# RESHAPING ACTION FLOW: BEHAVIOR CORRECTION FOR FLOW-MATCHING VISION-LANGUAGE-ACTION POLICIES

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Vision-language-action (VLA) models have shown remarkable capability across diverse manipulation tasks. After deployment, however, a pretrained policy may encounter situations unseen during training, such as an unexpected obstacle that requires a change in its usual behavior. Existing approaches either intervene at inference time, leaving the policy unchanged, or fine-tune it on new demonstrations, which can disrupt its original capabilities. In this paper, we propose a localized behavior correction framework for flow-matching VLA policies. Given correction demonstrations, our framework learns the desired correction through a guided velocity target while preserving the policy's original competence through retain-subspace gradient projection. We further adapt the preservation strength across the correction trajectory, applying stronger protection to frames that induce larger corrective updates and are more likely to interfere with the original capabilities. Extensive experiments on robot manipulation demonstrate that our method effectively corrects deployed policies while preserving their original competence, with adaptive preservation further improving retention.

## 1 INTRODUCTION

Vision-language-action (VLA) models now perform a broad range of manipulation tasks from visual observations and language instructions (Brohan et al., 2023; Open X-Embodiment Collaboration et al., 2024; Kim et al., 2024; Gemini Robotics Team et al., 2025), with recent flow-matching policies extending this competence to precise, high-frequency continuous control (Black et al., 2025b; Bjorck et al., 2025; Black et al., 2025a). Yet a policy can perform a task reliably and still violate a requirement it was never trained to satisfy. For example, even when explicitly informed that a nearby candle is lit, a policy may follow its usual direct trajectory and pass dangerously close to the flame (Zhang et al., 2026a). The problem is not simply to teach a new task, but to make an existing behavior change when a new execution condition calls for it.

One way to meet these new execution requirements is to intervene at runtime. Some approaches filter or override the policy's predicted actions using external mechanisms (Ames et al., 2017; Hsu et al., 2024), such as the control barrier function (CBF)-based corrections in AEGIS (Hu et al., 2026) and Knows (Park et al., 2026). Inference-time steering instead re-ranks the policy's sampled actions (Nakamoto et al., 2024), guides its sampling (Liu et al., 2026) or its sampling noise through a separately trained policy (Wagenmaker et al., 2025), or shifts its internal activations (Häon et al., 2025; Miao et al., 2026) toward a desired behavior. In both cases, the correction must be reapplied at deployment rather than acquired as a persistent update to the policy. This motivates updating the policy itself to satisfy the new requirement while preserving its existing task competence.

Correction demonstrations can illustrate the desired execution, providing supervision for fine-tuning the deployed policy. Fitting these demonstrations alone, however, places no explicit constraint on other contexts and may degrade previously acquired capabilities (Luo et al., 2025; Qi et al., 2024; Hancock et al., 2026). Existing approaches mitigate such forgetting by merging fine-tuned weights toward the pretrained ones with a global coefficient (Yadav et al., 2026), or by scaling each training update according to the model's current loss (Zhang et al., 2026b). These methods primarily regulate the extent of parameter updates without explicitly protecting directions associated with retained



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

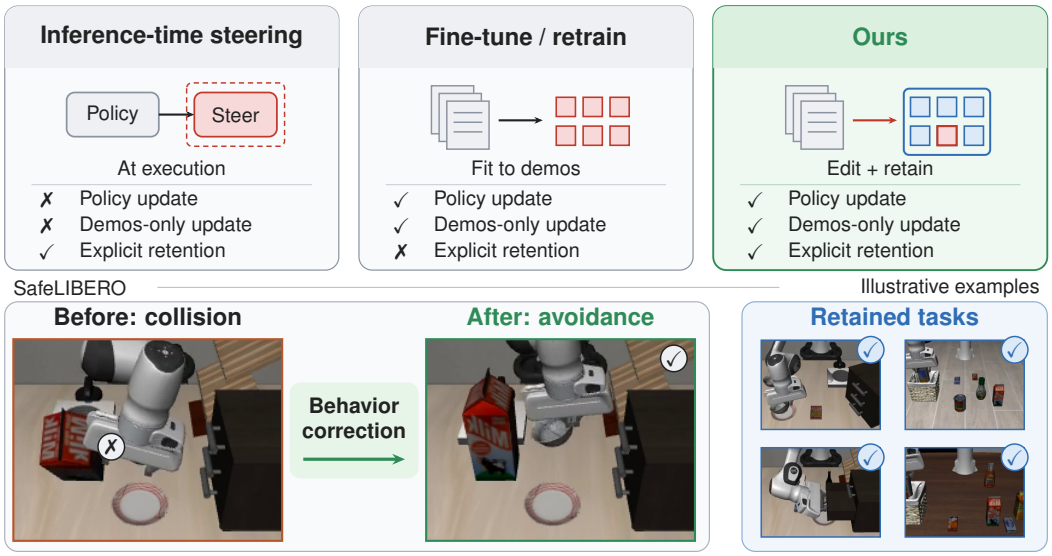


Figure 1: **Editing a deployed flow-matching VLA.** Inference-time steering corrects the behavior at execution without updating the policy, and fine-tuning on demonstrations of the safe behavior acquires the avoidance but does not explicitly retain the competence outside the edit. Ours steers the predicted velocity toward the demonstrated one and constrains the update so that the directions the rest of the policy depends on are removed from the gradient. On SafeLIBERO, the edited policy avoids the obstacle it collided with before and keeps its original tasks. Panels are illustrative; measured results are reported in Table 1.

behavior. A different line of work, VLA unlearning, combines targeted behavior suppression with retain objectives (Ranjan & Polyzou, 2026), but suppression alone does not specify what the policy should do instead. Going beyond suppression, concurrent work redirects a policy toward a desired behavior mode through a persistent weight update, but obtains the redirection signal from an auxiliary classifier trained beforehand (Wang et al., 2026). The challenge is therefore to achieve localized behavioral correction by learning the demonstrated change while preserving task competence wherever the correction does not apply.

In this paper, we propose a localized behavior reshaping framework for flow-matching VLAs that learns demonstrated corrections while preserving existing task competence. Inspired by guidance-based concept erasure (Gandikota et al., 2023), we construct a target velocity by adding to the frozen base policy's prediction a scaled difference between the demonstration-defined velocity and that prediction. This extrapolation amplifies the correction direction, and we train the policy to predict the resulting target. To support locality, we construct a retain subspace from the retain dataset and use gradient projection to limit updates along its directions. Different frames of a correction trajectory, however, may require different degrees of change, with larger required changes inducing larger updates. We therefore adapt the preservation strength across frames, applying stronger retain-subspace protection to larger corrective updates to reduce their potential interference with retained competence, while keeping the overall preservation budget fixed. Together, flow reshaping and adaptive preservation allow the policy to learn the demonstrated corrections while limiting disruption to its existing task competence.

We now summarize our contributions as follows:

- We formulate *localized behavioral correction* for flow-matching VLA policies, where a policy must acquire a demonstrated behavioral change while preserving the original capabilities that should remain unchanged.
- We propose a guided velocity target that extrapolates beyond the demonstrated velocity away from the frozen base policy's prediction, amplifying the correction direction for learning from correction demonstrations.

- We introduce adaptive retain-subspace projection, which protects directions associated with retained competence and applies stronger protection to larger corrective updates to reduce their potential interference with the policy's original capabilities.

## 2 Related Work

**Adapting VLA policies.** Existing approaches modify policy behavior at inference time or through parameter updates. Inference-time steering (Ho & Salimans, 2022; Schramowski et al., 2023; Turner et al., 2023; Li et al., 2023; Zou et al., 2023) computes a correction to a model's activations or outputs and applies it during generation. Shielding and safe-VLA methods (Alshiekh et al., 2018; Hu et al., 2026) similarly intervene during execution rather than changing the policy. Training-time methods instead update the policy itself. Preference-based fine-tuning and safe RL (Zhang et al., 2024; García & Fernández, 2015; Achiam et al., 2017; Tang et al., 2026; Zhang et al., 2025) use preference, reward, or cost signals to shape behavior. VLA-Forget (Ranjan & Polyzou, 2026) suppresses specified behaviors while using retain objectives to limit changes to other capabilities. Our work uses correction demonstrations to specify the desired execution. To incorporate this supervision into the policy, we draw on guidance-based model modification (Gandikota et al., 2023), which fine-tunes a model against modified prediction targets. We construct these targets from the demonstration-defined velocity and the frozen base policy's predicted velocity.

**Preserving task competence.** Continual learning (Kirkpatrick et al., 2017; Li & Hoiem, 2018; Rebuffi et al., 2017; Zeng et al., 2019; Farajtabar et al., 2020; Saha et al., 2021; Wang et al., 2021; Qiao et al., 2025; Saha & Roy, 2023) studies how models acquire new knowledge while retaining previously learned capabilities. Regularization methods penalize changes to parameters or outputs associated with earlier tasks (Kirkpatrick et al., 2017; Li & Hoiem, 2018), while replay methods revisit earlier examples during training (Rebuffi et al., 2017). Projection methods constrain updates by limiting gradient components along directions associated with earlier data (Zeng et al., 2019; Farajtabar et al., 2020; Saha et al., 2021; Wang et al., 2021; Qiao et al., 2025). Scaled projection further adjusts protection according to the estimated importance of retained directions (Saha & Roy, 2023). We build on gradient projection to preserve task competence during behavioral reshaping. A correction may require substantial changes in some action chunks while leaving others largely unchanged, with larger required changes inducing larger corrective updates. We therefore adapt retain-subspace protection across the correction trajectory, applying stronger protection to larger corrective updates to reduce their potential interference with retained competence.

## 3 Method

### 3.1 Problem Formulation

We first define the setting and notation of our method. A pretrained flow-matching VLA policy can behave undesirably when it encounters execution requirements not covered by its training (e.g., a policy trained without obstacles may collide with them when they block the way). We aim to reshape such behavior with demonstration trajectories $\mathcal{D}_{\text{edit}} = \{\xi^{(n)}\}_{n=1}^{N}$ of the desired behavior, as detailed in Section 3.2. Each trajectory $\xi = \{(o_t, s_t, l, a_t)\}_{t=1}^{T}$ records a camera observation $o_t$, proprioceptive state $s_t$, text instruction $l$, and action $a_t$ at time step $t$. The 7-DoF action $a_t = [a_t^{\text{pos}}, a_t^{\text{ori}}, a_t^{\text{grip}}]$ consists of a position change $a_t^{\text{pos}} \in \mathbb{R}^3$, an orientation change $a_t^{\text{ori}} \in \mathbb{R}^3$, and a binary gripper command $a_t^{\text{grip}} \in \mathbb{R}^1$. For training, we sample chunks $A_t = (a_t, \ldots, a_{t+H-1}) \in \mathbb{R}^{H\times 7}$ of $H$ consecutive actions together with their corresponding $(o_t, s_t, l)$. The reshaping procedure must also preserve the policy's original task competence in contexts where the correction does not apply. For this purpose, we use original training data $\mathcal{D}_{\text{retain}}$ as a preservation reference, as discussed in section 3.3 and section 3.4.

### 3.2 Behavior Reshaping with Guided Velocity Targets

As depicted in Figure 2, the frozen vision-language backbone encodes the camera observation $o_t$ and text instruction $l$ into context embeddings $\boldsymbol{c}_t = (\boldsymbol{c}_{t,1}, \ldots, \boldsymbol{c}_{t,L})$, $\boldsymbol{c}_{t,i} \in \mathbb{R}^d$. The flow-matching action module, parameterized by $\theta_0$, takes $\boldsymbol{c}_t$ and the proprioceptive state $s_t$ and generates a chunk of

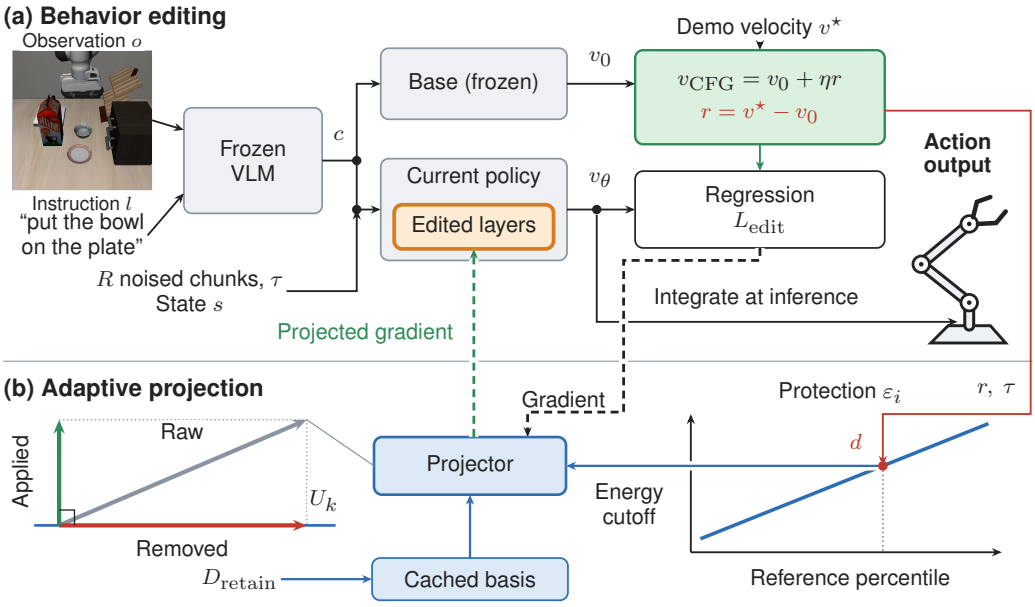


Figure 2: **Post-deployment editing of a flow-matching VLA.** *(a)* The deployed policy is frozen except for a few linear layers of the action module (for GR00T, the cross-attention key and value projections $\boldsymbol{W}_K, \boldsymbol{W}_V$, through which the context $\boldsymbol{c}$ enters action generation). The edit regresses the policy's velocity onto the guided target $v_{\text{CFG}}$ built from the frozen base policy's prediction $v_0$ and the demonstrated velocity $v^\star = A - z$ (equation 1 and equation 2). *(b)* Before each optimizer step, every gradient is projected off the subspace that holds a fraction $\epsilon$ of retained-feature energy (equation 3), so updates along the protected retained directions are removed; the projection is implemented in the retain basis, where AdamW preserves the constraint (section 3.3, section B.3). The protected fraction is set per frame: the percentile of a frame's discrepancy $d$ in the reference sets its threshold $\epsilon_i$ (equation 5); the weights average $\frac{1}{2}$, so the mean threshold stays at $\epsilon_0$ over the reference.

consecutive actions starting at $a_t$ by integrating its velocity field $v_{\theta_0}(A_\tau, \tau, \boldsymbol{c}_t, s_t)$ from noise (Black et al., 2025b; Bjorck et al., 2025). To reshape the action module's behavior, we adopt a guided velocity target inspired by classifier-free guidance (Ho & Salimans, 2022) and guidance-based concept erasure (Gandikota et al., 2023; Gao et al., 2025). We drop the subscript $t$ from here on for convenience:

$$v_{\text{CFG}}(A_\tau, \tau, \boldsymbol{c}, s) \;=\; v_{\theta_0}(A_\tau, \tau, \boldsymbol{c}, s) + \eta\,(v^\star - v_{\theta_0}(A_\tau, \tau, \boldsymbol{c}, s)). \tag{1}$$

Here the frozen base policy's prediction $v_0 = v_{\theta_0}(A_\tau, \tau, \boldsymbol{c}, s)$ is the reference to move away from, and $v^\star$ is the demonstration-defined velocity to move toward. The velocity $v^\star$ is built from the demonstrated chunk $A$, the noise $z$, and the flow time $\tau$ through the flow-matching interpolation $A_\tau$ (section A). The training loss for $\theta$, initialized from $\theta_0$, is thus

$$\mathcal{L}_{\text{edit}}(\theta) \;=\; \mathbb{E}_{\xi\sim\mathcal{D}_{\text{edit}},\,t}\;\mathbb{E}_{z,\tau}\,\big\|\,v_\theta(A_\tau, \tau, \boldsymbol{c}, s) \;-\; v_{\text{CFG}}(A_\tau, \tau, \boldsymbol{c}, s)\big\|_2^2. \tag{2}$$

Here $\eta = 0$ matches the frozen base policy's predictions on sampled inputs from correction demonstrations, and $\eta = 1$ is flow-matching behavior cloning on the demonstrations. A larger $\eta > 1$ pushes the target further beyond the demonstrated velocity along the correction direction $r = v^\star - v_0$. This objective specifies the desired correction but does not explicitly constrain its effect on the policy's original task competence.

### 3.3 Behavior Preservation with Adaptive Projection

When reshaping the policy with equation 2, we need to keep its original ability. We constrain the weight update with Gradient Projection Memory (GPM) (Saha et al., 2021), so that it has little effect on the retain set $\mathcal{D}_{\text{retain}}$, and we adjust the strength of this protection for each frame of the demonstrations $\mathcal{D}_{\text{edit}}$ according to how far the frozen policy is from the demonstration at that frame. We call this *locality gating*.

For a linear layer, GPM builds an orthonormal basis $\boldsymbol{U} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \dots]$ of its inputs on $\mathcal{D}_{\text{retain}}$, with a singular value $\lambda_i$ of the retained inputs' second-moment matrix (section B.1) for each basis vector $\boldsymbol{u}_i$, sorted as $\lambda_1 \geq \lambda_2 \geq \cdots$. It then constrains the weight update $\Delta \boldsymbol{W}$ so that the layer's output stays nearly unchanged on the retain data, i.e., $\Delta \boldsymbol{W}\, \boldsymbol{x} \approx 0$ for retained inputs $\boldsymbol{x}$. A larger $\lambda_i$ means more energy of the retained inputs along $\boldsymbol{u}_i$, so a weight change along $\boldsymbol{u}_i$ affects the retained tasks more. A threshold $\epsilon \in (0, 1)$ keeps the fewest leading directions reaching a fraction $\epsilon$ of the total energy,

$$\boldsymbol{U}_k = [\, \boldsymbol{u}_1, \dots, \boldsymbol{u}_{k(\epsilon)} \,], \qquad k(\epsilon) = \min\Big\{k : \textstyle\sum_{i \leq k} \lambda_i \big/ \sum_i \lambda_i \geq \epsilon\Big\}. \tag{3}$$

A larger $\epsilon$ thus protects more directions. Projecting the gradient of $\boldsymbol{W}$ directly keeps $\Delta \boldsymbol{W}\, \boldsymbol{U}_k = 0$ only under plain gradient descent and not under an adaptive optimizer such as AdamW, so we project the gradient in the basis $\boldsymbol{U}$ instead: we write the edited weight as

$$\boldsymbol{W} \;=\; \boldsymbol{W}_0 + \boldsymbol{C}\, \boldsymbol{U}^\top,$$

train $\boldsymbol{C}$ from zero, and zero the first $k$ columns of its gradient. Then $\Delta \boldsymbol{W}\, \boldsymbol{U}_k = \boldsymbol{C}\, \boldsymbol{U}^\top \boldsymbol{U}_k = 0$ holds exactly under any optimizer that updates parameters elementwise (Figure 2b; section B.3).

We next bring locality into this protection, since not every frame of $\mathcal{D}_{\text{edit}}$ asks the policy for the same change. A frame where the frozen policy is far from the demonstration brings a larger correction than one where it is close. Therefore, we give the frames of larger correction a higher $\epsilon$ and the rest a lower one, so that larger updates are held to a more protective subspace. We measure the correction of a frame as a scalar discrepancy $d$, the distance in action space between the demonstrated action and the action that the frozen policy predicts (section B.2). For each frame, we compare its discrepancy $d$ with those of the frames in $\mathcal{D}_{\text{edit}}$, and record its percentile $p \in (0, 1)$. We give each frame a weight according to its percentile:

$$w \;=\; \Phi\big(\Phi^{-1}(p)/\kappa\big), \tag{4}$$

with a hyperparameter $\kappa$ and the CDF $\Phi$ of the standard normal distribution. Akin to the logit-normal timestep density of rectified flows (Esser et al., 2024), we use the CDF of the standard normal distribution to reshape the distribution of the weights while keeping their relative order. Setting $\kappa = 1$ recovers $w = p$, $\kappa < 1$ pushes the weights toward 0 and 1, and $\kappa > 1$ concentrates them near $1/2$ (Figure 3, right). Each frame $i$ in a batch then receives its own threshold parameter for equation 3,

$$\epsilon_i \;=\; \epsilon_0 \;+\; \beta\,\big(\tfrac{1}{2} - w_i\big), \tag{5}$$

with $\beta$ the swing (Figure 2b). The mean of $w$ over $\mathcal{D}_{\text{edit}}$ is $1/2$ for every $\kappa$, so the mean of $\epsilon_i$ stays at $\epsilon_0$, and $\beta < 0$ gives the frames of large correction the higher threshold.

### 3.4 Learning Strategy and Inference

We edit a few linear layers of the action module and keep the rest of the policy frozen, including the backbone and all biases (section B). Before editing, we prepare two things with the frozen policy. For each edited layer, we build the basis $\boldsymbol{U}$ and the rank function $k(\epsilon)$ of equation 3 from the layer's inputs on $\mathcal{D}_{\text{retain}}$ data. For the gate, we compute the discrepancies on a sample of frames in $\mathcal{D}_{\text{edit}}$ (section B.2), and we take a frame's percentile during editing relative to these values. Each editing step then encodes a batch of demonstration frames once with the frozen backbone and evaluates the base model $v_{\theta_0}$ and the current policy $v_\theta$ on the same $R$ noised chunks of each frame. Each frame receives its own threshold $\epsilon_i$ from equation 5, hence its own rank $k_i = k(\epsilon_i)$, and its gradient reaches only the columns $j > k_i$ of $\boldsymbol{C}$.

At inference time, the edited policy runs exactly as the base policy did. We store $\boldsymbol{W}_0 + \boldsymbol{C}\, \boldsymbol{U}^\top$ as an ordinary weight, so the edited policy needs no guidance at sampling time, no extra forward pass, and no change to the sampler. Beyond the optimizer's settings, our method has four hyperparameters, $\eta$, $\epsilon_0$, $\beta$, and $\kappa$. Their values, the benchmark, the data, the evaluation splits, and the metrics are given in section 4.

## 4 Experiments

### 4.1 Setup

**Implementation details.** We apply our method to two flow-matching VLAs, QwenGR00T and $\pi_{0.5}$ (Black et al., 2025a). QwenGR00T uses Qwen3-VL 2B (Bai et al., 2025) as the vision-language

Table 1: **Performance on SafeLIBERO (Level I) and VLA-Arena (L0).** S∧S: successful and collision-free; CC: cumulative cost on the Safety category; Dist., Extr., Long: the Distractor, Extrapolation, and Long Horizon categories; H: harmonic mean of Retain and the edit score (section 4.1); ↑ (↓): higher (lower) is better. † The method does not act on these tasks, so the value equals Base's: AEGIS and COAST act at inference time and are not applied on LIBERO, and AEGIS is run only on the five VLA-Arena safety suites (section C). Within each model, the best result in each column is in bold and the second best is underlined, excluding Base and the values marked †.

| | | SafeLIBERO | | | | | VLA-Arena | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Model** | **Method** | **SR ↑** | **Safe ↑** | **S∧S ↑** | **Retain ↑** | **H ↑** | **Safety ↑** | **CC ↓** | **Dist. ↑** | **Extr. ↑** | **Long ↑** | **Retain ↑** | **H ↑** |
| **QwenGR00T** | Base | 25.3 | 6.2 | 4.2 | 95.2 | 8.0 | 2.1 | 22.84 | 4.0 | 0.7 | 4.0 | 95.2 | 5.3 |
| | COAST | 25.9 | 5.9 | 4.0 | 95.2† | 7.7 | 1.7 | 22.81 | 2.0 | 1.3 | 3.7 | 95.2† | 4.3 |
| | AEGIS | 27.3 | 55.8 | 20.6 | 95.2† | 33.9 | 1.5 | 4.40 | 4.0† | 0.7† | 4.0† | 95.2† | 4.9 |
| | SFT | 65.3 | 55.5 | 47.2 | 74.9 | 57.9 | **58.5** | 4.39 | **74.3** | **20.4** | **86.7** | 57.4 | 58.7 |
| | ConSFT | **67.8** | 55.9 | **48.2** | 74.7 | 58.6 | 54.9 | **3.89** | 73.3 | 16.4 | 75.3 | 60.4 | 57.6 |
| | **Ours** | 63.8 | **57.3** | 46.3 | **85.3** | **60.0** | 44.9 | 6.38 | 64.3 | 17.8 | 74.0 | **78.5** | **61.3** |
| $\pi_{0.5}$ | Base | 51.7 | 8.8 | 7.7 | 96.8 | 14.2 | 20.8 | 29.72 | 23.0 | 2.7 | 41.3 | 96.8 | 35.8 |
| | COAST | 57.3 | 8.2 | 7.0 | 96.8† | 13.1 | 22.1 | 30.06 | 20.7 | 3.3 | 44.3 | 96.8† | 36.7 |
| | AEGIS | 64.8 | **60.8** | **45.8** | 96.8† | **62.2** | 20.7 | 13.88 | 23.0† | 2.7† | 41.3† | 96.8† | 35.7 |
| | SFT | 75.3 | 48.0 | 43.7 | 89.1 | 58.6 | 35.5 | 12.50 | 49.7 | 8.0 | **64.0** | 84.5 | 53.6 |
| | ConSFT | **76.8** | 46.0 | 42.8 | 91.0 | 58.2 | **38.4** | 13.13 | 47.7 | 6.0 | 58.0 | 90.9 | 53.1 |
| | **Ours** | 69.5 | 56.2 | 45.2 | **94.8** | 61.2 | 32.7 | **12.20** | **50.0** | **9.3** | 59.3 | **95.2** | **54.2** |

backbone with a flow-matching diffusion transformer (DiT) action module (Bjorck et al., 2025), and $\pi_{0.5}$ uses PaliGemma (Beyer et al., 2024) with an action expert. Both are initialized from public checkpoints pretrained on the LIBERO demonstrations (Liu et al., 2023). The backbone stays frozen, and we edit only a few linear layers of the action module: the key and value projections of the DiT cross-attention for QwenGR00T (25.2M parameters, 1.1% of the model), and the output projection of each action-expert attention layer for $\pi_{0.5}$ (37.7M parameters, 0.9% of the model). We set $\beta = -0.1$ and the gate shape $\kappa = 1$ throughout, $\eta = 1.1$ and $\epsilon_0 = 0.80$ for QwenGR00T on both benchmarks, and, for $\pi_{0.5}$, $\eta = 1.25$ and $\epsilon_0 = 0.80$ on SafeLIBERO and $\eta = 1.1$ and $\epsilon_0 = 0.60$ on VLA-Arena. Both models are edited for 2,000 steps on a single NVIDIA H200 GPU with AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.95$), gradient clipping at 1.0, and a learning rate warmed up linearly over the first 100 steps and then decayed by a cosine schedule from $10^{-4}$ to $10^{-6}$, with batch size 64 for QwenGR00T and 24 for $\pi_{0.5}$ (section B). Each edit takes under four hours, and the edited policy has the architecture and inference cost of the base policy (section 3.4). At inference, QwenGR00T integrates 4 Euler steps into a chunk of 8 actions and executes all of it, while $\pi_{0.5}$ integrates 10 steps into a chunk of 10 actions and replans after the first 5, as in their original configurations.

**Data, benchmarks, and metrics.** We evaluate on two benchmarks. SafeLIBERO (Hu et al., 2026) places obstacles near the manipulated objects of LIBERO tasks at increasing levels of difficulty, of which we use Level I. Its $\mathcal{D}_{\text{edit}}$ consists of 669 demonstrations that complete the goal, object, and spatial tasks without collision, collected by rolling out QwenGR00T and $\pi_{0.5}$ under AEGIS (Hu et al., 2026) and used to edit both models.[1] VLA-Arena (Zhang et al., 2026a) spans four categories (Safety, Distractor, Extrapolation, and Long Horizon), and its $\mathcal{D}_{\text{edit}}$ consists of the 3,000 demonstrations of its official Level 0 (L0) training set, covering every L0 task. On both benchmarks, $\mathcal{D}_{\text{retain}}$ is the original LIBERO (spatial, object, goal, and long), the training distribution of the base policies.

Each benchmark follows its standard per-episode setting (step limits, initial states, and success and cost definitions). We evaluate the goal, object, and spatial suites of SafeLIBERO with 50 episodes per task and every VLA-Arena L0 task with 30 episodes per task, and measure retention (Retain) on the four LIBERO suites with 50 episodes per task, as in OpenVLA (Kim et al., 2024). On SafeLIBERO, we report the success rate (SR), the collision-free rate (Safe), and the rate of episodes that both succeed and avoid collision (S∧S), which corresponds to the safe success recorded by the released AEGIS code (Hu et al., 2026). On VLA-Arena, we report the official SR, averaged

[1]The long suite is not used for editing: with obstacles in place, its tasks are rarely completed without collision, so too few collision-free demonstrations could be collected.

within each of the four categories, and the cumulative cost (CC) of safety violations per episode, averaged over the Safety suites since only they define a cost. Retain is the success rate on the original LIBERO. To measure whether a method achieves the edit and retention together, we report H, the harmonic mean of Retain and the edit score, which is S∧S on SafeLIBERO and the mean SR of the four categories on VLA-Arena. Per-suite results are provided in section C.

**Baselines.** We compare with COAST (Miao et al., 2026), which steers the frozen policy's hidden states at inference time; AEGIS (Hu et al., 2026), which filters the policy's actions at run time through its CBF-based safety layer; supervised fine-tuning (SFT) of the same layers on the same demonstrations as our method; and ConSFT (Zhang et al., 2026b), which builds on SFT and down-weights samples of high loss to limit forgetting.

### 4.2 Quantitative Evaluation

**SafeLIBERO.** We first evaluate whether our method corrects the obstacle-avoidance behavior on SafeLIBERO, where the base policies collide in most episodes (Table 1). COAST leaves S∧S near the base level, and AEGIS raises it to 20.6 on QwenGR00T and 45.8 on $\pi_{0.5}$ by filtering every action at run time. Fine-tuning corrects the behavior at a cost to the original tasks: SFT and ConSFT reach an S∧S of 47–48 on QwenGR00T and about 43 on $\pi_{0.5}$, but lose about 20 and 6–8 points of Retain. Our method reaches a comparable S∧S on QwenGR00T (46.3) and the highest S∧S of the fine-tuned methods on $\pi_{0.5}$ (45.2), while keeping Retain 10.4 and 5.7 points above SFT. It thus attains the highest H on QwenGR00T, and on $\pi_{0.5}$ an H of 61.2, above SFT and ConSFT and 1.0 point below AEGIS (62.2). Our S∧S rests on avoiding collisions rather than completing more tasks: among the fine-tuned methods, our method attains the highest Safe on both models (57.3 and 56.2) but the lowest SR (63.8 and 69.5). Our method keeps a higher Retain than SFT on every LIBERO suite (Table 8), with the largest gaps on LIBERO-Object for QwenGR00T (91.6 against 74.2) and on LIBERO-Long for $\pi_{0.5}$ (90.4 against 78.2). AEGIS obtains its S∧S by running VLM-based obstacle identification and its CBF-based safety layer at every step, and its successful collision-free rollouts form $\mathcal{D}_{\text{edit}}$; on $\pi_{0.5}$, the edited policy comes within 0.6 points of this S∧S (45.2 against 45.8) without either component at deployment. Section 4.3 examines which components account for these results.

**VLA-Arena.** To examine whether the method carries over to other scenarios, we further evaluate on VLA-Arena, whose L0 tasks all lie outside the training distribution of both base policies. Here the methods that leave the policy unchanged barely help. COAST fits its operator for a task only from rollouts in which the base policy both succeeds and fails, which the base policies rarely produce on VLA-Arena, so it is fitted on few tasks and changes little. AEGIS, run only on the safety suites, acts only on an obstacle named by the task, lowering CC without raising SR. Our method raises the mean SR over the four categories from 2.7 to 50.3 on QwenGR00T and from 22.0 to 37.8 on $\pi_{0.5}$ and lowers CC by about 70% and 60%. Its mean SR is lower than SFT's on QwenGR00T and close to it on $\pi_{0.5}$, but its Retain stays 21 and 11 points above SFT's, and on LIBERO-Long the gap grows to 29 points (58.0 against 29.2 on QwenGR00T), so it attains the highest H on both models. On $\pi_{0.5}$, it also attains the lowest CC and the highest Distractor and Extrapolation SR of all methods. Across both architectures and both benchmarks, our method thus reaches an edit score close to SFT's in most settings while preserving much more of the original competence, a favorable balance between correcting the behavior and retaining the original tasks.

### 4.3 Analysis

We analyze the components of our method with QwenGR00T on SafeLIBERO, using the training and evaluation setup of section 4.1 and changing one factor at a time (Table 2). Each group contains the main setting ($\eta = 1.1$, $\beta = -0.1$, $\kappa = 1$), which reaches an S∧S of 46.3, a Retain of 85.3, and an H of 60.0.

**Guidance strength.** Guidance strength trades task completion for safety. We vary $\eta$ in equation 1 from 1, where the target is the demonstrated velocity itself as in SFT, to 2, keeping the projection and the gate (Table 2(a)). A larger $\eta$ pushes the target further from the base prediction $v_0$ and raises Safe steadily, from 48.7 to 79.2, but SR peaks at $\eta = 1.1$ and collapses to 17.7 at $\eta = 2$: a

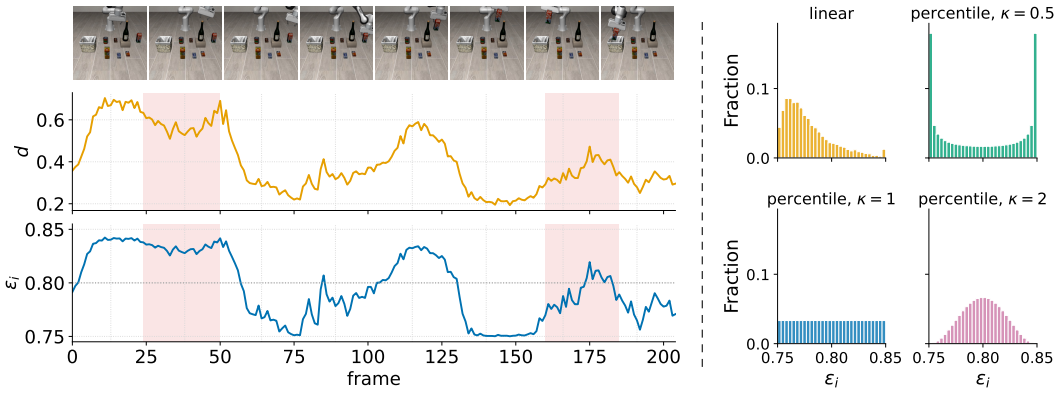


Figure 3: **Locality gating along a demonstration.** Left: frames of a SafeLIBERO demonstration (top), the discrepancy $d$ (equation 6) between the demonstrated action and the base policy's one-step estimate at each frame (middle), and the per-frame threshold $\epsilon_i$ the gate assigns (bottom); each frame's values are computed from the action chunk starting at that frame. Within the two shaded segments the discrepancy varies by a similar amount, but $\epsilon_i$ moves by different amounts (section 4.3). With $\kappa = 1$ the threshold is $\epsilon_i = \epsilon_0 + \beta(0.5 - p_i)$, where $p_i$ is the frame's percentile in the frozen reference, and with $\beta < 0$ the frames that ask for the largest change receive the tightest constraint. Right: the distribution of $\epsilon_i$ over the frozen reference under each gate of Table 2(c), the linear map of the discrepancy, rescaled between its 1st and 99th percentiles, and the percentile shaped by $\kappa$.

Table 2: **Analysis on SafeLIBERO with QwenGR00T.** Each group varies one factor and keeps the others at the main setting, shaded ($\eta = 1.1$, $\beta = -0.1$, $\kappa = 1$): (a) the guidance strength $\eta$ of equation 1; (b) the swing $\beta$ of equation 5, where $\beta = 0$ is a fixed $\epsilon$ and $\beta < 0$ tightens the constraint on frames of large discrepancy; (c) the gate, a linear map of the discrepancy or its percentile shaped by $\kappa$ (the full direction $\times$ $\kappa$ grid is in Table 4). Within each group, the best value of a column is in bold and the second best is underlined.

(**a**) Guidance strength $\eta$

| $\eta$ | SR ↑ | Safe ↑ | S∧S ↑ | Retain ↑ | H ↑ |
|---|---|---|---|---|---|
| 1.0 | <u>61.2</u> | 48.7 | 41.7 | **88.0** | 56.5 |
| 1.1 | **63.8** | 57.3 | **46.3** | <u>85.3</u> | **60.0** |
| 1.25 | 58.2 | 62.7 | <u>44.0</u> | 81.5 | <u>57.1</u> |
| 1.5 | 41.0 | <u>73.7</u> | 34.8 | 73.8 | 47.3 |
| 2.0 | 17.7 | **79.2** | 15.3 | 59.9 | 24.4 |

(**b**) Swing $\beta$

| $\beta$ | SR ↑ | Safe ↑ | S∧S ↑ | Retain ↑ | H ↑ |
|---|---|---|---|---|---|
| +0.1 | 62.8 | <u>58.2</u> | 44.2 | 76.0 | 55.9 |
| 0 | 62.5 | **60.2** | <u>46.0</u> | 80.6 | <u>58.6</u> |
| −0.05 | <u>64.7</u> | 55.0 | 44.5 | 84.2 | 58.2 |
| −0.1 | 63.8 | 57.3 | **46.3** | 85.3 | **60.0** |
| −0.2 | **65.5** | 51.5 | 43.7 | <u>87.2</u> | 58.2 |
| −0.3 | 61.8 | 50.0 | 43.5 | **87.7** | 58.1 |

(**c**) Locality gate shape

| Gate | SR ↑ | Safe ↑ | S∧S ↑ | Retain ↑ | H ↑ |
|---|---|---|---|---|---|
| linear | 63.0 | <u>55.0</u> | <u>44.3</u> | 84.1 | <u>58.0</u> |
| percentile, $\kappa = 0.5$ | **64.7** | 53.2 | 42.3 | **85.9** | 56.7 |
| percentile, $\kappa = 1$ | <u>63.8</u> | **57.3** | **46.3** | <u>85.3</u> | **60.0** |
| percentile, $\kappa = 2$ | 60.8 | 54.2 | 42.0 | 83.5 | 55.9 |

strong extrapolation drives the velocity field away from the demonstrations, and the robot avoids the obstacle without completing the task. Retain also falls with $\eta$, as a larger extrapolation asks for a larger update, so S∧S and H peak at a moderate $\eta = 1.1$, which we use for QwenGR00T.

**Swing direction and range.** Tightening the constraint on high-discrepancy frames improves retention. We vary the swing $\beta$ in equation 5 at a fixed $\epsilon_0$ and $\kappa = 1$; the mean of $\epsilon_i$ stays at $\epsilon_0$, and $\beta > 0$ loosens while $\beta < 0$ tightens the constraint on frames of large discrepancy (Table 2(b)). Retain rises steadily as $\beta$ goes from $+0.1$ to $-0.3$, from 76.0 to 87.7: the looser direction falls close to SFT's 74.9, while the main setting gains 4.7 points over a fixed $\epsilon$ at a comparable S∧S and attains

Table 3: **Ablation on SafeLIBERO with QwenGR00T.** Starting from the base policy, each row adds one component: SFT fits the demonstrated velocity, $\eta = 1.1$ extrapolates the target (**Ours** w/o projection, which also has no gate, since the gate sets the projection's threshold), the retain-subspace projection is added with a fixed $\epsilon$ (**Ours** w/o locality gating), and locality gating replaces the fixed $\epsilon$ (**Ours**). The best value of a column is in bold and the second best is underlined, excluding Base.

| Method | SR↑ | Safe↑ | S∧S↑ | Retain↑ | H↑ |
|---|---|---|---|---|---|
| Base | 25.3 | 6.2 | 4.2 | 95.2 | 8.0 |
| SFT | **65.3** | 55.5 | <u>47.2</u> | 74.9 | 57.9 |
| **Ours** w/o projection | 63.0 | **63.8** | **47.8** | 68.1 | 56.2 |
| **Ours** w/o locality gating | 62.5 | <u>60.2</u> | 46.0 | <u>80.6</u> | <u>58.6</u> |
| **Ours** | <u>63.8</u> | 57.3 | 46.3 | **85.3** | **60.0** |

the highest H. Tightening further costs collision avoidance, and S∧S drops. Along a demonstration, $\epsilon_i$ follows the discrepancy, rising above $\epsilon_0$ where the demonstration departs most from the base policy (Figure 3, left). We attribute the gain to the interplay between the edit target and the projection: on a high-discrepancy frame the target already moves the policy far from $v_0$, and a tighter constraint keeps this large update out of the directions the retained tasks use, while a low-discrepancy frame asks for a small change whose freedom costs little. This agrees with Zhang et al. (2026b), who argue that high-loss samples disproportionately amplify the forgetting risk; ConSFT shrinks their update, whereas our gate keeps their full target and restricts its directions in weight space.

**Locality gate shape.** The distribution of $\epsilon_i$ over frames shapes the trade-off. At the main swing ($\beta = -0.1$), we compare the percentile with a linear map of the discrepancy rescaled to $[0, 1]$, and reshape the percentile with $\kappa$ (Table 2(c)). The discrepancy is right-skewed over the reference, so the linear map gives most frames a small $p_i$, and the mean of $\epsilon_i$ falls below $\epsilon_0$ (Figure 3, right); the frames receive less protection on average, and H falls to 58.0. The percentile is uniform over the reference, which keeps the mean of $\epsilon_i$ at $\epsilon_0$ by construction and makes the gate independent of the scale of the discrepancy; along a demonstration, it moves $\epsilon_i$ by the fraction of reference frames a change in discrepancy crosses rather than by the change itself (Figure 3, left). Among the shapes, the uniform one ($\kappa = 1$) of the main setting attains the highest H: a U shape ($\kappa = 0.5$) pins frames at the two ends of the range and keeps Retain but loses Safe, while a bell shape ($\kappa = 2$) keeps most frames close to $\epsilon_0$.

**Ablation study.** Table 3 adds the components one at a time, starting from SFT. Extrapolating the target with $\eta = 1.1$ raises Safe by 8.3 points but costs another 6.8 points of Retain. The retain-subspace projection with a fixed $\epsilon$ raises Retain by 12.5 points, as it confines the update to directions the retained tasks do not use, and locality gating at the same mean $\epsilon$ adds 4.7 more. The projection and the gate thus each add retention while S∧S, the edit score, stays within 1.8 points, and together they raise H from 57.9 for SFT to 60.0, the highest of all variants.

## 5 Conclusion

We studied localized behavior correction for flow-matching VLA policies, where a deployed policy must acquire a demonstrated change in behavior while preserving the competence that the change does not concern. Our framework learns the correction through a guided velocity target that extrapolates beyond the demonstrated velocity, away from the frozen base policy's prediction, while preserving original competence by projecting updates away from a retain subspace built from the original task data. We further adapt the projection strength across correction frames, applying stronger protection to frames with larger base-to-demonstration discrepancies. Experiments with two flow-matching VLA policies on SafeLIBERO and VLA-Arena show that our method achieves correction performance close to supervised fine-tuning in most settings while retaining substantially more of the original competence, yielding a stronger edit–retention trade-off across both models and benchmarks. Future work will explore improved correction objectives that further increase task success while limiting interference with retained capabilities.

### AI USE STATEMENT

We used generative AI tools, namely a coding assistant based on a large language model, in the following ways. For research execution, it helped us implement the training, evaluation and job-orchestration code, provided feedback on the design of experiments such as hyper-parameter grids and evaluation protocols, and aggregated the evaluation logs. For the method, we derived that the column-masked eigenbasis parametrization enforces the retain projection exactly under element-wise optimizers such as AdamW and verified it numerically, and the assistant helped review the derivation. For data, the SafeLIBERO demonstrations are collected with AEGIS (Hu et al., 2026), whose obstacle identification relies on a vision-language model, and the collection scripts were written with the coding assistant. For writing, it translated the authors' drafts into English, helped turn the authors' notes and outlines into draft passages, suggested revisions, and checked spelling and grammar. For retrieval, it helped search the literature for related work. The authors reviewed and revised all AI-assisted content and take full responsibility for this paper.

### REPRODUCIBILITY STATEMENT

Section 4.1 states the models, edited layers, hyper-parameters, data, and evaluation protocol, including episode counts and evaluated levels, and section B gives the training noise sampling and the inference configuration of each model. The construction of the retain subspace and of the locality gate's reference is described in section B.1 and section B.2. Both base policies are public checkpoints pretrained on LIBERO, and all benchmarks are public simulators; the SafeLIBERO editing demonstrations are collected by the rollouts described in section 4.1. Per-suite results are in section C. The code will be released upon acceptance.

## REFERENCES

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pp. 22–31, 2017.

Mohammed Alshiekh, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. Safe reinforcement learning via shielding. In *AAAI Conference on Artificial Intelligence*, 2018.

Aaron D. Ames, Xiangru Xu, Jessy W. Grizzle, and Paulo Tabuada. Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control*, 62(8): 3861–3876, 2017. doi: 10.1109/TAC.2016.2638961.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, Thomas Unterthiner, Daniel Keysers, Skanda Koppula, Fangyu Liu, Adam Grycner, Alexey Gritsenko, Neil Houlsby, Manoj Kumar, Keran Rong, Julian Eisenschlos, Rishabh Kabra, Matthias Bauer, Matko Bošnjak, Xi Chen, Matthias Minderer, Paul Voigtlaender, Ioana Bica, Ivana Balazevic, Joan Puigcerver, Pinelopi Papalampidi, Olivier Henaff, Xi Xiong, Radu Soricut, Jeremiah Harmsen, and Xiaohua Zhai. PaliGemma: A versatile 3B VLM for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz,

Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. GR00T N1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: A vision-language-action model with open-world generalization. In *Conference on Robot Learning (CoRL)*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40, 2025a.

Kevin Black, Noah Brown, Danny Driess, et al. $\pi_0$: A vision-language-action flow model for general robot control. In *Robotics: Science and Systems (RSS)*, 2025b. arXiv:2410.24164.

Anthony Brohan, Noah Brown, Justice Carbajal, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 12606–12633, 2024.

Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 108 of *Proceedings of Machine Learning Research*, pp. 3762–3773, 2020.

Rohit Gandikota, Joanna Materzyńska, Jaden Fiotto-Kaufman, and David Bau. Erasing concepts from diffusion models. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2426–2436, 2023.

Daiheng Gao, Shilin Lu, Wenbo Zhou, Jiaming Chu, Jie Zhang, Mengxi Jia, Bang Zhang, Zhaoxin Fan, and Weiming Zhang. EraseAnything: Enabling concept erasure in rectified flow transformers. In *International Conference on Machine Learning (ICML)*, pp. 18470–18494, 2025.

Javier García and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(42):1437–1480, 2015.

Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, et al. Gemini Robotics: Bringing AI into the physical world. *arXiv preprint arXiv:2503.20020*, 2025.

Asher J. Hancock, Xindi Wu, Lihan Zha, Olga Russakovsky, and Anirudha Majumdar. Actions as language: Fine-tuning VLMs into VLAs without catastrophic forgetting. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2509.22195.

Bear Häon, Kaylene Caswell Stocking, Ian Chuang, and Claire Tomlin. Mechanistic interpretability for steering vision-language-action models. In *Conference on Robot Learning (CoRL)*, volume 305 of *Proceedings of Machine Learning Research*, pp. 2743–2762, 2025. arXiv:2509.00328.

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

Kai-Chieh Hsu, Haimin Hu, and Jaime F. Fisac. The safety filter: A unified view of safety-critical control in autonomous systems. *Annual Review of Control, Robotics, and Autonomous Systems*, 7(1):47–72, 2024. doi: 10.1146/annurev-control-071723-102940. arXiv:2309.05837.

Songqiao Hu, Zeyi Liu, Shuang Liu, Jun Cen, Zihan Meng, Shihefeng Wang, Xiang Li, and Xiao He. VLSA: Vision-language-action models with plug-and-play safety constraint layer. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2026. arXiv:2512.11891.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, et al. OpenVLA: An open-source vision-language-action model. In *Conference on Robot Learning (CoRL)*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713, 2024. arXiv:2406.09246.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2018. doi: 10.1109/TPAMI.2017.2773081.

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 (ICLR)*, 2023.

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 (NeurIPS), Datasets and Benchmarks Track*, 2023. arXiv:2306.03310.

Shuo Liu, Ishneet Sukhvinder Singh, Yiqing Xu, Jiafei Duan, and Ranjay Krishna. VLS: Steering pretrained robot policies via vision-language models. *arXiv preprint arXiv:2602.03973*, 2026.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *IEEE Transactions on Audio, Speech and Language Processing*, 33:3776–3786, 2025. doi: 10.1109/TASLPRO.2025.3606231. arXiv:2308.08747.

Miranda Muqing Miao, Subin Kim, Brandon Yang, and Lyle Ungar. Contrastive conceptor activation steering (COAST): Unlocking vision-language-action models through hidden states. *arXiv preprint arXiv:2605.17144*, 2026.

Mitsuhiko Nakamoto, Oier Mees, Aviral Kumar, and Sergey Levine. Steering your generalists: Improving robotic foundation models via value guidance. In *Conference on Robot Learning (CoRL)*, volume 270 of *Proceedings of Machine Learning Research*, pp. 4996–5013, 2024. arXiv:2410.13816.

Open X-Embodiment Collaboration, Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, et al. Open X-Embodiment: Robotic learning datasets and RT-X models. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

Seongbin Park, Fan Zhang, Baharan Mirzasoleiman, Shahriar Talebi, and Nader Sehatbakhsh. Your model already knows: Attention-guided safety filter for vision-language-action models. *arXiv preprint arXiv:2606.09749*, 2026.

Xiangyu Qi, Yi Zeng, Tinghao Xie, Pin-Yu Chen, Ruoxi Jia, Prateek Mittal, and Peter Henderson. Fine-tuning aligned language models compromises safety, even when users do not intend to! In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2310.03693.

Jingyang Qiao, Zhizhong Zhang, Xin Tan, Yanyun Qu, Wensheng Zhang, Zhi Han, and Yuan Xie. Gradient projection for continual parameter-efficient tuning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(10):9316–9329, 2025. arXiv:2405.13383.

Ravi Ranjan and Agoritsa Polyzou. VLA-Forget: Vision-language-action unlearning for embodied foundation models. In *Proceedings of the 4th Workshop on Towards Knowledgeable Foundation Models (KnowFM 2026), ACL*, pp. 60–77, 2026. arXiv:2604.03956.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. iCaRL: Incremental classifier and representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2001–2010, 2017.

Gobinda Saha and Kaushik Roy. Continual learning with scaled gradient projection. In *AAAI Conference on Artificial Intelligence*, volume 37, pp. 9677–9685, 2023.

Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. In *International Conference on Learning Representations (ICLR)*, 2021.

Patrick Schramowski, Manuel Brack, Björn Deiseroth, and Kristian Kersting. Safe latent diffusion: Mitigating inappropriate degeneration in diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Kai Tang, Peidong Jia, Zhong Chu, Jixian Wu, Rui Ma, Jiajun Cao, et al. SafeDojo: Safe reinforcement learning for VLA via interactive world model. *arXiv preprint arXiv:2606.20698*, 2026.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*, 2023.

Andrew Wagenmaker, Yunchu Zhang, Mitsuhiko Nakamoto, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering your diffusion policy with latent space reinforcement learning. In *Conference on Robot Learning (CoRL)*, volume 305 of *Proceedings of Machine Learning Research*, pp. 258–282, 2025. arXiv:2506.15799.

Hao Wang, Jiuzhou Lei, Dayou Li, Bangya Liu, Minghui Zheng, Manling Li, Ruohan Zhang, and Zhiwen Fan. Behavior uncloning: Distilling mode redirection into policy weights without inference-time steering. *arXiv preprint arXiv:2606.29201*, 2026.

Shipeng Wang, Xiaorong Li, Jian Sun, and Zongben Xu. Training networks in null space of feature covariance for continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 184–193, 2021. arXiv:2103.07113.

Yajat Yadav, Zhiyuan Zhou, Andrew Wagenmaker, Karl Pertsch, and Sergey Levine. Robust finetuning of vision-language-action robot policies via parameter merging. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2512.08333.

Guanxiong Zeng, Yang Chen, Bo Cui, and Shan Yu. Continual learning of context-dependent processing in neural networks. *Nature Machine Intelligence*, 1(8):364–372, 2019.

Borong Zhang, Yuhao Zhang, Jiaming Ji, Yingshan Lei, Juntao Dai, Yuanpei Chen, and Yaodong Yang. SafeVLA: Towards safety alignment of vision-language-action model via constrained learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. Spotlight.

Borong Zhang, Jiahao Li, Jiachen Shen, Yuhao Zhang, Yishuai Cai, Lu Liu, Hailu Ji, Yuanpei Chen, Juntao Dai, Jiaming Ji, and Yaodong Yang. VLA-Arena: An open-source framework for benchmarking vision-language-action models. In *International Conference on Machine Learning (ICML)*, 2026a. arXiv:2512.22539.

Tianyi Zhang, Shaopeng Zhai, Haoran Zhang, Fuxian Huang, and Qi Zhang. Preserving foundational capabilities in flow-matching VLAs through conservative SFT. *arXiv preprint arXiv:2605.08879*, 2026b.

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. *arXiv preprint arXiv:2411.19309*, 2024.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, et al. Representation engineering: A top-down approach to AI transparency. *arXiv preprint arXiv:2310.01405*, 2023.

# APPENDIX

## A FLOW-MATCHING INTERPOLATION

For a demonstrated chunk $A \in \mathbb{R}^{H\times 7}$ and noise $z \sim \mathcal{N}(0, \boldsymbol{I})$ of the same shape, the flow-matching interpolant is $A_\tau = (1-\tau)\, z + \tau\, A$ with $\tau \in [0,1]$, and the demonstrated velocity is $v^\star = A - z$ (Lipman et al., 2023). We evaluate the base model $v_{\theta_0}$ and the current policy $v_\theta$ at the same $(A_\tau, \tau, \boldsymbol{c}, s)$, and draw $\tau$ from the training-time distribution of the action module.

## B IMPLEMENTATION DETAILS

We edit only linear layers of the flow-matching action module and keep every other weight frozen. For the GR00T action module (Bjorck et al., 2025), a diffusion transformer, we edit the key and value projections of every cross-attention block. For $\pi_{0.5}$ (Black et al., 2025a), we edit the output projection of every attention block of the action expert. Only the weight matrices of the edited layers are trained, and their biases stay at the base values, since a bias update would shift the layer's output on every input, the retained ones included. QwenGR00T draws 8 pairs $(z, \tau)$ per frame with $\tau \sim \mathrm{Beta}(1.5, 1.0)$, and $\pi_{0.5}$ keeps its own flow-time distribution. At inference, both models start from Gaussian noise and integrate the learned velocity field with Euler steps. QwenGR00T integrates 4 steps to an 8-step action chunk and executes the whole chunk before querying the policy again. $\pi_{0.5}$ integrates 10 steps to a 10-step action chunk, executes its first 5 actions, and then replans from the new observation.

### B.1 RETAIN SUBSPACE

The retain subspace of equation 3 is estimated from all frames of $\mathcal{D}_{\mathrm{retain}}$ in one forward pass. For each edited layer, we record its input $\boldsymbol{x}$ on these frames, form the second moment $\boldsymbol{M} = \mathbb{E}[\boldsymbol{x}\boldsymbol{x}^\top]$, and take its singular value decomposition $\boldsymbol{M} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^\top$. The columns of $\boldsymbol{U}$ and the diagonal of $\boldsymbol{\Sigma}$ are the basis $\boldsymbol{u}_i$ and the singular values $\lambda_i$ of section 3.3. All cross-attention blocks of GR00T attend to the same context embeddings, so its edited layers share one $\boldsymbol{M}$ and one $\boldsymbol{U}$. Each edited layer of $\pi_{0.5}$ has its own $\boldsymbol{M}$, its own $\boldsymbol{U}$, and its own rank $k(\epsilon)$ at the shared threshold $\epsilon$.

### B.2 LOCALITY GATE

The main text assigns one discrepancy $d$, one gate weight $w$, and one threshold $\epsilon_i$ to each frame. Each editing step draws $R$ pairs $(z, \tau)$ per frame, and every draw gives one noised chunk $A_\tau$, one base prediction $v_0$, and one correction $r = v^\star - v_0$. Under the interpolant of section A, the demonstrated chunk satisfies $A = A_\tau + (1-\tau)\, v^\star$, and the chunk the frozen policy predicts is $\hat{A} = A_\tau + (1-\tau)\, v_0$, so that $A - \hat{A} = (1-\tau)\, r$. The discrepancy of a frame is the per-action norm of this difference, averaged over the chunk and over the $R$ draws,

$$d \;=\; \frac{1}{R}\sum_{j=1}^{R}\frac{1}{H}\sum_{h=1}^{H}\big\| A_h - \hat{A}_h^{(j)}\big\|_2, \tag{6}$$

with $A_h \in \mathbb{R}^7$ the $h$-th action of $A$ and $\hat{A}_h^{(j)}$ the $h$-th action of $\hat{A}$ under the $j$-th draw. Measuring $A - \hat{A}$ rather than $r$ removes the factor $1/(1-\tau)$ from the velocity gap. Without it, two frames equally far from the demonstration would receive different discrepancies for no reason but their draw of $\tau$. Averaging over the $R$ draws removes most of the remaining variation due to $z$ and $\tau$. Under the interpolant convention of $\pi_0$, the factor is $\tau$ instead.

The percentile $p$ of section 3.3 is taken against a reference computed once with the frozen policy before any editing. The reference is the base model's discrepancies on frames drawn from $\mathcal{D}_{\mathrm{edit}}$: 10,240 frames with 8 draws of $(z, \tau)$ each for QwenGR00T, matching the 8 pairs it draws per frame in training, and 20,480 frames with one draw each for $\pi_{0.5}$. With the reference sorted as $d_{(1)} \le \cdots \le d_{(M)}$, the percentile of a discrepancy $d$ is the fraction of reference values below $d$, with those equal to $d$ counted as one half,

$$p \;=\; \frac{\#\{m : d_{(m)} < d\} + \#\{m : d_{(m)} \le d\}}{2M}, \tag{7}$$

so that equal discrepancies share one percentile. A discrepancy below or above every reference value, for which equation 7 gives 0 or 1, is clamped to $1/(2M)$ or $1 - 1/(2M)$, so that $\Phi^{-1}(p)$ in equation 4 stays finite. Over a reference without ties, $p$ takes the values $(m - 1/2)/M$ once each, so $w$ is symmetric about $1/2$ and averages $1/2$ for every $\kappa$. Each edited layer solves its own rank $k_i = k(\epsilon_i)$ on its own cumulative-energy curve of equation 3. We clamp $\epsilon_i$ to a fixed working range and log how often the clamp is active.

Table 4 crosses the swing direction with the gate shape.

Table 4: **Swing direction and gate shape on SafeLIBERO with QwenGR00T.** Looser ($\beta > 0$) gives high-discrepancy frames a lower $\epsilon_i$; tighter ($\beta < 0$) gives them a higher one. The trend in $\kappa$ reverses between the two directions. The best value of a column is in bold and the second best is underlined.

| **High-$d$ frames** | $\kappa$ | **SR↑** | **Safe↑** | **S∧S↑** | **Retain↑** | **H↑** |
|---|---|---|---|---|---|---|
| fixed $\epsilon$ ($\beta = 0$) | – | 62.5 | <u>60.2</u> | 46.0 | 80.6 | <u>58.6</u> |
| looser ($\beta = +0.1$) | 0.5 (U-shaped) | 59.7 | <u>60.2</u> | 43.8 | 73.6 | 54.9 |
| | 1.0 (uniform) | 62.8 | 58.2 | 44.2 | 76.0 | 55.9 |
| | 2.0 (bell-shaped) | 63.3 | **60.5** | **46.7** | 78.5 | 58.5 |
| tighter ($\beta = -0.1$) | 0.5 (U-shaped) | **64.7** | 53.2 | 42.3 | **85.9** | 56.7 |
| | 1.0 (uniform) | <u>63.8</u> | 57.3 | <u>46.3</u> | <u>85.3</u> | **60.0** |
| | 2.0 (bell-shaped) | 60.8 | 54.2 | 42.0 | 83.5 | 55.9 |

### B.3 Enforcing the Projection

Projecting the gradient before every optimizer update keeps $\Delta \boldsymbol{W}\, \boldsymbol{U}_k = 0$ exactly under plain gradient descent. Under AdamW it does not. AdamW divides each entry of the update by its own running scale, and an entry-wise rescaled matrix no longer satisfies $\Delta \boldsymbol{W}\, \boldsymbol{U}_k = 0$. Its weight decay also shrinks $\boldsymbol{W}_0$ itself. The parameterization $\boldsymbol{W} = \boldsymbol{W}_0 + \boldsymbol{C}\, \boldsymbol{U}^\top$ of section 3.3 avoids both. Since $\Delta \boldsymbol{W}\, \boldsymbol{u}_m$ equals the $m$-th column of $\boldsymbol{C}$, the constraint $\Delta \boldsymbol{W}\, \boldsymbol{U}_k = 0$ is a mask on the first $k$ columns of $\boldsymbol{C}$. The mask is elementwise, so every elementwise operation of AdamW preserves it, including the decoupled weight decay. With a per-frame rank $k_i$, column $j$ of $\boldsymbol{C}$ collects the gradient of the frames with $k_i < j$. We sort the frames of a batch by increasing rank, run one backward pass per distinct rank, and accumulate the gradient of each pass into the columns of $\boldsymbol{C}$ it may touch. One optimizer update then follows. Setting $\beta = 0$ yields a single group and the fixed threshold on the same code path. After editing, we materialize $\boldsymbol{W}_0 + \boldsymbol{C}\, \boldsymbol{U}^\top$ into an ordinary weight, so inference is unchanged.

## C Per-Suite Results

Tables 5–8 break down the averages of Table 1 by suite.

Table 5: **Per-suite results on SafeLIBERO (Level I).** AEGIS (Hu et al., 2026) filters the policy's actions through its CBF-based safety layer. † The method does not act on this suite, so the value equals Base's: in our implementation, COAST fits its operator for a task only when the base policy has at least three successful and three failed rollouts on it; no Object task of QwenGR00T meets this. Within each model, the best value of a column is in bold and the second best is underlined, excluding Base and the values marked †.

| | | Goal | | | Object | | | Spatial | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Model** | **Method** | SR ↑ | Safe ↑ | S∧S ↑ | SR ↑ | Safe ↑ | S∧S ↑ | SR ↑ | Safe ↑ | S∧S ↑ |
| QwenGR00T | Base | 24.0 | 6.0 | 1.0 | 3.5 | 2.5 | 2.0 | 48.5 | 10.0 | 9.5 |
| | COAST (Miao et al., 2026) | 26.0 | 5.3 | 0.5 | 3.5† | 2.5† | 2.0† | 48.3 | 10.0 | 9.5 |
| | AEGIS (Hu et al., 2026) | 28.7 | 84.3 | 27.2 | 10.0 | 21.2 | 3.3 | 43.2 | **61.8** | 31.3 |
| | SFT | **88.0** | 83.5 | 76.0 | **50.5** | **31.5** | **26.0** | 57.5 | 51.5 | 39.5 |
| | ConSFT (Zhang et al., 2026b) | 87.0 | 84.7 | **77.0** | 48.5 | 28.8 | 22.2 | **68.0** | 54.3 | **45.5** |
| | **Ours** | 87.0 | **85.0** | **77.0** | 47.5 | 30.5 | 22.5 | 57.0 | 56.5 | 39.5 |
| $\pi_{0.5}$ | Base | 50.0 | 3.5 | 1.5 | 37.0 | 9.0 | 7.5 | 68.0 | 14.0 | 14.0 |
| | COAST (Miao et al., 2026) | 66.0 | 3.5 | 1.5 | 37.5 | 7.0 | 6.0 | 68.5 | 14.0 | 13.5 |
| | AEGIS (Hu et al., 2026) | 81.5 | **96.0** | **79.5** | 55.5 | **30.5** | **16.5** | 57.5 | **56.0** | 41.5 |
| | SFT | 88.5 | 82.0 | 73.0 | **60.5** | 17.5 | 16.0 | 77.0 | 44.5 | 42.0 |
| | ConSFT (Zhang et al., 2026b) | **90.5** | 78.0 | 71.0 | 59.0 | 17.5 | 15.5 | **81.0** | 42.5 | 42.0 |
| | **Ours** | 83.5 | 90.5 | 75.5 | 45.5 | 23.5 | 11.5 | 79.5 | 54.5 | **48.5** |

Table 6: **Per-suite success rate (SR) on VLA-Arena (L0).** Avg. is the mean over the four categories (Safety, Distractor, Extrapolation, Long Horizon) of the per-category mean SR, as in Table 1. AEGIS (Hu et al., 2026) filters the policy's actions through its CBF-based safety layer and is run on the five safety suites. † The method does not act on this suite, so the value equals Base's: AEGIS is not run outside the safety suites, and in our implementation, COAST fits its operator for a task only when the base policy has at least three successful and three failed rollouts on it. Within each model, the best value of a column is in bold and the second best is underlined, excluding Base and the values marked †.

| **Model** | **Method** | Static Obst. | Cautious Grasp | Hazard Avoid. | State Pres. | Dynamic Obst. | Static Distr. | Dynamic Distr. | Prep. Comb. | Task Workfl. | Unseen Obj. | Long Horizon | **Avg.** ↑ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| QwenGR00T | Base | 7.3 | 0.0 | 2.7 | 0.0 | 0.7 | 8.0 | 0.0 | 0.0 | 0.0 | 2.0 | 4.0 | 2.7 |
| | COAST (Miao et al., 2026) | 4.7 | 0.0† | 3.3 | 0.0† | 0.7† | 4.0 | 0.0† | 0.0† | 0.0† | 4.0 | 3.7 | 2.2 |
| | AEGIS (Hu et al., 2026) | 7.3 | 0.0 | 0.0 | 0.0 | 0.0 | 8.0† | 0.0† | 0.0† | 0.0† | 2.0† | 4.0† | 2.5 |
| | SFT | **76.0** | **61.3** | **36.0** | **68.0** | 51.3 | 76.0 | **72.7** | **18.0** | **25.3** | 18.0 | **86.7** | **60.0** |
| | ConSFT (Zhang et al., 2026b) | 73.3 | 55.3 | 30.7 | 63.3 | **52.0** | **78.7** | 68.0 | 2.7 | 22.7 | 24.0 | 75.3 | 55.0 |
| | **Ours** | 70.0 | 26.7 | 32.0 | 50.0 | 46.0 | 62.0 | 66.7 | 10.7 | 18.0 | **24.7** | 74.0 | 50.3 |
| $\pi_{0.5}$ | Base | 34.7 | 14.0 | 0.7 | 54.7 | 0.0 | 45.3 | 0.7 | 0.0 | 3.3 | 4.7 | 41.3 | 22.0 |
| | COAST (Miao et al., 2026) | 36.7 | 16.0 | 0.7† | 57.3 | 0.0† | 40.7 | 0.7† | 0.0† | 4.0 | 6.0 | 44.3 | 22.6 |
| | AEGIS (Hu et al., 2026) | 34.7 | 14.0 | 0.0 | 54.7 | 0.0 | 45.3† | 0.7† | 0.0† | 3.3† | 4.7† | 41.3† | 21.9 |
| | SFT | 52.7 | 28.0 | **6.0** | 72.7 | 18.0 | **66.0** | 33.3 | **0.7** | **9.3** | 14.0 | **64.0** | **39.3** |
| | ConSFT (Zhang et al., 2026b) | **56.0** | **32.7** | 5.3 | **74.0** | **24.0** | 54.7 | **40.7** | 0.0 | 2.7 | 15.3 | 58.0 | 37.5 |
| | **Ours** | 52.0 | 18.3 | 5.3 | 70.0 | 18.0 | 62.0 | 38.0 | **0.7** | 8.0 | **19.3** | 59.3 | 37.8 |

Table 7: **Per-suite cumulative cost (CC) on VLA-Arena (L0).** Lower is better; only the five safety suites define cost. † COAST does not act on this suite, so the value equals Base's: in our implementation, COAST fits its operator for a task only when the base policy has at least three successful and three failed rollouts on it. Within each model, the best value of a column is in bold and the second best is underlined, excluding Base and the values marked †.

| Model | Method | Static Obst. | Cautious Grasp | Hazard Avoid. | State Pres. | Dynamic Obst. | Avg. ↓ |
|---|---|---|---|---|---|---|---|
| QwenGR00T | Base | 0.00 | 0.11 | 2.85 | 0.00 | 111.22 | 22.84 |
| | COAST (Miao et al., 2026) | 0.00 | 0.11† | 2.72 | 0.00† | 111.22† | 22.81 |
| | AEGIS (Hu et al., 2026) | 0.00 | **0.11** | 0.96 | 0.00 | 20.90 | 4.40 |
| | SFT | 0.00 | <u>3.82</u> | <u>0.83</u> | 0.00 | <u>17.29</u> | <u>4.39</u> |
| | ConSFT (Zhang et al., 2026b) | 0.00 | 4.87 | **0.76** | 0.00 | **13.81** | **3.89** |
| | **Ours** | 0.00 | 4.13 | 1.28 | 0.00 | 26.48 | 6.38 |
| $\pi_{0.5}$ | Base | 0.00 | 6.27 | 2.66 | 0.00 | 139.67 | 29.72 |
| | COAST (Miao et al., 2026) | 0.00 | 7.95 | 2.66† | 0.00 | 139.67† | 30.06 |
| | AEGIS (Hu et al., 2026) | 0.00 | <u>6.27</u> | **1.15** | 0.00 | 62.00 | 13.88 |
| | SFT | 0.00 | 6.43 | 2.18 | 0.00 | <u>53.87</u> | <u>12.50</u> |
| | ConSFT (Zhang et al., 2026b) | 0.00 | 6.75 | 3.47 | 0.00 | 55.45 | 13.13 |
| | **Ours** | 0.00 | **5.49** | <u>2.01</u> | 0.00 | **53.51** | **12.20** |

Table 8: **Per-suite retention on LIBERO of the edited checkpoints.** † AEGIS (Hu et al., 2026) and COAST (Miao et al., 2026) act at inference time and are not applied on LIBERO, so the value equals Base's. Within each model, the best value of a column is in bold and the second best is underlined, excluding Base and the values marked †.

| | | Edited on SafeLIBERO (Retain ↑) | | | | Edited on VLA-Arena (Retain ↑) | | | |
|---|---|---|---|---|---|---|---|---|---|
| Model | Method | Spatial | Object | Goal | Long | Spatial | Object | Goal | Long |
| QwenGR00T | Base | 96.6 | 97.8 | 97.6 | 88.8 | 96.6 | 97.8 | 97.6 | 88.8 |
| | COAST (Miao et al., 2026) | 96.6† | 97.8† | 97.6† | 88.8† | 96.6† | 97.8† | 97.6† | 88.8† |
| | AEGIS (Hu et al., 2026) | 96.6† | 97.8† | 97.6† | 88.8† | 96.6† | 97.8† | 97.6† | 88.8† |
| | SFT | 76.6 | <u>74.2</u> | <u>78.6</u> | <u>70.2</u> | <u>78.4</u> | <u>66.6</u> | 55.4 | 29.2 |
| | ConSFT (Zhang et al., 2026b) | <u>82.9</u> | 70.8 | 75.8 | 69.5 | 77.8 | 64.4 | <u>64.0</u> | <u>35.4</u> |
| | **Ours** | **83.8** | **91.6** | **83.4** | **82.2** | **88.6** | **91.4** | **76.0** | **58.0** |
| $\pi_{0.5}$ | Base | 98.6 | 98.6 | 97.2 | 92.8 | 98.6 | 98.6 | 97.2 | 92.8 |
| | COAST (Miao et al., 2026) | 98.6† | 98.6† | 97.2† | 92.8† | 98.6† | 98.6† | 97.2† | 92.8† |
| | AEGIS (Hu et al., 2026) | 98.6† | 98.6† | 97.2† | 92.8† | 98.6† | 98.6† | 97.2† | 92.8† |
| | SFT | 96.8 | 92.6 | 88.6 | 78.2 | 93.0 | 82.8 | 86.8 | 75.2 |
| | ConSFT (Zhang et al., 2026b) | <u>97.0</u> | <u>94.8</u> | <u>90.2</u> | <u>81.8</u> | <u>97.6</u> | <u>85.2</u> | <u>94.6</u> | <u>86.2</u> |
| | **Ours** | **98.4** | **95.8** | **94.4** | **90.4** | **98.0** | **96.8** | **96.8** | **89.2** |