VLA-MOPD: ON-POLICY DISTILLATION FOR MULTI-TASK VLA POLICY CONSOLIDATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Physical intelligence requires embodied agents to ground language in observations and execute actions while retaining useful skills. Vision-language-action (VLA) policies offer a shared interface, but task-specific post-training often produces separate specialists. Consolidation is difficult: mixed-task reinforcement learning provides sparse trajectory-level credit, while offline distillation may miss student-visited observations and endpoint distillation does not supervise intermediate action-generation states. We introduce VLA-MOPD, a VLA multi-teacher on-policy distillation protocol for consolidating flow-matching VLA specialists. After acquiring task or suite experts, a shared student collects robot rollouts and queries the instruction-routed frozen teacher at each visited observation and intermediate ordinary differential equation (ODE) state. Detached per-flow-step transition targets supervise valid action chunks through a masked local matching loss; an optional reference-policy term supports retention on other tasks. Starting from one trajectory per LIBERO task, VLA-MOPD raises mean success-once from 75.4% to 98.9%, compared with 98.15% for endpoint action distillation. It improves mean RoboTwin 2.0 success from 73.0% to 89.3%; on a Piper robot, two consolidation rounds improve average success from 70.0% to 90.0%. In a matched Spatial-adaptation study, reference matching limits the Object suite decrease to 3.0 points, versus 13.2 without the reference term.

1 INTRODUCTION

Physical intelligence requires agents to connect language and perception to actions whose consequences change future observations. Embodied agents must therefore ground instructions in the current physical state and execute actions with feedback. PaLM-E incorporates visual and robot-state inputs into a language model for embodied reasoning, while SayCan grounds high-level plans in the feasibility of pretrained robot skills (Driess et al., 2023; Ahn et al., 2022).

Vision-language-action (VLA) models provide a direct interface from observations and instructions to robot actions. RT-1 and RT-2 established large-scale language-conditioned control, while Open X-Embodiment broadened the available training data across institutions and embodiments (Brohan et al., 2023; Zitkovich et al., 2023; Open X-Embodiment Collaboration, 2024). Octo and OpenVLA provide reusable generalist initializations (Team et al., 2024; Kim et al., 2024). Diffusion- and flow-based policies generate continuous action chunks through an iterative trajectory (Chi et al., 2023; Black et al., 2024; Physical Intelligence et al., 2025). Yet task-specific post-training often leaves improvements distributed across separate specialist checkpoints.

We study how to consolidate these specialists into one flow-matching VLA. A flow-matching VLA encounters two coupled trajectories: the *outer robot trajectory*, whose states are observations reached after executing action chunks, and the *inner flow trajectory*, whose ODE states transform noise into each action chunk. Fixed specialist demonstrations may miss observations reached by the changing student. Because closed-loop errors alter future states, the relevant training distribution is the student's own occupancy. On-policy reinforcement learning (RL) naturally visits this distribution, but the multi-task Group Relative Policy Optimization (GRPO) baseline (MixRL) (Li et al., 2026a; Chen et al., 2025b) assigns one binary success value to an entire rollout. This gives limited credit information for individual action chunks, especially before successful rollouts become common, and shared updates can encounter cross-task gradient interference (Yu et al., 2020).

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.

Distillation provides direct teacher targets on student states. Classical interactive imitation learning such as Dataset Aggregation (DAgger) queries an expert for an action at each learner-induced observation (Ross et al., 2011). In our setting, matching the teacher's final action at these observations gives the endpoint action distillation baseline. We instead draw on DiffusionOPD and Flow-OPD, which study local transition or vector-field matching in continuous generative models (Li et al., 2026b; Fang et al., 2026), to provide denser supervision: the routed teacher supplies a detached per-flow-step transition target at every inner flow state. VLA-MOPD first acquires and freezes task or suite specialists, then lets a student collect robot rollouts and routes each instruction to its teacher. The student matches these targets across valid action chunks, including failed rollouts, while an optional reference term applies the same rule to a frozen copy of the initial policy on retained tasks.

We evaluate VLA-MOPD on four LIBERO suites and three RoboTwin 2.0 tasks (Liu et al., 2023; Chen et al., 2025c), as well as three tasks on a Piper robot. Starting from one demonstration per LIBERO task, mean success-once rises from 75.4% to 98.9%, compared with 98.15% for endpoint action distillation. The RoboTwin mean rises from 73.0% to 89.3%; on the Piper robot, two consolidation rounds raise average success from 70.0% to 90.0%. In a matched Spatial-adaptation comparison, reference matching limits the Object suite decrease to 3.0 points, versus 13.2 without the reference term.

Our contributions are (i) a two-stage recipe that decouples specialist acquisition from multi-task VLA consolidation; (ii) a local ODE transition matching objective evaluated at student-induced outer observations and inner flow states; and (iii) evidence of capability consolidation across simulated benchmarks and real-robot tasks, together with an analysis of retention.

2 Related Work

2.1 Vision-Language-Action Models

Scaling robot data and model capacity has produced policies that transfer across tasks, embodiments, and language instructions. RT-1 and RT-2 study large-scale real-robot and vision–language policy training, while Open X-Embodiment standardizes multi-institution robot data for cross-embodiment learning (Brohan et al., 2023; Zitkovich et al., 2023; Open X-Embodiment Collaboration, 2024). Octo and OpenVLA provide open generalist initializations; OpenVLA-OFT shows that action representation, chunking, and decoding choices affect downstream adaptation (Team et al., 2024; Kim et al., 2024; 2025). Diffusion Policy and the π_0 family instead generate continuous action sequences through diffusion or flow matching (Chi et al., 2023; Black et al., 2024; Physical Intelligence et al., 2025). Recent VLA designs explore partial reasoning–action decoupling, instruction tuning, hybrid diffusion–autoregressive generation, fast–slow control, latent spatio-temporal reasoning, and multi-frame modeling (Fang et al., 2025; Yang et al., 2026; Chen et al., 2025a; Liu et al., 2025; 2026; Li et al., 2025). These works provide the policy and adaptation substrate for our study.

2.2 Interactive and Reinforcement Learning for VLA Post-Training

Interactive imitation learning addresses distribution shift by querying an expert on learner-induced observations. DAgger is the canonical example (Ross et al., 2011); our setting retains this outer on-policy idea and adds targets at intermediate states of a continuous action generator. Recent VLA post-training methods use environment interaction as additional supervision: SimpleVLA-RL applies GRPO to an autoregressive action-token VLA, while π_{RL} introduces Flow-Noise and Flow-SDE variants for PPO or GRPO on flow-based π_0 and $\pi_{0.5}$ policies (Li et al., 2026a; Chen et al., 2025b). VLA-OPD queries a frozen teacher for action-token distributions on student-visited states and optimizes a reverse Kullback–Leibler (KL) objective (Zhong et al., 2026). Residual controllers and lightweight actor–critic heads offer other adaptation routes (Silver et al., 2018; Johannink et al., 2019; Xu et al., 2026; Xiao et al., 2025).

Most of these methods emphasize task-specific adaptation or direct improvement from interaction. Our setting first acquires task- or suite-level specialists and then integrates them into one shared student, emphasizing consolidation across complementary capabilities.

2.3 On-Policy Distillation for Multi-Teacher Consolidation

Multi-teacher on-policy distillation (OPD) separates specialist acquisition from integration by training domain specialists and distilling them into one student on the student's own rollouts. Language-model work uses dense token-level supervision (Agarwal et al., 2024; Ma et al., 2026; Xiaomi LLM-Core Team, 2026; DeepSeek-AI, 2026). VLA-OPD applies this idea to an autoregressive action-token policy and matches token distributions at visited observations. Our student instead matches continuous flow-action transitions: the flow-matching action expert exposes intermediate ODE states and step means that provide the local targets used for consolidation.

MergeVLA instead preserves task-specific components through sparse task masks and composable action experts, whereas our focus is teacher-guided on-policy consolidation of a shared flow-matching policy (Fu et al., 2026).

For continuous generative models, DiffusionOPD derives a closed-form per-step KL objective for diffusion transitions. Flow-OPD adapts the same idea to flow matching: it routes student samples to the relevant specialist and uses vector-field discrepancies as dense feedback for policy-gradient updates (Li et al., 2026b; Fang et al., 2026). We transfer this transition-matching recipe to flow-matching VLA action experts: the outer robot rollout supplies visited observations, the inner flow trajectory supplies intermediate states, and the task-routed teacher supplies a target at each state.

3 Motivation: Decoupling Skill Acquisition and Consolidation

The consolidation objective must evaluate teacher targets where the student actually acts and generates action chunks. The following bound isolates the effect of replacing that joint state distribution with another one.

State-distribution mismatch. Let $\ell(c, x)$ be a bounded per-state matching loss and let d and $\widetilde{d}$ be two observation–flow-state distributions. If $0 \le \ell \le M$, then

$$\left|\mathbb{E}_{d}[\ell] - \mathbb{E}_{\widetilde{d}}[\ell]\right| \le M\,\mathrm{TV}(d, \widetilde{d}). \tag{1}$$

This change-of-measure bound formalizes why student-induced states are the natural evaluation distribution for transition matching.

4 Methodology

4.1 Framework Overview

Multi-task supervised fine-tuning (SFT) initializes π_{θ_0}; supervised or RL adaptation produces frozen specialists $\pi_i^{\mathrm{T}} = \pi_{\phi_i}$ for tasks or suites $i \in \mathcal{I}$. Figure 1 shows each consolidation iteration: one student rollout collection, task-routed transition labeling, and one student update. Its z_k denotes $x_{t,k}$ below, and its local endpoint is the ODE step mean at flow step k. The trainable student parameters are its vision-language model (VLM) Low-Rank Adaptation (LoRA) modules and full action expert.

At decision step t, let $c_t = (o_t, \ell_i)$ contain the observation and task instruction. The flow-matching head generates a normalized action chunk in $\mathbb{R}^{H\times d}$ from Gaussian noise with an ODE solver. For $1 = s_0 > \cdots > s_K = 0$, define $\delta_k = s_k - s_{k+1}$ and the ODE step mean at flow step k

$$\mu_{\theta,k}(x, c) = x - \delta_k v_\theta(x, s_k, c), \qquad k = 0, \ldots, K-1, \tag{2}$$

where v_θ is the velocity field and $\mu_{\theta,k}$ is the *ODE step mean*. A frozen snapshot generates $x_{t,0} \sim \mathcal{N}(0, I)$ and recursively applies $x_{t,k+1} = \mu_{\bar{\theta},k}(x_{t,k}, c_t)$; $x_{t,K}$ is decoded and executed. We store c_t, the teacher identifier, and the full chain $x_{t,0:K}$.

At these states, the routed teacher supplies detached targets $\widehat{\mu}_{i,t,k} = \mathrm{sg}[\mu_{\phi_i,k}(x_{t,k}, c_t)]$. Teacher and student compute separate VLM prefixes but predict in shared normalized action coordinates. We cache targets, update only the student's VLM LoRA and action expert, and refresh the student rollout after each update (Algorithm 1).

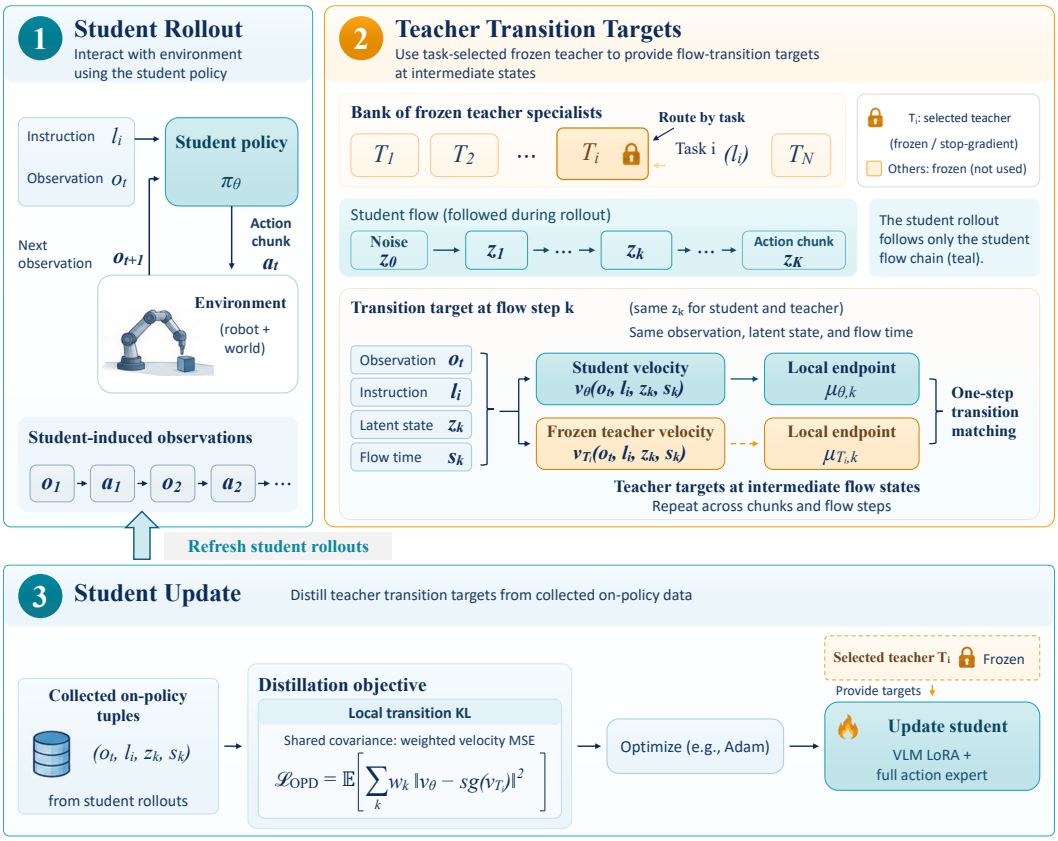


Figure 1: VLA-MOPD consolidation. Student rollouts provide observations and ODE states; the task-routed specialist labels transitions. The student matches these targets and refreshes rollouts. Velocity and ODE step-mean regression are Euler-equivalent; the transition-KL interpretation assumes shared Gaussian covariance (Eq. 4).

4.2 Optimization Objective

Following DiffusionOPD (Li et al., 2026b), we start from a stochastic transition-KL view, then use direct transition matching for our deterministic ODE sampler. Let $d_i^{\bar{\theta}}(c)$ be the student snapshot's observation–instruction distribution and $q_{\bar{\theta}}(x_{0:K} \mid c)$ its ODE-chain law. At each stored x_k, we compare the student and frozen-teacher transitions. For a fixed snapshot $\bar{\theta}$, the stochastic reference objective is

$$\mathcal{L}_{\text{OPD}}(\theta;\bar{\theta}) = \mathbb{E}_{\substack{i\sim p(i),\ c\sim d_i^{\bar{\theta}} \\ x_{0:K}\sim q_{\bar{\theta}}(\cdot|c)}} \left[\frac{1}{K} \sum_{k=0}^{K-1} \mathrm{KL}\big(p_{\theta,k}(\cdot \mid x_k, c) \,\|\, p^{\mathrm{T}}_{i,k}(\cdot \mid x_k, c)\big) \right]. \tag{3}$$

Here the student supplies both the robot observations and intermediate flow states, while the instruction selects the teacher.

Closed-form KL. For the closed-form transition calculation, first consider stochastic transitions with shared covariance. Write $p_{\theta,k} = \mathcal{N}(\mu_{\theta,k}, \Sigma_k)$ and $p^{\mathrm{T}}_{i,k} = \mathcal{N}(\mu_{\phi_i,k}, \Sigma_k)$, with both means conditioned on the same (x_k, c). Using the Euler parameterization $\mu_{\theta,k} = x_k - \delta_k v_\theta(x_k, s_k, c)$ and $\mu_{\phi_i,k} = x_k - \delta_k v_{\phi_i}(x_k, s_k, c)$, the common state cancels and

$$\mathrm{KL}\big(p_{\theta,k} \,\|\, p^{\mathrm{T}}_{i,k}\big) = \tfrac{1}{2}\Delta\mu_k^\top \Sigma_k^{-1} \Delta\mu_k = \frac{\delta_k^2}{2\sigma_k^2} \|v_\theta(x_k, s_k, c) - v_{\phi_i}(x_k, s_k, c)\|_F^2 \quad \text{if } \Sigma_k = \sigma_k^2 I, \tag{4}$$

where $\Delta\mu_k = \mu_{\theta,k} - \mu_{\phi_i,k}$. This is an exact KL only for the stated stochastic Gaussian kernels.

Algorithm 1 VLA-MOPD: Specialist Acquisition and On-Policy Consolidation

Require: Multi-task SFT data $\mathcal{D}_{\text{SFT}}$, specialist acquisition procedure $\mathcal{A}$
Require: Teacher index set $\mathcal{I}$, task sampling schedule $p(i)$, and flow schedule $s_{0:K}$
$\theta_0 \leftarrow \text{SFT}(\mathcal{D}_{\text{SFT}})$; $\theta \leftarrow \theta_0$
for $i \in \mathcal{I}$ **do**
  $\phi_i \leftarrow \mathcal{A}(i)$; freeze ϕ_i
end for
$\theta_{\text{train}} \leftarrow (\theta_{\text{VLM-LoRA}}, \theta_{\text{action}})$
while the configured training budget remains **do**
  $\bar{\theta} \leftarrow \theta$
  $\mathcal{D}_{\text{roll}} \leftarrow \text{RolloutODE}(\pi_{\bar{\theta}}, p(i), s_{0:K})$
    $\mathcal{D}_{\text{roll}} = \{(i, c, x_{0:K}, m)\}$; m keeps valid action chunks and masks terminated or padded rows
  for each $(i, c, x_{0:K}, m) \in \mathcal{D}_{\text{roll}}$ **do**
    $e_i^{\text{T}} \leftarrow \text{Prefix}_{\phi_i}(c)$ ▷ Teacher's own VLM
    for $k = 0, \ldots, K-1$ **do**
      $v_{i,k}^{\text{T}} \leftarrow \text{Velocity}_{\phi_i}(x_k, s_k, c; e_i^{\text{T}})$
      $\widehat{\mu}_{i,k} \leftarrow \text{sg}[x_k - \delta_k v_{i,k}^{\text{T}}]$; cache $\widehat{\mu}_{i,k}$
    end for
  end for
  for each update minibatch $\mathcal{B} \subset \mathcal{D}_{\text{roll}}$ **do**
    $e^{\text{S}} \leftarrow \text{Prefix}_{\theta}(c_{\mathcal{B}})$ ▷ Trainable student prefix
    for $k = 0, \ldots, K-1$ **do**
      $v_{\theta,k} \leftarrow \text{Velocity}_{\theta}(x_{\mathcal{B},k}, s_k, c_{\mathcal{B}}; e^{\text{S}})$
      $\mu_{\theta,k} \leftarrow x_{\mathcal{B},k} - \delta_k v_{\theta,k}$
    end for
    $\widehat{\mathcal{L}}_{\text{ODE}} \leftarrow \text{MaskedMeanMatch}(\mu_\theta, \widehat{\mu}, m)$ ▷ Eq. 6
    $\theta_{\text{train}} \leftarrow \text{Update}(\theta_{\text{train}}, \nabla_{\theta_{\text{train}}} \widehat{\mathcal{L}}_{\text{ODE}})$
  end for
end while
return π_θ

Deterministic ODE matching. Our action generator instead uses deterministic Euler steps $x_{k+1} = \mu_{\theta,k}(x_k, c)$. As in the deterministic regime of DiffusionOPD, we directly match the student and teacher ODE step means at the same stored student state:

$$\ell_{i,k}^{\text{ODE}}(x_k, c) = \tfrac{1}{2}\|\mu_{\theta,k}(x_k, c) - \mu_{\phi_i,k}(x_k, c)\|_F^2 = \tfrac{\delta_k^2}{2}\|v_\theta(x_k, s_k, c) - v_{\phi_i}(x_k, s_k, c)\|_F^2. \tag{5}$$

This is a squared transition-matching surrogate, not a KL between deterministic kernels: distinct Dirac transitions have no finite KL.

VLA action-chunk objective. For each robot observation c_t, the student generates an $H \times d$ action chunk and stores its inner ODE states. We query the routed teacher at those states, detach its targets, and average Eq. 5 over flow steps and action elements:

$$\mathcal{L}_{\text{ODE}}(\theta; \bar{\theta}) = \mathbb{E}_{\substack{i \sim p(i),\ c \sim d_i^{\bar{\theta}} \\ x_{0:K} \sim q_{\bar{\theta}}(\cdot|c)}} \left[\frac{1}{2KHd} \sum_{k=0}^{K-1} \|\mu_{\theta,k}(x_k, c) - \text{sg}[\mu_{\phi_i,k}(x_k, c)]\|_F^2 \right]. \tag{6}$$

Equation 6 uses uniform mean-space weighting, so its equivalent velocity error carries δ_k^2 but no inverse-covariance factor. Appendix A gives masking and normalization details; Appendix B expands the KL-to-ODE connection.

Frozen-snapshot projection. For a fixed stored state, write $J_{\theta,k} = \partial v_\theta(x_k, s_k, c)/\partial\theta$. Differentiating the velocity-field objective above gives the update direction

$$\nabla_\theta \mathcal{L}_{\text{ODE}} = \mathbb{E}\left[\frac{1}{KHd} \sum_{k=0}^{K-1} \delta_k^2 J_{\theta,k}^\top \big(v_\theta(x_k, s_k, c) - v_{\phi_i}(x_k, s_k, c)\big) \right], \tag{7}$$

where the expectation is over the frozen snapshot distributions in Eq. 3, valid terms are selected by the same mask as Eq. 10, and $q_{\bar{\theta}}$ and $d_i^{\bar{\theta}}$ are not differentiated. Thus, OPD performs a local projection of the student field on the student-induced outer and inner state distributions. The teacher labels every valid chunk at the student's visited states, including failed rollouts; we differentiate Eq. 6 directly through the student's VLM LoRA and action expert with stored states and teacher targets fixed, then refresh the rollout snapshot.

4.3 Reference-Policy Regularization for Retention

To preserve capabilities outside the adaptation set, we freeze the starting policy $\pi_{\mathrm{ref}} = \pi_{\theta_0}$ and match its step means on retained-task observations, using the same rule as the specialists.

Let $\mathcal{D}_{\mathrm{retain}}$ denote these retained-task observations and $q_{\bar{\theta}}(x_{0:K} \mid c)$ the student snapshot's ODE chain distribution. For valid chunks, the reference loss is

$$\mathcal{L}_{\mathrm{ref}}(\theta;\bar{\theta}) = \mathbb{E}_{\substack{c\sim\mathcal{D}_{\mathrm{retain}} \\ x_{0:K}\sim q_{\bar{\theta}}(\cdot|c)}} \left[\frac{1}{2KHd} \sum_{k=0}^{K-1} \|\mu_{\theta,k}(x_k,c) - \mathrm{sg}[\mu_{\theta_0,k}(x_k,c)]\|_F^2 \right]. \tag{8}$$

The combined objective is

$$\mathcal{L}_{\mathrm{total}} = \mathcal{L}_{\mathrm{ODE}} + \beta\mathcal{L}_{\mathrm{ref}}, \qquad \beta \geq 0. \tag{9}$$

We call this reference mean matching. At each retained-task state, it is the local Gaussian-KL proxy induced by Eq. 4. Retained-task observations can come from environment resets without expert trajectories.

5 Experiments

5.1 Experimental Setup

We evaluate success-once throughout and report the unweighted mean when multiple suites or tasks are available. Student consolidation updates the VLM LoRA modules and the full action expert, while the remaining VLM parameters and all teacher parameters are frozen.

LIBERO. We use the four single-arm manipulation suites Spatial, Object, Goal, and LIBERO-10 from LIBERO (Liu et al., 2023). To test the extreme data-scarcity setting, the student is initialized by 1-traj SFT, using one demonstration per task. The full-data reference setting uses 50 trajectories per task. Student and specialist teacher checkpoints are obtained with the $\pi_{0.5}$ SFT and Proximal Policy Optimization (PPO) recipes in RLinf (Zang et al., 2025), respectively. We use the standard LIBERO success-once evaluation protocol. Each LIBERO suite is evaluated on 500 episodes; we report one training seed and compare the same four-suite success rates across methods.

RoboTwin 2.0. We select three representative dual-arm tasks from RoboTwin 2.0 (Chen et al., 2025c): *Place Empty Cup*, *Pick Dual Bottles*, and *Stack Bowls Two*. Their episode horizons correspond to medium, short, and long tasks, respectively. The common student initialization is a multi-task SFT model trained on 50 clean trajectories per task. The "Student initialization" row reports this checkpoint; VLA-MOPD further consolidates it with task-routed PPO teachers, while the teacher row reports the specialists themselves. Each RoboTwin task is evaluated on 100 episodes using the official task-specific limits, and we report success-once rates.

5.2 Main Results

LIBERO. Table 1 places published references alongside our controlled runs. Published references use their reported backbones, data, and evaluation protocols, while our method comparisons use the same initialization and update protocol; the average is the unweighted mean of the four suites.

RoboTwin 2.0. Table 2 reports the three-task RoboTwin results. Starting from the common 50-traj SFT student, VLA-MOPD improves the mean success-once rate from 73.0% to 89.3%, closing most of the gap to the routed teachers (94.0%).

Table 1: Main results on LIBERO (success-once rate, %). VLA-MOPD is trained from a 1-traj-per-task student initialization. Published references use their reported backbones and data. The routed teacher uses PPO-LIBERO-130 for Spatial/Object/Goal and PPO120 for LIBERO-10.

Method	Spatial	Object	Goal	LIBERO-10	Avg.
Full dataset (50-traj)					
Octo (Team et al., 2024)	78.9	85.7	84.6	51.1	75.1
OpenVLA (Kim et al., 2024)	84.7	88.4	79.2	53.7	76.5
NORA (Hung et al., 2025)	92.2	95.4	89.4	74.6	87.9
π_0 + FAST (Black et al., 2024; Pertsch et al., 2025)	96.4	96.8	88.6	60.2	85.5
$\pi_{0.5}$ (Physical Intelligence et al., 2025)	98.8	98.2	98.0	92.4	96.9
Our runs					
Student initialization (base SFT, 1-traj)	85.8	95.6	78.0	42.2	75.4
Routed PPO teachers	97.4	**99.2**	**98.6**	98.8	98.5
Endpoint action distillation	99.0	99.2	96.8	97.6	98.2
VLA-MOPD (ours)	**99.4**	99.0	98.0	**99.0**	**98.9**

Table 2: Main results on RoboTwin 2.0 (success-once rate, %). Each entry is evaluated over 100 episodes.

Method	Place Empty Cup	Pick Dual Bottles	Stack Bowls Two	Avg.
Student initialization (50-traj SFT)	69.0	75.0	75.0	73.0
Routed PPO teachers	**93.0**	**97.0**	**92.0**	**94.0**
VLA-MOPD (ours)	87.0	93.0	88.0	89.3

5.3 Ablations

Training Efficiency. We compare VLA-MOPD and MixRL over the first 500 training steps, starting from the same multi-task SFT student. Figure 2 reports success-once rates for each LIBERO suite and their unweighted average. We measure efficiency by convergence in student optimizer steps under a matched run configuration. MixRL uses the GRPO algorithm implemented as RLinf's Pi-RL baseline (Chen et al., 2025b; Zang et al., 2025), with the same environment count, rollout count per optimizer step, and total rollout budget as the main VLA-MOPD run. The recorded VLA-MOPD run improves non-monotonically from 75.4% at initialization to 98.1% by step 300 and 98.85% at the last recorded checkpoint; the early dip is visible in Figure 2. MixRL remains lower and oscillatory: at the shared step 400, it reaches 82.55% versus 97.6% for VLA-MOPD. The gap is largest on Goal and LIBERO-10. The gap is consistent with VLA-MOPD receiving teacher transition targets at each student observation and action chunk, while MixRL must discover successful trajectories from a terminal binary reward.

Intermediate flow-state supervision. To isolate the supervision target, we compare VLA-MOPD with endpoint action distillation using the same on-policy student observations and teacher action labels, but supervising only the final action. At 500 steps, endpoint action distillation reaches 98.15% average success, while VLA-MOPD reaches 98.9%. Spatial and Object are comparable (99.0% vs. 99.4%, and 99.2% vs. 99.0%, respectively); VLA-MOPD gains 1.2 points on Goal and 1.4 points on LIBERO-10. This pattern suggests that intermediate transition supervision is particularly useful for more compositional goals and longer-horizon tasks, where local action errors can accumulate across action chunks.

Forgetting and reference retention. This ablation trains on Spatial and evaluates Object with reference OPD ($\beta = 1$), no-reference OPD ($\beta = 0$), and SFT. The reference is a frozen copy of the starting student. We define Object forgetting as the change from the common base, $\Delta_{\text{Object}} = S_{\text{Object}}(\pi_\theta) - S_{\text{Object}}(\pi_0)$; negative values indicate forgetting. Snapshots are selected by near-matched Spatial success, then evaluated separately on Object (Figure 3(a)). From the 85.8% Spatial/95.6% Object base, matched Spatial scores of 96.6%, 98.0%, and 96.6% yield Object scores of 92.6%, 82.4%, and 82.8% for reference OPD, no-reference OPD, and SFT. The corresponding drops are 3.0, 13.2, and 12.8 points; reference OPD reduces the drop by 10.2 points versus no-reference OPD.

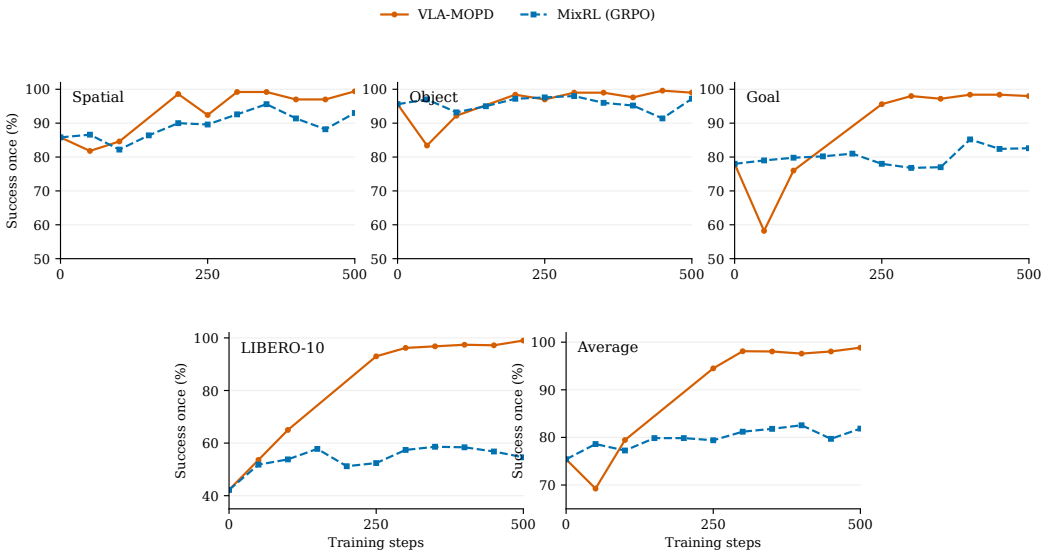
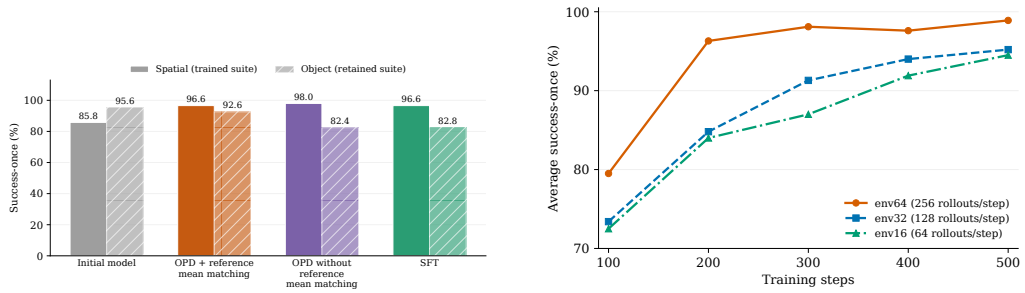


Figure 2: Convergence over the first 500 training steps. The five panels show success-once rates for the four LIBERO suites and their unweighted average. Points mark available evaluations; checkpoints with missing suite scores have no average. MixRL and VLA-MOPD use matched rollout budgets and are plotted at 50-step checkpoints without smoothing.



(a) (b)

Figure 3: Retention and rollout-budget analyses. (a) Object retention at model snapshots with near-matched Spatial adaptation. Each selected snapshot is evaluated separately on Spatial and Object; Spatial establishes the matched adaptation level and Object is the retention outcome. (b) Average success during rollout-budget sensitivity with 64, 32, and 16 parallel environments.

Rollout-Budget Sensitivity. Env64, env32, and env16 collect 256, 128, and 64 trajectories per optimizer step. Figure 3(b) shows that env64 reaches 96.3% at step 200 and 98.1% at step 300, compared with $84.8/91.3\%$ for env32 and $84.0/87.0\%$ for env16. The gap narrows by step 500, when the three settings reach 98.9%, 95.2%, and 94.5%. These curves characterize convergence per optimizer step under the three rollout-budget settings.

5.4 Real-World Experiment

We use real-world data from a Piper single-arm robot and consider three tasks: Pick Banana, Stack Cup, and Sort Number. The student is initialized by mixed-task SFT of a $\pi_{0.5}$ model on the collected multi-task real-world demonstrations. For each task, we train a separate $\pi_{0.5}$ teacher with single-task SFT on that task's real-world demonstrations. The student then performs two rollout rounds; each round collects 50 trajectories per task. We use the resulting student observations and saved flow states as on-policy data for OPD, and train for one epoch after each rollout round. Over 30

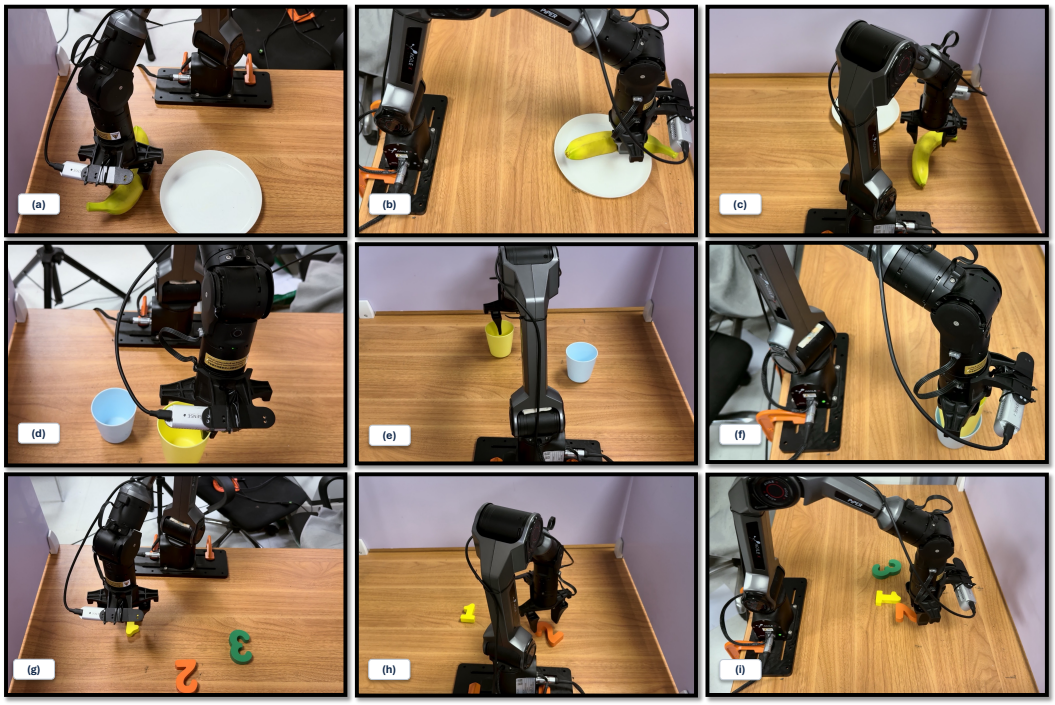


Figure 4: Real-world experiment scenes. Rows show Pick Banana (a–c), Stack Cup (d–f), and Sort Number (g–i).

evaluation trials per task, VLA-MOPD reaches 90.0%, compared with 70.0% for the student and 93.3% for the routed teacher. Task-wise rates are shown in Table 3.

Table 3: Real-world experiment success-once rates (%; 30 trials per task).

Method	Pick Banana	Stack Cup	Sort Number	Avg.
Student	86.7	63.3	60.0	70.0
Teacher	**100.0**	**96.7**	**83.3**	**93.3**
VLA-MOPD	**100.0**	93.3	76.7	90.0

6 Conclusion

We present VLA-MOPD as a two-stage protocol for consolidating task or suite specialists into one flow-matching VLA. The student follows its own rollouts and receives local transition targets at the intermediate states of its action generator, connecting specialist supervision to the states encountered by the consolidated policy. Across the evaluated benchmarks and real-robot setting, the resulting student demonstrates effective multi-task capability consolidation and improves over its initialization.

7 Limitations

VLA-MOPD requires task or suite specialist teachers before consolidation. Acquiring and storing these teachers adds upfront training and maintenance cost, especially as the number of task families grows. The quality and task coverage of the available teachers also determine which capabilities the student can consolidate.

AI Assistance Statement

In this work, we used generative AI tools to provide feedback on research methodology and to interpret results. We have not used generative AI tools to formulate mathematical claims, provide critical ingredients for proving mathematical claims, or assist in the writing of proofs, and the rest of the required disclosure tasks are not applicable to this work. Additionally, we used generative AI tools to create and edit software code, create and modify scientific figures, draft parts of this paper, and edit the paper to improve readability. We have reviewed all AI-assisted work: every AI-generated passage, line of code, figure, and claim was read, verified, and edited by the authors, and all reported numbers were checked against the raw experimental results. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-Policy Distillation of Language Models: Learning from Self-Generated Mistakes. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2306.13649.

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, et al. Do as i can, not as i say: Grounding language in robotic affordances, 2022. URL https://arxiv.org/abs/2204.01691.

Kevin Black et al. π_0: A Vision-Language-Action Flow Model for General Robot Control, 2024. URL https://arxiv.org/abs/2410.24164.

Anthony Brohan et al. RT-1: Robotics Transformer for Real-World Control at Scale. In *Robotics: Science and Systems*, 2023.

Hao Chen, Jiaming Liu, Chenyang Gu, Zhuoyang Liu, Renrui Zhang, Xiaoqi Li, Xiao He, Yandong Guo, Chi-Wing Fu, Shanghang Zhang, et al. Fast-in-Slow: A Dual-System Foundation Model Unifying Fast Manipulation within Slow Reasoning, 2025a. URL https://arxiv.org/abs/2506.01953.

Kang Chen, Zhihao Liu, Tonghe Zhang, Zhen Guo, Si Xu, Hao Lin, Hongzhi Zang, Xiang Li, Quanlu Zhang, Zhaofei Yu, Guoliang Fan, Tiejun Huang, Yu Wang, and Chao Yu. π_{RL}: Online RL Fine-tuning for Flow-based Vision-Language-Action Models, 2025b. URL https://arxiv.org/abs/2510.25889.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Qiwei Liang, Zixuan Li, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. RoboTwin 2.0: A Scalable Data Generator and Benchmark with Strong Domain Randomization for Robust Bimanual Robotic Manipulation, 2025c. URL https://arxiv.org/abs/2506.18088.

Cheng Chi et al. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. *The International Journal of Robotics Research*, 2023.

DeepSeek-AI. DeepSeek-V4: Towards Highly Efficient Million-Token Context Intelligence, 2026. URL https://arxiv.org/abs/2606.19348.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, et al. PaLM-E: An embodied multimodal language model, 2023. URL https://arxiv.org/abs/2303.03378.

Zhen Fang, Zhuoyang Liu, Jiaming Liu, Hao Chen, Yu Zeng, Shiting Huang, Zehui Chen, Lin Chen, Shanghang Zhang, and Feng Zhao. DualVLA: Building a Generalizable Embodied Agent via Partial Decoupling of Reasoning and Action, 2025. URL https://arxiv.org/abs/2511.22134.

Zhen Fang, Wenxuan Huang, Yu Zeng, Yiming Zhao, Shuang Chen, Kaituo Feng, Yunlong Lin, Lin Chen, Zehui Chen, Shaosheng Cao, and Feng Zhao. Flow-OPD: On-Policy Distillation for Flow Matching Models, 2026. URL https://arxiv.org/abs/2605.08063.

Yuxia Fu, Zhizhen Zhang, Yuqi Zhang, Zijian Wang, Zi Huang, and Yadan Luo. MergeVLA: Cross-Skill Model Merging Toward a Generalist Vision-Language-Action Agent. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/papers/Fu_MergeVLA_Cross-Skill_Model_Merging_Toward_a_Generalist_Vision-Language-Action_Agent_CVPR_2026_paper.pdf.

Chia-Yu Hung, Qi Sun, Pengfei Hong, Amir Zadeh, Chuan Li, U-Xuan Tan, Navonil Majumder, and Soujanya Poria. NORA: A Small Open-Sourced Generalist Vision Language Action Model for Embodied Tasks, 2025. URL https://arxiv.org/abs/2504.19854.

Tobias Johannink, Shikhar Bahl, Ashvin Nair, Jianlan Luo, Avinash Kumar, Matthias Loskyll, Juan Aparicio Ojea, Eugen Solowjow, and Sergey Levine. Residual Reinforcement Learning for Robot Control. In *IEEE International Conference on Robotics and Automation*, pp. 6023–6029, 2019. URL https://arxiv.org/abs/1812.03201.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-Tuning Vision-Language-Action Models: Optimizing Speed and Success, 2025. URL https://arxiv.org/abs/2502.19645.

Moo Jin Kim et al. OpenVLA: An Open-Source Vision-Language-Action Model, 2024. URL https://arxiv.org/abs/2406.09246.

Hao Li, Shuai Yang, Yilun Chen, Xinyi Chen, Xiaoda Yang, Yang Tian, Hanqing Wang, Tai Wang, Dahua Lin, Feng Zhao, et al. CronusVLA: Towards Efficient and Robust Manipulation via Multi-Frame Vision-Language-Action Modeling, 2025. URL https://arxiv.org/abs/2506.19816.

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*, 2026a. URL https://openreview.net/forum?id=TQhSodCM4r.

Quanhao Li, Junqiu Yu, Kaixun Jiang, Yujie Wei, Zhen Xing, Pandeng Li, Ruihang Chu, Shiwei Zhang, Yu Liu, and Zuxuan Wu. DiffusionOPD: A Unified Perspective of On-Policy Distillation in Diffusion Models, 2026b. URL https://arxiv.org/abs/2605.15055.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking Knowledge Transfer for Lifelong Robot Learning, 2023. URL https://arxiv.org/abs/2306.03310.

Jiaming Liu, Hao Chen, Pengju An, Zhuoyang Liu, Renrui Zhang, Chenyang Gu, Xiaoqi Li, Ziyu Guo, Sixiang Chen, Mengzhen Liu, et al. HybridVLA: Collaborative Diffusion and Autoregression in a Unified Vision-Language-Action Model, 2025. URL https://arxiv.org/abs/2503.10631.

Zhuoyang Liu, Jiaming Liu, Hao Chen, Jiale Yu, Ziyu Guo, Chengkai Hou, Chenyang Gu, Xiangju Mi, Renrui Zhang, Kun Wu, et al. LaST$_0$: Latent Spatio-Temporal Chain-of-Thought for Robotic Vision-Language-Action Models, 2026. URL https://arxiv.org/abs/2601.05248.

Wenhan Ma, Jianyu Wei, Liang Zhao, Hailin Zhang, Bangjun Xiao, Lei Li, Qibin Yang, Bofei Gao, Yudong Wang, Rang Li, Jinhao Dong, Zhifang Sui, and Fuli Luo. MOPD: Multi-Teacher On-Policy Distillation for Capability Integration in LLM Post-Training, 2026. URL https://arxiv.org/abs/2606.30406.

Open X-Embodiment Collaboration. Open X-Embodiment: Robotic Learning Datasets and RT-X Models. In *2024 IEEE International Conference on Robotics and Automation*, 2024.

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, 2025. URL https://arxiv.org/abs/2501.09747.

Physical Intelligence, Kevin Black, Noah Brown, et al. $\pi_{0.5}$: A Vision-Language-Action Model with Open-World Generalization, 2025. URL https://arxiv.org/abs/2504.16054.

Stephane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 627–635, 2011. URL https://proceedings.mlr.press/v15/ross11a.html.

Tom Silver, Kelsey R. Allen, Joshua B. Tenenbaum, and Leslie P. Kaelbling. Residual Policy Learning, 2018. URL https://arxiv.org/abs/1812.06298.

Octo Team, Dibya Ghosh, et al. Octo: An Open-Source Generalist Robot Policy, 2024. URL https://arxiv.org/abs/2405.12213.

Wenli Xiao, Haotian Lin, Andy Peng, Haoru Xue, Tairan He, Yuqi Xie, Fengyuan Hu, Jimmy Wu, Zhengyi Luo, Linxi Fan, Guanya Shi, and Yuke Zhu. Self-Improving Vision-Language-Action Models with Data Generation via Residual RL, 2025. URL https://arxiv.org/abs/2511.00091.

Xiaomi LLM-Core Team. MiMo-V2-Flash Technical Report, 2026. URL https://arxiv.org/abs/2601.02780.

Charles Xu, Jost Tobias Springenberg, Michael Equi, Ali Amin, Adnan Esmail, Sergey Levine, and Liyiming Ke. RL Token: Bootstrapping Online RL with Vision-Language-Action Models, 2026. URL https://arxiv.org/abs/2604.23073.

Shuai Yang, Hao Li, Bin Wang, Yilun Chen, Yang Tian, Tai Wang, Hanqing Wang, Feng Zhao, Yiyi Liao, and Jiangmiao Pang. Vision-Language-Action Instruction Tuning: From Understanding to Manipulation. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=tsxwloasw5.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient Surgery for Multi-Task Learning. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://arxiv.org/abs/2001.06782.

Hongzhi Zang et al. RLinf-VLA: A Unified and Efficient Framework for VLA+RL Training, 2025. URL https://arxiv.org/abs/2510.06710.

Zhide Zhong, Haodong Yan, Junfeng Li, Junjie He, Tianran Zhang, and Haoang Li. VLA-OPD: Bridging Offline SFT and Online RL for Vision-Language-Action Models via On-Policy Distillation, 2026. URL https://arxiv.org/abs/2603.26666.

Brianna Zitkovich et al. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control. In *Conference on Robot Learning*, 2023.

A Implementation Details

Field	Setting
Base policy	$\pi_{0.5}$; multi-task SFT student initialization.
Student updates	Native VLM LoRA parameters and the full action expert; the remaining VLM parameters are frozen.
Teacher bank	Independent frozen task- or suite-specialist policies, routed by the stored teacher identifier.
Conditioning	Teacher and student encode the same observation independently and each reuse their own prefix across flow steps.
Rollout	Student environment interaction; saved normalized ODE states for every action chunk. The evaluated flow configuration uses five solver steps and the same step schedule for teacher and student targets.
Action horizon	LIBERO action chunks contain 10 actions; RoboTwin uses an action horizon of 50.
Objective	Half squared ODE step-mean error, averaged over flow steps and valid model action elements.
Teacher targets	Computed on stored student states and cached before student optimization.
Backward pass	Shared trainable prefix graph; losses from all selected flow steps are averaged before backward.
Rollout budget	64 training environments; four rollout epochs per optimizer step, one for each suite (Spatial/Object/Goal/LIBERO-10). The suite-specific maximum steps are 240/240/320/480, respectively.
Actor batches	Microbatch size 8 and global batch size 1024.
Optimization	Adam with learning rate 2.5×10^{-5}, $(\beta_1, \beta_2) = (0.9, 0.95)$, weight decay 0.01, gradient clipping at 1.0, and cosine decay.
Training schedule	1000 optimizer steps, checkpoint saved every 50 steps, random seed 42.
OPD target computation	Teacher microbatch size 4; all five flow steps are used; $\beta = 0$ and teacher-ratio schedule fixed at zero.

The implementation uses the configured five-step `flow_ode` schedule; the cached teacher target and student prediction use the same normalized step coordinates. The action dimension d in Eq. 6 denotes the model's internal normalized action representation, which can exceed the number of robot control coordinates. Matching is performed before decoding and uses all coordinates returned by the action head. Normalization assets are shared between teacher and student. This aligns their output coordinates while allowing their VLM encoders and action experts to have different weights.

For an update microbatch $\mathcal{B}$, let $m_{b,h} \in \{0, 1\}$ be the validity mask, broadcast over the d model action coordinates. The estimator of Eq. 6 used by the implementation is

$$\widehat{\mathcal{L}}_{\mathrm{ODE}} = \frac{1}{2KZ_{\mathcal{B}}} \sum_{b \in \mathcal{B}} \sum_{k=0}^{K-1} \sum_{h=1}^{H} \sum_{j=1}^{d} m_{b,h} \left[\mu_{\theta,b,k,h,j} - \widehat{\mu}_{b,k,h,j} \right]^2, \tag{10}$$

where $Z_{\mathcal{B}} = \max(1, d \sum_{b,h} m_{b,h})$, $\mu_{\theta,b,k} = \mu_{\theta,k}(x_{b,k}, c_b)$, and $\widehat{\mu}_{b,k} = \mathrm{sg}[\mu_{\phi_{i_b},k}(x_{b,k}, c_b)]$. For chunk-level validity masks, the mask is broadcast over all positions and coordinates of the action chunk. Completed rollout slots and padded samples are excluded. The implementation computes a normalized masked loss within each microbatch, then averages its gradients across accumulation steps and distributed workers. Accordingly, it averages microbatch-normalized losses; when valid counts differ, this need not equal one global valid-element mean. Both successful and unsuccessful episodes can supply valid training samples; restricting distillation to failed episodes is an optional sampling filter. The student computes its trainable VLM prefix once per microbatch and reuses it across the denoising-step predictions. Their losses are averaged before a joint backward pass through the shared prefix graph and action expert.

B From Reverse KL to ODE Step-Mean Matching

This section makes explicit the connection between the reverse-KL definition in DiffusionOPD (Li et al., 2026b) and the deterministic loss used in our implementation. Fix an instruction–observation pair c, a stored student state x_k, and flow step k. Let $p_{\theta,k}(\cdot \mid x_k, c)$ denote the student transition and let $p^{\mathrm{T}}_{i,k}(\cdot \mid x_k, c)$ denote the transition of the routed, frozen teacher. The reverse transition objective is

$$\mathcal{L}_{\mathrm{rev}} = \mathbb{E}_{\substack{i\sim p(i),\ c\sim d_i^{\bar\theta} \\ x_{0:K}\sim q_{\bar\theta}(\cdot|c)}} \left[\frac{1}{K} \sum_{k=0}^{K-1} \mathrm{KL}\big(p_{\theta,k}(\cdot \mid x_k, c) \,\big\|\, p^{\mathrm{T}}_{i,k}(\cdot \mid x_k, c)\big) \right]. \tag{11}$$

The first argument is the student, so this is the reverse KL convention used here. The snapshot distribution $d_i^{\bar\theta}$ and the stored-state law $q_{\bar\theta}$ are held fixed during the update.

Reverse KL for shared-covariance transitions. Assume temporarily that both transitions are Gaussian, $p_{\theta,k} = \mathcal{N}(\mu_{\theta,k}, \Sigma_k)$ and $p^{\mathrm{T}}_{i,k} = \mathcal{N}(\mu^{\mathrm{T}}_{i,k}, \Sigma_k)$, with the same positive covariance at the same state. The general Gaussian KL reduces to

$$\mathrm{KL}(p_{\theta,k} \| p^{\mathrm{T}}_{i,k}) = \frac{1}{2}(\mu_{\theta,k} - \mu^{\mathrm{T}}_{i,k})^\top \Sigma_k^{-1} (\mu_{\theta,k} - \mu^{\mathrm{T}}_{i,k}). \tag{12}$$

For $\Sigma_k = \sigma_k^2 I$, this is

$$\mathrm{KL}(p_{\theta,k} \| p^{\mathrm{T}}_{i,k}) = \frac{1}{2\sigma_k^2} \left\| \mu_{\theta,k} - \mu^{\mathrm{T}}_{i,k} \right\|_F^2. \tag{13}$$

Velocity-field form. Write one Euler transition of the student and teacher flow fields as

$$\mu_{\theta,k} = x_k - \delta_k v_\theta(x_k, s_k, c), \qquad \mu^{\mathrm{T}}_{i,k} = x_k - \delta_k v^{\mathrm{T}}_i(x_k, s_k, c). \tag{14}$$

Because both transitions start from the same stored state, the state cancels:

$$\mu_{\theta,k} - \mu^{\mathrm{T}}_{i,k} = -\delta_k \big(v_\theta(x_k, s_k, c) - v^{\mathrm{T}}_i(x_k, s_k, c)\big). \tag{15}$$

Substituting Eq. 15 into Eq. 13 gives the step-size-weighted velocity objective

$$\mathrm{KL}(p_{\theta,k} \| p^{\mathrm{T}}_{i,k}) = \frac{\delta_k^2}{2\sigma_k^2} \left\| v_\theta(x_k, s_k, c) - v^{\mathrm{T}}_i(x_k, s_k, c) \right\|_F^2. \tag{16}$$

Thus the reverse-KL transition view and velocity-field regression are the same quadratic form under the shared-covariance assumption.

Deterministic ODE transition matching. Following the deterministic regime in DiffusionOPD, our sampler uses direct squared matching of ODE step means rather than a conditional KL:

$$\mathcal{L}_{\mathrm{mean}} = \mathbb{E}_{\substack{i\sim p(i),\ c\sim d_i^{\bar\theta} \\ x_{0:K}\sim q_{\bar\theta}(\cdot|c)}} \left[\frac{1}{2KHd} \sum_{k=0}^{K-1} \left\| \mu_{\theta,k}(x_k, c) - \mathrm{sg}\big[\mu^{\mathrm{T}}_{i,k}(x_k, c)\big] \right\|_F^2 \right]. \tag{17}$$

The stop-gradient keeps the teacher target fixed. Combining Eq. 15 with Eq. 17 shows that each term is proportional to $\delta_k^2 \| v_\theta - v^{\mathrm{T}}_i \|_F^2$; the factor $1/(Hd)$ is only the per-element normalization. If the shared covariance is constant, the omitted σ^{-2} is a global scale absorbed by the loss weight or learning rate. If σ_k varies with k, using Eq. 17 amounts to an explicit choice of uniform mean-space weighting rather than the additional σ_k^{-2} time weighting in Eq. 16.

The deterministic loss is the ODE mean-matching counterpart of the stochastic transition objective. It is not obtained by setting $\sigma_k^2 = 0$ in a Gaussian KL: distinct Dirac transitions do not have a finite KL. Nor does Eq. 17 claim to be an exact KL between the final action distributions. Its on-policy character comes from evaluating both fields at the saved student states x_k; the recorded rollout law is detached, while gradients flow through the current student's prefix and action expert. This is also why the teacher is queried at every stored state rather than used to generate a separate complete action chunk.

C Real-World Evaluation Details

For OPD training, the student performs two rollout rounds. Each round collects 50 trajectories per task, and the resulting on-policy data is used for one training epoch before the next rollout round. The training policy predicts and executes action horizons of 50, with five flow-ODE denoising steps. For the Piper evaluation, the policy predicts an action horizon of 50 but the controller executes only the first 20 actions before requesting the next prediction.

Task	Instruction
Pick Banana	"Grasp the banana from the table and place it on the plate."
Stack Cup	"Stack the yellow cup into the blue cup."
Sort Number	"Sort the numbers."