wu, He Zhang, Junbo Tan, Xueqian Wang, and Zhengyou Zhang. FlowPRO: Reward-free reinforced fine-tuning of flow-matching VLAs via proximalized preference optimization, 2026. URL https://arxiv.org/abs/2606.05468.

Wenke Xia, Yichu Yang, Hongtao Wu, Xiao Ma, Tao Kong, and Di Hu. Human-assisted robotic policy refinement via action preference optimization. In *Advances in Neural Information Processing Systems*, volume 38, pp. 36746–36768, 2025. doi: 10.52202/085713-1235. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/347110fb894281e5e937f6ccd998a6eb-Abstract-Conference.html.

Chaofan Zhang, Peng Hao, Xiaoge Cao, Xiaoshuai Hao, Shaowei Cui, and Shuo Wang. VTLA: Vision-tactile-language-action model with preference learning for insertion manipulation. *Biomimetic Intelligence and Robotics*, pp. 100333, 2026a. doi: 10.1016/j.birob.2026.100333. URL https://doi.org/10.1016/j.birob.2026.100333.

Zijian Zhang, Kaiyuan Zheng, Zhaorun Chen, Joel Jang, Yi Li, Siwei Han, Chaoqi Wang, Mingyu Ding, Dieter Fox, and Huaxiu Yao. GRAPE: Generalizing robot policy via preference alignment. In *IEEE International Conference on Robotics and Automation*, 2026b. URL https://arxiv.org/abs/2411.19309.

Fangqi Zhu, Zhengyang Yan, Zicong Hong, Quanxin Shou, Xiao Ma, and Song Guo. WMPO: World model-based policy optimization for vision-language-action models. In *International Conference on Learning Representations*, pp. 62486–62502, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/6605c7d1c1ae1521b328d036df5be1ae-Abstract-Conference.html.

A EXPERIMENTAL DETAILS

This appendix specifies the experimental instantiation of TS-CFG and the controls used in Section 4. Table 6 collects the numerical settings.

Table 6: Implementation and evaluation settings. Checkpoint 5,000 follows the existing benchmark-specific loop convention: 5,000 updates for RoboTwin and 5,001 iterations indexed 0–5,000 for RoboCasa.

Setting	RoboTwin	RoboCasa
Policy release	Motus $\pi_{0.5}$ RoboTwin2	RoboCasa $\pi_{0.5}$ human300 (75k)
Trainable LoRA modules	Action expert	Language backbone + action expert
LoRA rank / α	Action: 32 / 32	Language: 16 / 16; action: 32 / 32
Post-training data	50 tasks; clean50 + randomized500 per task	Human pretraining data for 34 tasks
Task sampling	Uniform task, then indexed window	Dataset weights proportional to size$^{0.4}$
Training seed	31	42
Global batch / accelerators	8 / 4	8 / 4
Checkpoint evaluated	5,000	5,000
Learning rate	5×10^{-7}, constant	100-step warmup to 5×10^{-7}, then constant
Optimizer	AdamW	AdamW
Adam $(\beta_1, \beta_2, \epsilon)$	$(0.9, 0.95, 10^{-8})$	$(0.9, 0.95, 10^{-8})$
Weight decay / gradient clip	0 / global norm 1	10^{-10} / global norm 1
Image resolution / cameras	224×224 / 3	224×224 / 3
Action horizon / model width	32 / 32	50 / 32
Physical control dimensions	14	12
Action normalization	Released quantile statistics	Released mean/std statistics
Inversion / reconstruction	10 / 10 Euler steps	10 / 10 Euler steps
Temporal offset	5 frames	5 frames
DPO β	2	2
Rollouts per task	50	50
Environment / policy seed	0 / 0	7 / 0
Executed chunk rows (zero-based)	1–15	0–4
Deployment integration steps	10	10

A.1 DATA AND PREPROCESSING

RoboTwin. We use the ALOHA-AgileX clean and randomized demonstration sets for all 50 tasks. Each observation contains the head camera, left wrist camera, right wrist camera, joint state, and an episode instruction. The training action target begins at the observed frame and contains 32 consecutive recorded actions. We retain the released policy's ALOHA coordinate conversion, state-relative joint-action representation, quantile normalization, and padding to 32 channels. Training samples tasks uniformly and then samples a valid indexed window within the selected task; all 550 demonstrations per task remain available. The instruction is drawn deterministically from the episode's seen-instruction list using the frame index and training seed. Consequently, original and temporal contexts can use different paraphrases of the same task.

RoboCasa. We use the released human-pretrained $\pi_{0.5}$ checkpoint at 75,000 steps. Our additional updates use only the pretraining-split human demonstrations of the 34 tasks listed below, rather than the target-split demonstration collection. The observation combines left and right agent views, a wrist view, the robot state, and the recorded task instruction. The 12-dimensional control representation comprises end-effector, gripper, base-motion, and control-mode components and is padded to width 32 before applying the released mean/std statistics. Action chunks contain 50 steps. The dataset

mixture uses weights proportional to each constituent dataset's size raised to 0.4, retaining the existing trajectory/frame sampling procedure. Target evaluation includes the 34 overlapping task identities and 16 additional composite tasks; overlap in task identity does not mean that target-split demonstrations are used in post-training.

The RoboCasa post-training tasks are:

> CloseBlenderLid, CloseFridge, CloseToasterOvenDoor, CoffeeSetupMug, NavigateKitchen, OpenCabinet, OpenDrawer, OpenStandMixerHead, PickPlaceCounterToCabinet, PickPlaceCounterToStove, PickPlaceDrawerToCounter, PickPlaceSinkToCounter, PickPlaceToasterToCounter, SlideDishwasherRack, TurnOffStove, TurnOnElectricKettle, TurnOnMicrowave, TurnOnSinkFaucet, DeliverStraw, GetToastedBread, KettleBoiling, LoadDishwasher, PackIdenticalLunches, PreSoakPan, PrepareCoffee, RinseSinkBasin, ScrubCuttingBoard, SearingMeat, SetUpCuttingStation, StackBowlsCabinet, SteamInMicrowave, StirVegetables, StoreLeftoversInBowl, and WashLettuce.

Training images and temporal contexts. Images are resized with padding to 224×224. We retain the base training preprocessing: a 95% crop resized back to the input resolution and a rotation within $\pm 5^\circ$ for the non-wrist image slot, followed by brightness, contrast, and saturation jitter with strengths 0.3, 0.4, and 0.5. The original observation is augmented once and reused for all four preference-loss evaluations. Negative generation uses evaluation-mode preprocessing; the explicit corruption controls described below are applied only to their alternative conditioning branch. The temporal context supplies the observations, state, and associated instruction at the chosen frame. We select the frame five steps earlier in the same episode, or five steps later when the earlier frame is unavailable. If neither is available, the loader uses another nearby frame: the closest available distinct frame in RoboTwin or an adjacent frame in RoboCasa. The offset ablations use only examples with a context at the specified distance.

A.2 Optimization and numerical construction

Policy updates. For RoboTwin, rank32 LoRA with $\alpha = 32$ is applied to the action expert's attention and MLP projections, for 13,860,864 trainable parameters. The adapters initialize their first factor from a zero-mean Gaussian with standard deviation 0.01 and their second factor to zero. For RoboCasa, LoRA is applied to attention and MLP projections in both the language backbone (rank 16, $\alpha = 16$) and the action expert (rank 32, $\alpha = 32$). The original model weights remain frozen. Within each benchmark, BC, DITTO*, VTLA*, TS-CFG, and TS-CFG + LN use this same adapter scope. Optimization uses AdamW with the settings in Table 6 and a batch of two examples per accelerator. We use the final checkpoint of the fixed budget for each reported configuration. For every preference pair, one sampled time and one Gaussian tensor are reused across both actions and both models.

Pair generation. Inversion evaluates the original-context field at times $1 - j/10$, $j = 0, \ldots, 9$, with step size -0.1. Reconstruction evaluates the mixed field at times $k/10$, $k = 0, \ldots, 9$, with step size $+0.1$. The temporal coefficient $\lambda_k = 1 - k/9$ is therefore one at the first reconstruction evaluation and zero at the last. The same temporal context is retained for all ten reconstruction steps. Both fields are evaluated at the same intermediate action tensor and time. Negatives are generated by the frozen starting policy on each training iteration and detached before the policy update. No candidate bank or environment evaluation is used for TS-CFG's pairs.

Action tensor and loss reduction. Inversion, reconstruction, and Gaussian perturbations operate on all $H \times 32$ coordinates. Generated values in padded channels are retained; they are neither overwritten from the demonstration nor reset to zero. The MSE is averaged across all horizon positions and channels before forming the reference-relative logistic loss, which is then averaged over examples. Training times follow $\tau = 0.999(1 - b)$, with $b \sim \mathrm{Beta}(1.5, 1)$, following the base flow-model recipe in our noise-to-action time convention. The sampled error receives unit weight. The released implementation uses the opposite time direction: our velocity $v_\phi(x, \tau, c)$ is its output at time $1 - \tau$, multiplied by -1. This change of notation preserves the generated actions and squared flow errors.

Evaluation protocol. The policy uses its ordinary ten-step flow sampler. RoboTwin starts from environment seed zero and uses unseen instructions, with the evaluator selecting valid episodes until 50 rollouts are recorded for each task. RoboCasa uses target-split kitchens, episode seeds 7–56, and

task-specific episode horizons. Policy sampling starts from seed zero in both benchmarks; Table 6 specifies the executed action chunks.

A.3 Definitions of the ablation controls

Random initialization. RandInit replaces the inverted demonstration latent with $z \sim \mathcal{N}(0, I)$ and applies the same temporal guidance as TS-CFG. This tests whether guidance remains useful when generation no longer starts from a latent tied to the demonstrated action.

Direct action perturbations. ActNoise adds Gaussian noise directly to the demonstrated action: $a^- = a^+ + \sigma_a \xi$, with $\xi \sim \mathcal{N}(0, I)$. ActMix interpolates it with the action chunk $a^+_{d(i)}$ recorded at the temporal context: $a^- = (1-\rho)a^+ + \rho a^+_{d(i)}$. We set their strengths to match TS-CFG's average squared deviation from the demonstration, so these controls compare perturbation directions at a matched average magnitude. Using 500 training examples, ten per task, we compute

$$R^2 = \mathbb{E}_{\text{cal}} \operatorname{MSE}(a^-_{\text{TS-CFG}}, a^+), \qquad Q^2 = \mathbb{E}_{\text{cal}} \operatorname{MSE}(a^+_{d(i)}, a^+), \tag{12}$$

$$\sigma_a = \sqrt{R^2} = 0.04076783, \qquad \rho = \sqrt{R^2/Q^2} = 0.47214542. \tag{13}$$

The expectation is over this calibration set; the resulting σ_a and ρ are held fixed for all training examples. Both operations use the full normalized, padded action tensor without clipping. In ActMix, each chunk retains the dataset's action representation: joint displacements are measured from the robot state at that chunk's own starting frame before normalization.

Temporal selection and strength. The gap controls change the recorded-frame offset to one, three, or ten while retaining all other default settings. The random-context control samples uniformly from the other indexed frames of the same episode and fixes that choice for the entire reconstruction. The amplitude controls use $\lambda_k = L(1 - k/9)$ with $L \in \{0.50, 0.75, 1.00\}$. The constant-guidance CFGI variants instead use $\lambda_k \equiv \lambda$ with $\lambda \in \{0.10, 0.25, 0.50\}$ at every reconstruction step. For the latent perturbation control,

$$\widetilde{z}_i = z_i + \sigma_z \xi, \qquad \xi \sim \mathcal{N}(0, I), \qquad x_i^{(0)} = \widetilde{z}_i, \tag{14}$$

where $\sigma_z \in \{0, 0.05, 0.10\}$ is measured in model latent coordinates. Noise is added once after inversion, without rescaling the latent or adding noise at later integration steps. The preference expectation also averages over this draw, independently of the Gaussian used in the flow-training loss. The LN-only control uses $\sigma_z = 0.05$ but sets $\lambda_k = 0$ for every reconstruction step, so both inversion and reconstruction are conditioned on c_i alone. The positive action and preference update are unchanged.

Instruction and image perturbations. These controls replace the temporal context with a corrupted original context while retaining original-context inversion and the decreasing guidance schedule. For text, we uniformly select 25%, 50%, or 75% of eligible instruction tokens, rounded to the nearest integer, replace their token IDs with zero, and remove them from the attention mask. State and prompt-template tokens are preserved. For images, we modify each present camera view after resizing and padding to 224×224; mask percentages refer to this entire image area. Scattered masking zeros 26 uniformly selected cells of a 16×16 grid, with each cell covering 14×14 pixels, approximately 10% of the canvas. The 25% cutout zeros a contiguous 8×8 block of these cells at a sampled valid grid location. Blur uses a Gaussian with standard deviation two pixels and a 13×13 kernel. Masked pixels are filled with mid-gray (zero in the normalized $[-1, 1]$ RGB representation). Each corruption is sampled once per negative and kept fixed throughout reconstruction; the policy's evaluation inputs remain uncorrupted.