SHOULD THE ACTION EXPERT TRAIN THE VLM? ACTION-TOKEN SUPERVISION AND GRADIENT ROUTING IN MIXTURE-OF-TRANSFORMER VLAS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision–language–action (VLA) models increasingly pair a pre-trained vision–language model (VLM) with a flow-matching expert that conditions its action predictions on the VLM's key–value representations. Those representations must carry information useful for control, yet the VLM was never trained for the action expert's flow-matching objective. Two signals can address this: gradients back-propagated from the expert's loss (*coupling*), or VLM supervision to predict discrete tokens encoding the same target actions. Prominent training recipes forgo the first, blocking expert-to-VLM gradients (*insulation*) to protect the VLM's existing capabilities, while also supervising it on vision–language data and action tokens. Because vision–language co-training also aims to preserve those capabilities, the additional protection provided by insulation is unclear. At the same time, insulation removes a direct signal about which features continuous action prediction requires. On the other hand, action-token supervision offers an alternative way to learn action-relevant representations, which may explain why insulation succeeds in practice. We disentangle these factors in autonomous driving and manipulation by independently varying action-token supervision and whether the expert's loss updates the VLM. In open-loop driving, insulation is competitive with coupling when action tokens are supervised, but increases trajectory-prediction error by roughly 25% when they are not. When both components are trained jointly from the outset, coupling increases closed-loop driving route progress by approximately 17% with action-token supervision and 40% without it, and similarly improves robot manipulation performance under both settings. These findings show that the value of insulation depends on the VLM's action supervision and the evaluation setting, challenging insulation as a default VLA training recipe.

1 INTRODUCTION

Pretrained vision–language models (VLMs) bring broad perceptual, semantic, and reasoning capabilities to embodied control. Robotic control, however, requires precise continuous outputs at frequencies poorly suited to autoregressive decoding with large VLMs. Many recent vision–language–action (VLA) models therefore use Mixture-of-Transformers-style architectures (Liang et al., 2024), pairing an autoregressive VLM with a smaller diffusion or flow-matching action expert (Black et al., 2024). The VLM processes visual observations and language instructions, while the expert efficiently generates continuous action chunks conditioned on the VLM's per-layer key–value (K/V) representations (Physical Intelligence et al., 2025).

This architecture leaves open a consequential training decision: should the continuous-action loss train only the expert, or also the VLM that produces its conditioning K/V cache? Allowing expert gradients to update the VLM (*coupling*) enables co-adaptation and can teach the backbone which information its representations must preserve for precise control. Yet it also exposes pretrained visual and linguistic features to gradients from a newly introduced, often randomly initialized expert, which may degrade the VLM's semantic and world understanding.

Popular approaches therefore block the action expert's gradients from entering the VLM (*insulation*), either through stop-gradient or frozen-backbone training (Physical Intelligence et al., 2025; Driess et al., 2026). However, full-parameter fine-tuning on a narrow robot training distribution

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.

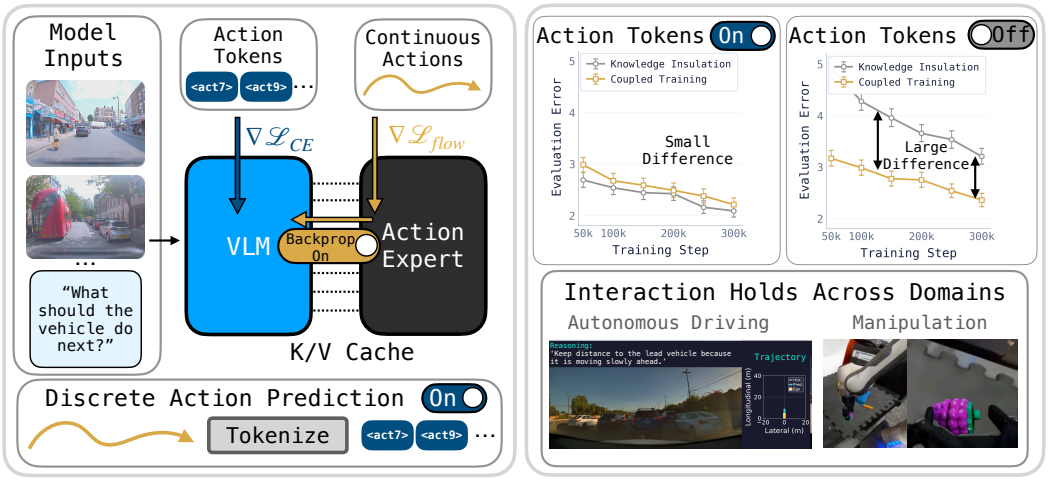

Figure 1: **Action-token supervision shapes the effect of expert-to-VLM gradient flow. Left:** the VLM is supervised to predict discrete action tokens ($\nabla \mathcal{L}_{\text{CE}}$) while a flow-matching expert decodes continuous actions from the VLM's K/V cache. We independently vary whether action tokens are supervised and whether the expert's gradient ($\nabla \mathcal{L}_{\text{flow}}$) updates the VLM. **Right:** open-loop driving evaluation error during training. With action-token supervision, insulated and coupled training differ little; without it, coupling is substantially better. We test this interaction in both autonomous driving and robot manipulation, under open- and closed-loop evaluation.

can degrade pretrained capabilities regardless of gradient routing (Hancock et al., 2025). When vision–language co-training already regularizes against this forgetting, it is unclear whether insulation provides additional protection.

Insulation is also rarely evaluated in isolation. Blocking expert gradients removes one source of action-relevant learning signal, but such systems typically supply another: they train the VLM to predict discrete action tokens (Pertsch et al., 2025), which directly supervise the behavior the expert models. Co-training on general vision–question-answering (VQA) and chain-of-thought examples may preserve the VLM's semantic capabilities, but it does not explicitly encourage representations relevant to control. In the absence of action-token supervision, insulation may therefore eliminate the only objective that directly adapts the VLM for trajectory prediction. Crucially, defining an analogous discrete proxy for continuous prediction targets is a domain-dependent design choice whose effectiveness must be established empirically. It is therefore unclear whether the reported benefits of insulation arise from gradient blocking itself or depend on this auxiliary supervision.

Prior work has not isolated this interaction. Knowledge Insulation reports benefits from insulated training while supervising the VLM with FAST action tokens and vision–language objectives, but does not evaluate insulation without FAST supervision (Driess et al., 2026; Pertsch et al., 2025). In contrast, Wall-OSS-0.5 reports stronger action performance from coupled training even when discrete-action and multimodal objectives are retained (Yu et al., 2026). Because these systems also differ in training recipes, initialization, data, and optimization, their seemingly opposing results cannot isolate the contributions of action-token supervision and expert-to-VLM gradient flow in downstream control or determine how the two mechanisms interact.

Contributions. In this work, we study how a VLA's representations are made action-relevant, and whether the two available signals—the expert's gradient and discrete action-token supervision—substitute for one another. We cross action-token supervision with expert-to-VLM gradient flow in autonomous driving and robot manipulation, under both joint training and training schedules in which one component is adapted first, holding data, architecture, optimization, and semantic co-training fixed. In open-loop driving evaluations, the two routing choices are nearly indistinguishable with action-token supervision, but insulation increases trajectory-prediction error by roughly 25% without it. This pattern holds across training schedules and in both in- and out-of-distribution evaluations. Under joint training, coupling improves closed-loop performance regardless of supervision: driving route progress increases by approximately 17% with action tokens and 40% without them, while manipulation task progress improves under both settings. Together, these results identify

action-token supervision as a factor previously entangled with gradient routing and show that **VLM insulation is a conditional choice, not a default training recipe for training VLAs**.

2 Related Work

Vision–language–action architectures. Early VLA policies incorporated control into pretrained vision–language models (VLMs) by predicting discretized actions autoregressively (Brohan et al., 2023; Kim et al., 2024). Many recent systems instead pair a VLM with a specialized continuous-action module. π_0 introduced a flow-matching expert conditioned on the VLM's per-layer key–value (K/V) cache, an architecture retained or adapted by $\pi_{0.5}$, Wall-OSS-0.5, and MolmoAct2 (Black et al., 2024; Physical Intelligence et al., 2025; Yu et al., 2026; Fang et al., 2026). SmolVLA, GR00T N1, Qwen-RobotManip, CogACT, and DexVLA use related flow- or diffusion-based action decoders with different interfaces to the VLM (Shukor et al., 2025; NVIDIA et al., 2025a; Yuan et al., 2026; Li et al., 2024; Wen et al., 2025). These systems instantiate a broader Mixture-of-Transformers pattern in which modality-specific computation communicates through attention (Liang et al., 2024).

Many autonomous-driving VLAs follow similar designs. EMMA, OpenDriveVLA, and AutoVLA generate future trajectories autoregressively as textual coordinates or discrete tokens, whereas Alpamayo-R1 combines autoregressive physical reasoning and trajectory prediction with a continuous action expert (Hwang et al., 2025; Zhou et al., 2025a;b; NVIDIA et al., 2025c). Related architectures combine autoregressive reasoning with continuous generation for images, video, and other physical-AI modalities (Zhou et al., 2024; NVIDIA et al., 2026). Despite their growing adoption, how continuous experts shape the autoregressive representations through the K/V cache they condition on remains poorly understood. We study this question in a controlled setting.

Action representation and backbone supervision. Action representation determines both how a policy is executed and how control supervision reaches the VLM. Discrete action tokens train the backbone through next-token prediction, while diffusion and flow matching directly model continuous, potentially multimodal action distributions (Chi et al., 2024; Black et al., 2024). FAST compresses action chunks into discrete tokens using a discrete cosine transform, OpenVLA-OFT predicts normalized continuous action chunks in parallel through an MLP head, and HybridVLA combines autoregressive and diffusion-based action prediction (Pertsch et al., 2025; Kim et al., 2025; Liu et al., 2025).

Table 1: Driving and manipulation VLAs that block action-expert gradients from the VLM through freezing or knowledge insulation (Driess et al., 2026). Checkmarks indicate VQA or discrete action-token supervision.

Policy	VQA Predicted	Discrete Actions Predicted
Autonomous Driving		
Alpamayo-R1 (NVIDIA et al., 2025c)	✓	✓
Qwen-Drive-1.0 (Zhou et al., 2026a)	✓	–
SpanVLA (Zhou et al., 2026b)	–	✓
UniDriveVLA (Li et al., 2026)	✓	–
Manipulation		
$\pi_{0.5}$ (Physical Intelligence et al., 2025)	✓	✓
MolmoAct2 (Fang et al., 2026)	✓	✓

Backbone supervision also affects whether pretrained capabilities survive VLA adaptation. End-to-end fine-tuning can distort useful features, motivating parameter-efficient adaptation, language-aligned action representations, and continued vision–language co-training (Kumar et al., 2022; Hancock et al., 2025; Grover et al., 2025; Zha et al., 2026). These approaches address how the VLM is trained, but leave unresolved a key interaction: whether autoregressive supervision over a discrete representation of the continuous action target affects the benefit of allowing gradients from the continuous expert to flow into the VLM backbone.

Gradient routing and training recipes. $\pi_{0.5}$, MolmoAct2, and Alpamayo-R1 adopt Knowledge Insulation: they train the continuous action expert without propagating its gradients into the VLM backbone, while updating the VLM through autoregressive action-token and semantic objectives (Driess et al., 2026; Physical Intelligence et al., 2025; Fang et al., 2026; NVIDIA et al., 2025c). Within this framework, $\pi_{0.5}$ and

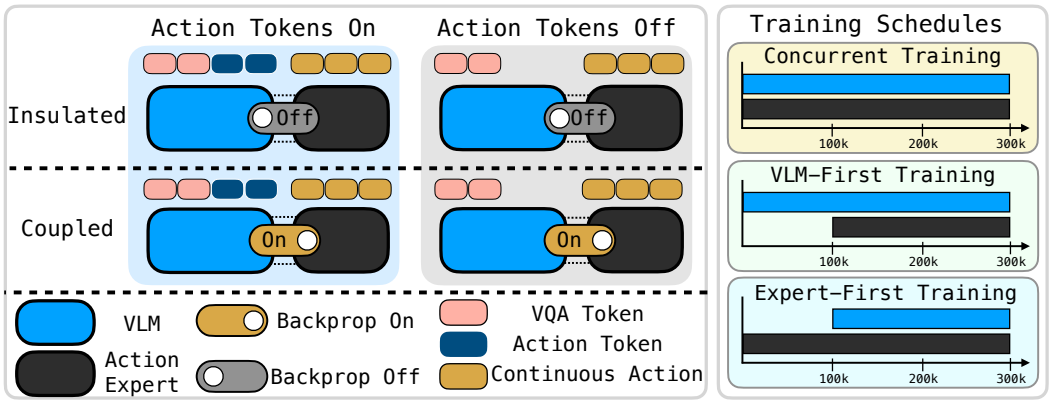

Figure 2: **Action-token supervision, gradient routing, and training schedules. Left**: we cross future-action-token supervision with coupled or insulated expert-to-VLM gradients. **Right**: bars show when the VLM (blue) and action expert (black) are trained. Concurrent training starts both modules together; VLM-first and expert-first training start one module after 100k steps of training the other module. All schedules run for 300k total steps.

MolmoAct2 use a two-stage schedule. The VLM is first adapted through autoregressive prediction of tokenized actions and semantic targets and the continuous expert is introduced afterward. Variants of this staged, insulated training recipe are common in recent VLAs; Table 1 summarizes representative works that insulate or freeze the VLM during action-expert training. By contrast, Wall-OSS-0.5 favors coupled optimization, allowing gradients from the continuous expert to update the VLM while retaining discrete-action and multimodal objectives (Yu et al., 2026). Although these works motivate different gradient-routing choices at a shared VLM–expert interface, their results are not directly comparable because they also differ in training schedule, initialization, supervision, data, and optimization.

The growing use of this hybrid autoregressive–continuous MoT framework has not resolved how models should be trained. $\pi_{0.7}$ scales a FAST-supervised version of the recipe; MolmoAct2 changes its gradient routing between broad post-training and embodiment-specific fine-tuning; and APT finds that action-expert preparation changes the relative performance of insulated and coupled training (Intelligence et al., 2026; Fang et al., 2026; Xu et al., 2026). Together with the contrasting Wall-OSS-0.5 results, these findings suggest that the preferred gradient routing depends on the surrounding training recipe. Related multimodal-pretraining studies similarly show that modalities can reinforce or interfere with one another depending on parameter sharing and training order (Han et al., 2026). Prior work therefore establishes gradient insulation as a consequential design choice, but does not isolate its interaction with autoregressive action supervision. We vary these decisions independently while holding semantic supervision and the VLM–expert architecture fixed.

3 Methodology

In this work, we test whether supervision on discrete future-action tokens changes the effect of propagating gradients from a continuous action expert into the VLM. We first study this question in autonomous driving, using the Alpamayo VLA architecture (NVIDIA et al., 2025c) and the hybrid autoregressive and flow-matching recipe used by $\pi_{0.5}$ and Knowledge Insulation (Physical Intelligence et al., 2025; Driess et al., 2026). We vary future-action-token supervision and expert-to-VLM gradient flow independently, while holding the architecture and semantic supervision fixed.

Mixture-of-Transformers VLA. The driving model pairs a Qwen3-VL-2B-Instruct (Yang et al., 2025) backbone with a same-width, non-causal flow-matching action expert. At each transformer layer, the expert attends to key–value representations produced by the corresponding VLM layer, while the VLM does not attend to the expert's continuous action inputs. The expert predicts K future control pairs, each comprising acceleration and curvature. Starting from the current ego state, these controls are integrated using a discretized unicycle model to recover position and heading

waypoints for low-level control (see A.1 for further details). We retain this architecture for every experiment.

Training includes trajectory examples with and without reasoning annotations. When present, the VLM predicts a concise, scene-grounded rationale identifying relevant road users or scene features and the intended driving response, followed by the future-trajectory start token. In action-token-supervised conditions, the VLM also predicts the subsequent future-trajectory (action) tokens. At evaluation, the VLM always generates a rationale and stops decoding at the start token. The action expert then predicts continuous controls conditioned on the resulting VLM K/V cache. Thus, future-action tokens are never decoded at evaluation and, when supervised, serve only as an auxiliary training signal shaping VLM representations.

The VLM is trained on the standard cross-entropy (CE) objective, while the expert is trained through rectified flow matching. The joint objective is $\mathcal{L} = \mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{flow}}$, where

$$\mathcal{L}_{\text{CE}} = -\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\frac{1}{N}\left(\sum_{i\in\mathcal{I}_{\text{base}}} \log p_\theta(y_i \mid x, y_{<i}) + \mathbf{1}_{\text{action}} \sum_{i\in\mathcal{I}_{\text{action}}} \log p_\theta(y_i \mid x, y_{<i})\right)\right], \quad (1)$$

$$\mathcal{L}_{\text{flow}} = \mathbb{E}_{(x,a)\sim\mathcal{D}_{\text{action}},\, t,\, \epsilon}\left[\left\| v_\phi(a_t, t; \widetilde{H}_\theta) - (a - \epsilon)\right\|_2^2\right], \qquad a_t = ta + (1-t)\epsilon. \quad (2)$$

We sample $b \sim \text{Beta}(1.5, 1.0)$, set $t = 0.999(1-b)$, and sample $\epsilon \sim \mathcal{N}(0, I)$. Parameters θ and ϕ belong to the VLM and action expert, respectively; x is the current timestep's visual observations, language context, and past trajectory history, y is the autoregressive target sequence, and a is the continuous trajectory. The term $\widetilde{H}_\theta$ denotes the VLM K/V representations supplied to the expert. For coupling, $\widetilde{H}_\theta = H_\theta$; for insulation, $\widetilde{H}_\theta = \text{sg}(H_\theta)$, where sg passes the representations unchanged to the expert but blocks gradients from the flow loss to the VLM (Driess et al., 2026). The set $\mathcal{I}_{\text{base}}$ contains always-supervised token, including VQA answers, driving rationales, and future-trajectory delimiters, while $\mathcal{I}_{\text{action}}$ contains future-trajectory value tokens. The indicator $\mathbf{1}_{\text{action}}$ enables or disables direct CE supervision on those value tokens. Both loss terms have unit weight.

Crucially, the discrete action tokens and continuous expert encode the *same future trajectory*. Disabling action-token supervision masks CE only on future-action values, which remain teacher-forced; trajectory delimiters remain supervised, and past-trajectory tokens remain in the input context. We fix N at the pre-mask count of supervised token positions, preserving the weight of other CE terms, and exclude future-action token states from the expert's K/V context.

Gradient routing. Gradient routing changes only the backward pass through the layerwise K/V interface. In the *coupled* condition, the flow loss propagates through this interface and updates both the expert and the VLM. In the *insulated* condition, these gradients are stopped before entering the VLM. Stop-gradient leaves the forward computation unchanged for fixed parameters, so both conditions use the same expert conditioning mechanism and flow-matching target. Figure 2 summarizes which target signals update the VLM in each condition. VQA, reasoning, and structural-token supervision remain active in every condition; past-trajectory tokens remain in the input context.

Training schedules. To test whether training order changes the interaction between action-token supervision and gradient routing, we evaluate all four conditions under three training schedules (see Figure 2). In *concurrent training*, the pretrained VLM and a randomly initialized action expert are trained together from the outset. In *expert-first training*, motivated by APT's expert-preparation strategy (Xu et al., 2026), the expert is trained for 100k steps against a frozen VLM before joint training begins. Neither action-token supervision nor gradient routing affects this first stage because the VLM is frozen. In *VLM-first training*, reflecting the staged adaptation used by $\pi_{0.5}$ and MolmoAct2 (Physical Intelligence et al., 2025; Fang et al., 2026), the VLM is trained alone for 100k steps before a newly initialized expert is introduced. Action-token supervision varies during this first stage, while gradient routing applies only after the expert is introduced. Experts share the same initialization across conditions.

All concurrent-training stages use the same mixture of trajectory and vision–language batches, with trajectory-bearing batches constituting approximately 48.5% of the data. The VLM receives an update on every joint-training step, whereas the expert receives one only on trajectory-bearing steps. During the expert-first warm-up, every step updates the expert because it uses trajectory data ex-

clusively. Thus, matching total optimizer steps across histories does not equalize component-level update budgets; additional details appear in Appendix A.

We compare checkpoints after 300k total optimizer steps, when validation performance had stabilized. Each schedule has the same total step budget: 300k concurrent steps or a 100k single-component stage followed by 200k joint steps. Within each schedule, the four conditions share the starting pretrained checkpoint, expert initialization, batch sampling, and optimizer schedule; action-token supervision and expert-to-VLM gradient flow vary by design.

Our main experiments evaluate these interventions in autonomous driving. In Section 4.3, we repeat the concurrent-training experiment in closed-loop robot manipulation to test whether the findings extend beyond driving. Architecture, action-space, optimization, data, and flow-matching details appear in Appendix A.

4 EXPERIMENTS

In this section, we ask three questions: (**Q1**) Does action-token supervision change the most performative gradient routing mechanism? (**Q2**) Does pretraining the VLM or expert first change that answer? (**Q3**) Are findings consistent across application domains? We address these questions in the three results subsections below.

Open-loop driving. We evaluate on the publicly released Physical AI Autonomous Vehicles dataset, specifically its nominal (named "Public Golden") and out-of-distribution ("OOD") splits, which contain 1,434 and 273 scenes, respectively (NVIDIA Corporation, 2026). For each example, the policy samples six candidate trajectories. Our primary metric is average displacement error (ADE), the mean Euclidean distance between predicted and ground-truth positions over the future trajectory. We report ADE for the first sampled candidate. Minimum ADE (minADE) is the lowest ADE among the six candidates and serves as an oracle selection metric.

Closed-loop driving. We evaluate in AlpaSim, a closed-loop simulator with realistic sensor observations, vehicle dynamics, and traffic scenarios (NVIDIA et al., 2025b). Every condition is evaluated on the same 827 scenes with five stochastic rollouts per scene. We average rollouts within each scene to reduce evaluation variance.

Driving rationales. We use a frozen Lingo-Judge (Marcu et al., 2024) classifier to compare generated reasoning-traces with reference rationales on the Public Golden and OOD benchmarks. The first-sample pass rate is the fraction of examples whose first generated rationale receives a score above 0.5. We also report the fraction for which any of six candidates passes and the mean score across candidates.

Statistical analysis. For open-loop ADE and first-sample rationale pass rates, we compute 95% confidence intervals from 10,000 paired clip-cluster bootstrap resamples. Each resample draws whole clips with replacement and applies the same sampled clips to every condition. These intervals quantify sensitivity to the evaluated clips for fixed checkpoints.

4.1 CONCURRENT TRAINING

We first address Q1 in the simplest setting, where the VLM backbone and a randomly initialized action expert are trained together from the outset. Recall that all four conditions share the same data mixture, initialization, and optimization schedule.

Open-loop driving. Figure 3 shows that the effect of gradient routing depends on action-token supervision. With action-token supervision, insulation slightly lowers ADE on the Public Golden split, while the OOD routing difference is comparatively small. Without action-token supervision, the insulated model has substantially higher ADE in both evaluations. Relative to insulation, coupling reduces ADE by approximately 26% on the Public Golden dataset and 24% OOD.

This roughly one-quarter reduction in mean trajectory error is consistent with action-token supervision and expert gradients providing partially substitutable signals to the VLM. Action-token cross-entropy loss directly supervises the future trajectory even when expert gradients are blocked; without it, the flow loss is the only direct future-action objective that updates the VLM.

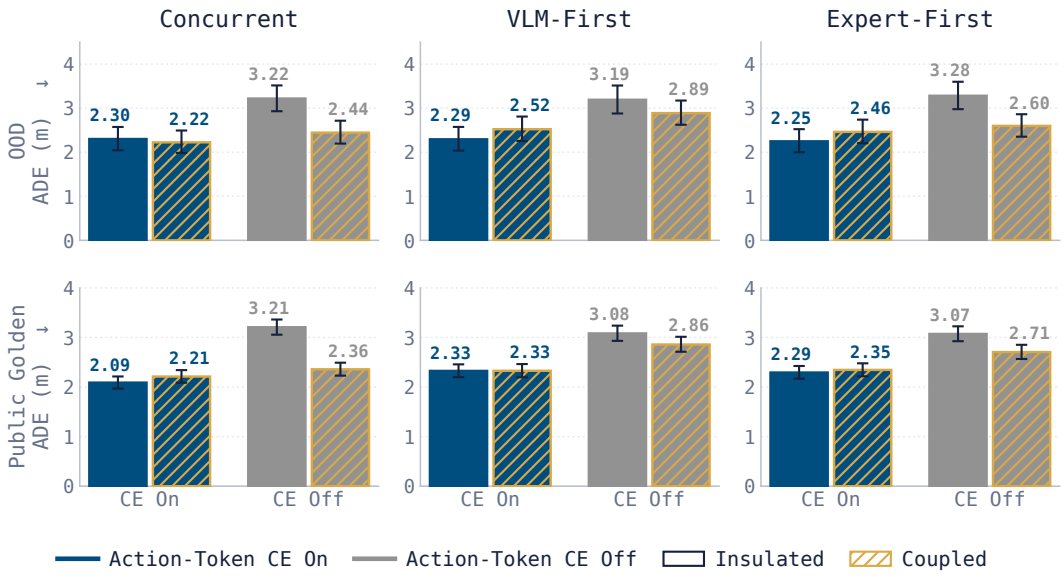

Figure 3: **Open-loop driving accuracy across training histories.** First-sample mean ADE on the Public Golden (1,434 examples) and OOD (273 examples) evaluation splits. Bars show the mean across scenes; error bars show 95% clip-cluster bootstrap confidence intervals. Lower is better.

Table 2: **Generated reasoning-trace quality.** First-sample Lingo-Judge pass rates for each model. In each example, the frozen Lingo-Judge classifier compares the reasoning-trace with a reference rationale (Marcu et al., 2024); an example passes when the judge score exceeds 0.5. Brackets give 95% clip-cluster bootstrap confidence intervals. Top-k is the fraction for which any of six sampled rationales passes, and Avg-k is the mean judge score across the six samples. Results are evaluated on Public Golden and OOD. Within each training schedule, the best metric is bolded. Higher is better.

			Public Golden ($N = 1{,}434$)			**OOD** ($N = 273$)		
Schedule	**Routing**	**Action CE**	**Top-1 [95% CI]**	**Top-k**	**Avg-k**	**Top-1 [95% CI]**	**Top-k**	**Avg-k**
Concurrent	Insulated	On	0.647 [0.621,0.673]	0.808	**0.630**	0.414 [0.355,0.474]	0.659	0.466
Concurrent	Coupled	On	**0.649** [0.623,0.674]	**0.823**	0.629	**0.425** [0.364,0.487]	**0.689**	**0.471**
Concurrent	Insulated	Off	0.566 [0.539,0.593]	0.767	0.573	0.363 [0.305,0.423]	0.652	0.435
Concurrent	Coupled	Off	0.635 [0.610,0.661]	0.811	0.622	0.348 [0.292,0.406]	0.685	0.450
VLM-First	Insulated	On	**0.639** [0.614,0.665]	0.807	**0.626**	0.392 [0.335,0.452]	0.652	0.462
VLM-First	Coupled	On	0.635 [0.609,0.661]	**0.814**	0.625	0.403 [0.344,0.464]	0.641	0.458
VLM-First	Insulated	Off	0.630 [0.603,0.657]	0.800	0.616	**0.447** [0.386,0.508]	**0.711**	**0.465**
VLM-First	Coupled	Off	0.613 [0.585,0.639]	0.798	0.611	0.396 [0.339,0.452]	0.667	0.451
Expert-First	Insulated	On	0.644 [0.618,0.670]	**0.819**	0.626	**0.458** [0.399,0.518]	**0.700**	**0.470**
Expert-First	Coupled	On	**0.649** [0.623,0.675]	0.811	**0.628**	0.388 [0.331,0.447]	0.659	0.453
Expert-First	Insulated	Off	0.620 [0.594,0.646]	0.804	0.610	0.407 [0.348,0.466]	0.659	0.450
Expert-First	Coupled	Off	0.612 [0.585,0.639]	0.798	0.611	0.392 [0.333,0.451]	0.692	0.459

Generated reasoning-trace quality. We use the first-sample Lingo-Judge pass rate as a diagnostic of the language generated during concurrent training (Table 2). With action-token supervision, insulated and coupled models obtain nearly identical pass rates on Public Golden (64.7% and 64.9%, respectively). Without action-token supervision, the coupled model has a higher observed pass rate on Public Golden (63.5% versus 56.6%), while the OOD routing difference is small and points in the opposite direction. These results demonstrate that *insulation provides little additional protection of vision–language capabilities when VQA-style co-training is present*.

Closed-loop driving. In concurrent training, coupling raises AlpaSim route progress from 0.600 to 0.703 with action-token supervision and from 0.425 to 0.601 without it (Table 3). The latter is a 17.6 percentage-point increase, or about 41% relative, on the same evaluation scenes. Coupling without tokens also reaches approximately the same route progress as insulated training with tokens. Thus, allowing the expert's loss to update the VLM yields a substantial gain in interactive driving, with the largest observed gain when action-token supervision is absent.

Thus, Q1 has a conditional answer: without action-token supervision, coupling substantially improves open- and closed-loop driving; with supervision, it still improves closed-loop progress, while the open-loop difference is small. The most important factor is that the VLM receives a direct action learning signal, supplied by either action-token supervision or expert-gradient coupling.

4.2 Dependence on Training History

To address Q2, we next test whether the supervision–routing pattern depends on training history by first adapting either the action expert or the VLM for 100k steps. We align staged and concurrent checkpoints by total optimizer steps; component-level gradient exposure nevertheless differs across histories by design (see Table 6 in the Appendix).

Open-loop driving. The supervision–routing pattern observed under concurrent training persists when either module is adapted first (Figure 3). Across both evaluation datasets, insulation increases mean ADE without action-token supervision in every history, while its effect with supervision is smaller and depends on training order. Thus, training order changes the size of the routing effect but does not remove its dependence on action-token supervision.

Table 3: **Closed-loop route progress across training histories.** Mean AlpaSim route progress on the same 827 scenes, with five stochastic rollouts per scene averaged within each scene. The best value in each row is bolded and the second-best is underlined. Higher is better.

	Insulated		Coupled	
History	**Tokens on**	**Tokens off**	**Tokens on**	**Tokens off**
Concurrent	0.600	0.425	**0.703**	<u>0.601</u>
VLM-First	**0.648**	0.555	<u>0.646</u>	0.587
Expert-First	0.597	0.538	**0.625**	<u>0.604</u>

Generated reasoning-trace quality. The staged Lingo-Judge results do not mirror the trajectory-ADE pattern (Table 2). Without action-token supervision, insulated models have higher observed first-sample pass rates on Public Golden under both staged schedules, even though coupled models have lower ADE. Routing differences on OOD vary. Thus, improvements in continuous trajectory prediction do not necessarily coincide with improvements in generated reasoning traces; neither gradient routing choice uniformly favors both outcomes.

Closed-loop driving. We evaluate both staged histories on the same AlpaSim scenes using the concurrent-training protocol from Section 4.1. Table 3 shows that coupling increases observed route progress without action-token supervision after both Expert-First and VLM-First training. With supervision, coupling raises progress after Expert-First training but makes little difference after VLM-First training. Coupling without tokens roughly matches insulated training with tokens in the Expert-First history, but falls below it in the VLM-First history. Concurrent coupled training with tokens achieves the highest observed progress overall (0.703). Full staged competence and safety results appear in Appendix Tables 11 and 12.

Thus, for Q2, training order changes the size of the routing effect, but coupling improves open-loop accuracy without action-token supervision under every schedule. Importantly, *the greatest absolute performance in action-prediction across all training histories is achieved by concurrent training*, which stands in contrast to nominal VLM-First recipes adopted by prior works (Physical Intelligence et al., 2025; Fang et al., 2026).

4.3 Transfer to Robot Manipulation

Finally, to address Q3, we repeat the concurrent comparison in robot manipulation using a $\pi_{0.5}$ re-implementation with a PaliGemma-3B VLM backbone and a flow-matching action expert (Beyer et al., 2024). We use the OpenPi implementation (Black et al., 2024) and substitute the FAST implementation from Zha et al. (2026). Training batches are divided equally between VQA and robot data. The VQA mixture comprises VQAv2 question answering (35%) (Goyal et al., 2017), COCO captioning (20%) (Chen et al., 2015) and localization (20%) (Lin et al., 2015), as well as Visual Genome question answering (15%) and region tasks (10%) (Krishna et al., 2016). The robot mixture contains 90% DROID (Khazatsky et al., 2025) and 10% PolaRiS training data (Jain et al., 2025); supervised conditions represent actions using FAST tokens (Pertsch et al., 2025).

We evaluate all policies in PolaRiS, a simulator designed for relative policy ranking that permits repeated evaluations from matched initial states (Jain et al., 2025). We train each model for 400k steps and evaluate it in three environments. 'BlockStackKitchen' requires placing and stacking blocks on a green tray amid distractors; 'FoodBussing' requires identifying food items and putting them into a bowl; and 'PanClean' requires using a yellow sponge on a blue-handled frying pan in a cluttered kitchen. For each condition, we run 50 rollouts from matched initial states in each environment and average results. Because binary success is sparse, we report normalized task progress, weighting the three environments equally.

The observed gain is larger without FAST: progress rises from 10.3% to 32.9%, compared with 19.5% to 30.7% with FAST (Table 4). Coupled training without FAST achieves the highest mean progress among our four conditions, about 1.7 times that of insulated training with FAST. For context, CoVeR reports 40.0% progress for a published $\pi_{0.5}$ checkpoint under the same PolaRiS evaluation protocol (Kwok et al., 2026). That checkpoint was pretrained on more than 10,000 hours of robot data (Physical Intelligence et al., 2025), whereas our robot training draws primarily from DROID, a 350-hour dataset (Khazatsky et al., 2025). Thus, the answer to Q3 is yes: coupling improves observed closed-loop manipulation progress with and without FAST supervision, despite insulation being the training recipe used by $\pi_{0.5}$ (Physical Intelligence et al., 2025).

Table 4: **Closed-loop manipulation progress.** Equal-weight mean normalized task progress across three PolaRiS environments after 400k training steps. Bold indicates the best of our four conditions. The published $\pi_{0.5}$ reference is from CoVeR (Kwok et al., 2026) and uses its evaluation protocol. Higher is better.

Condition	**Progress ↑**
Insulated / FAST on	19.5%
Coupled / FAST on	30.7%
Insulated / FAST off	10.3%
Coupled / FAST off	**32.9%**
Published $\pi_{0.5}$	40.0%

5 Discussion and Conclusion

Insulating the VLM from action-expert gradients is often motivated by preserving its pretrained capabilities. Our findings show that the consequences of this choice depend on the other signals available to the VLM. Action-token supervision directly trains it to predict future actions, which helps explain why blocking expert gradients has little effect on open-loop driving error when that supervision is present. Without action tokens, insulation substantially worsens trajectory prediction across evaluation sets and training schedules. Yet under concurrent training, coupling improves closed-loop route progress even with action-token supervision, and the same pattern appears in manipulation. The Lingo-Judge results show no consistent advantage for insulation in generated reasoning-trace quality under vision–language co-training, although this diagnostic cannot assess broader pretrained capabilities.

Together, these findings suggest that action-token supervision and expert gradients can partly substitute for each other in learning representations useful for control. With continued vision–language co-training, insulation does not appear necessary to preserve downstream reasoning quality. Moreover, concurrently training the VLM and action expert from the outset is a strong, simple starting point. In summary, *VLM insulation is a training decision, not a default VLA training recipe.*

Limitations. We study one model family per domain, and our closed-loop evaluations use simulated driving scenes and three simulated manipulation tasks. We do not test the manipulation policies on physical robots, an important next step for assessing whether the observed gains transfer to real-world control. Although our bootstrap intervals capture variation across evaluation clips for fixed checkpoints, compute constraints limited us to one training run per condition, leaving training-seed variation unmeasured. **Future work.** We are excited to test whether this interaction extends to more embodiments, world-action models, and video-prediction world models.

AI Use Statement

We used AI tools (Claude Code and Codex) to assist with coding, conducting experiments (running evaluations), plot generation, and polishing the manuscript's language. We developed the research idea, conducted the related-work review, drafted all sections, and created all figures. We are responsible for the experimental results and the paper's claims.

REFERENCES

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, 2024. URL https://arxiv.org/abs/2407.07726.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-2: Vision-language-action models transfer web knowledge to robotic control, 2023. URL https://arxiv.org/abs/2307.15818.

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollar, and C. Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server, 2015. URL https://arxiv.org/abs/1504.00325.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion, 2024. URL https://arxiv.org/abs/2303.04137.

Danny Driess, Jost Springenberg, Brian Ichter, Lili Yu, Adrian Li-Bell, Karl Pertsch, Allen Ren, Homer Walke, Quan Vuong, Lucy Xiaoyang Shi, et al. Knowledge insulating vision-language-action models: Train fast, run fast, generalize better. *Advances in Neural Information Processing Systems*, 38:102867–102888, 2026.

Haoquan Fang, Jiafei Duan, Donovan Clay, Sam Wang, Shuo Liu, Weikai Huang, Xiang Fan, Wei-Chuan Tsai, Shirui Chen, Yi Ru Wang, Shanli Xing, Jaemin Cho, Jae Sung Park, Ainaz Eftekhar, Peter Sushko, Karen Farley, Angad Wadhwa, Cole Harrison, Winson Han, Ying-Chun Lee, Eli VanderBilt, Rose Hendrix, Suveen Ellawela, Lucas Ngoo, Joyce Chai, Zhongzheng Ren, Ali Farhadi, Dieter Fox, and Ranjay Krishna. Molmoact2: Action reasoning models for real-world deployment, 2026. URL https://arxiv.org/abs/2605.02881.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering, 2017. URL https://arxiv.org/abs/1612.00837.

Shresth Grover, Akshay Gopalkrishnan, Bo Ai, Henrik I. Christensen, Hao Su, and Xuanlin Li. Enhancing generalization in vision-language-action models by preserving pretrained representations, 2025. URL https://arxiv.org/abs/2509.11417.

Junlin Han, Shengbang Tong, David Fan, Minghao Chen, Philip Torr, Filippos Kokkinos, and Mike Lewis. Towards physics of multimodal pretraining: Knowledge flow, modality synergy, early unification, and recipes, 2026. URL https://arxiv.org/abs/2608.05000.

Asher J. Hancock, Xindi Wu, Lihan Zha, Olga Russakovsky, and Anirudha Majumdar. Actions as language: Fine-tuning vlms into vlas without catastrophic forgetting, 2025. URL https://arxiv.org/abs/2509.22195.

Jyh-Jing Hwang, Runsheng Xu, Hubert Lin, Wei-Chih Hung, Jingwei Ji, Kristy Choi, Di Huang, Tong He, Paul Covington, Benjamin Sapp, Yin Zhou, James Guo, Dragomir Anguelov, and Mingxing Tan. Emma: End-to-end multimodal model for autonomous driving, 2025. URL https://arxiv.org/abs/2410.23262.

Physical Intelligence, Bo Ai, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Greg Balke, Kevin Black, George Bokinsky, Shihao Cao, Thomas Charbonnier, Vedant Choudhary, Foster Collins, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Maitrayee Dhaka, Jared DiCarlo, Danny Driess, Michael Equi, Adnan Esmail, Yunhao Fang, Chelsea Finn, Catherine Glossop, Thomas Godden, Ivan Goryachev, Lachlan Groom, Haroun Habeeb, Hunter Hancock, Karol Hausman, Gashon Hussein, Victor Hwang, Brian Ichter, Connor Jacobsen, Szymon Jakubczak, Rowan Jen, Tim Jones, Gregg Kammerer, Ben Katz, Liyiming Ke, Mairbek Khadikov, Chandra Kuchi, Marinda Lamb, Devin LeBlanc, Brendon LeCount, Sergey Levine, Xinyu Li, Adrian Li-Bell, Vladislav Lialin, Zhonglin Liang, Wallace Lim, Yao Lu, Enyu Luo, Vishnu Mano, Nandan Marwaha, Aikys Mongush, Liam Murphy, Suraj Nair, Tyler Patterson, Karl Pertsch, Allen Z. Ren, Gavin Schelske, Charvi Sharma, Baifeng Shi, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, Will Stoeckle, Jiaming Tang, Jimmy Tanner, Shalom Tekeste, Marcel Torne, Kyle Vedder, Quan Vuong, Anna Walling, Haohuan Wang, Jason Wang, XuDong Wang, Chris Whalen, Samuel Whitmore, Blake Williams, Charles Xu, Sukwon Yoo, Lili Yu, Wuming Zhang, Zhuoyang Zhang, and Ury Zhilinsky. $\pi_{0.7}$: a steerable generalist robotic foundation model with emergent capabilities, 2026. URL https://arxiv.org/abs/2604.15483.

Arhan Jain, Mingtong Zhang, Kanav Arora, William Chen, Marcel Torne, Muhammad Zubair Irshad, Sergey Zakharov, Yue Wang, Sergey Levine, Chelsea Finn, et al. Polaris: Scalable real-to-sim evaluations for generalist robot policies. *arXiv preprint arXiv:2512.16881*, 2025.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, Peter David Fagan, Joey Hejna, Masha Itkina, Marion Lepert, Yecheng Jason Ma, Patrick Tree Miller, Jimmy Wu, Suneel Belkhale, Shivin Dass, Huy Ha, Arhan Jain, Abraham Lee, Youngwoon Lee, Marius Memmel, Sungjae Park, Ilija Radosavovic, Kaiyuan Wang, Albert Zhan, Kevin Black, Cheng Chi, Kyle Beltran Hatch, Shan Lin, Jingpei Lu, Jean Mercat, Abdul Rehman, Pannag R Sanketi, Archit Sharma, Cody Simpson, Quan Vuong, Homer Rich Walke, Blake Wulfe, Ted Xiao, Jonathan Heewon Yang, Arefeh Yavary, Tony Z. Zhao, Christopher Agia, Rohan Baijal, Mateo Guaman Castro, Daphne Chen, Qiuyu Chen, Trinity Chung, Jaimyn Drake, Ethan Paul Foster, Jensen Gao, Vitor Guizilini, David Antonio Herrera, Minho Heo, Kyle Hsu, Jiaheng Hu, Muhammad Zubair Irshad, Donovon Jackson, Charlotte Le, Yunshuang Li, Kevin Lin, Roy Lin, Zehan Ma, Abhiram Maddukuri, Suvir Mirchandani, Daniel Morton, Tony Nguyen, Abigail O'Neill, Rosario Scalise, Derick Seale, Victor Son, Stephen Tian, Emi Tran, Andrew E. Wang, Yilin Wu, Annie Xie, Jingyun Yang, Patrick Yin, Yunchu Zhang, Osbert Bastani, Glen Berseth, Jeannette Bohg, Ken Goldberg, Abhinav Gupta, Abhishek Gupta, Dinesh Jayaraman, Joseph J Lim, Jitendra Malik, Roberto Martín-Martín, Subramanian Ramamoorthy, Dorsa Sadigh, Shuran Song, Jiajun Wu, Michael C. Yip, Yuke Zhu, Thomas Kollar, Sergey Levine, and Chelsea Finn. Droid: A large-scale in-the-wild robot manipulation dataset, 2025. URL https://arxiv.org/abs/2403.12945.

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

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.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Fei-Fei Li. Visual genome: Connecting language and vision using crowdsourced dense image annotations, 2016. URL https://arxiv.org/abs/1602.07332.

Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution, 2022. URL https://arxiv.org/abs/2202.10054.

Jacky Kwok, Xilun Zhang, Mengdi Xu, Yuejiang Liu, Azalia Mirhoseini, Chelsea Finn, and Marco Pavone. Scaling verification can be more effective than scaling policy learning for vision-language-action alignment, 2026. URL https://arxiv.org/abs/2602.12281.

Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, Xiaofan Wang, Bei Liu, Jianlong Fu, Jianmin Bao, Dong Chen, Yuanchun Shi, Jiaolong Yang, and Baining Guo. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation, 2024. URL https://arxiv.org/abs/2411.19650.

Yongkang Li, Lijun Zhou, Sixu Yan, Bencheng Liao, Tianyi Yan, Kaixin Xiong, Long Chen, Hongwei Xie, Bing Wang, Guang Chen, Hangjun Ye, Wenyu Liu, Haiyang Sun, and Xinggang Wang. Unidrivevla: Unifying understanding, perception, and action planning for autonomous driving, 2026. URL https://arxiv.org/abs/2604.02190.

Weixin Liang, Lili Yu, Liang Luo, Srinivasan Iyer, Ning Dong, Chunting Zhou, Gargi Ghosh, Mike Lewis, Wen-tau Yih, Luke Zettlemoyer, et al. Mixture-of-transformers: A sparse and scalable architecture for multi-modal foundation models. *arXiv preprint arXiv:2411.04996*, 2024.

Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2015. URL https://arxiv.org/abs/1405.0312.

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

Ana-Maria Marcu, Long Chen, Jan Hünermann, Alice Karnsund, Benoit Hanotte, Prajwal Chidananda, Saurabh Nair, Vijay Badrinarayanan, Alex Kendall, Jamie Shotton, Elahe Arani, and Oleg Sinavski. Lingoqa: Visual question answering for autonomous driving, 2024. URL https://arxiv.org/abs/2312.14115.

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

NVIDIA, Yulong Cao, Riccardo de Lutio, Sanja Fidler, Guillermo Garcia Cobo, Zan Gojcic, Maximilian Igl, Boris Ivanovic, Peter Karkus, Janick Martinez Esturo, Marco Pavone, Aaron Smith, Ellie Tanimura, Michal Tyszkiewicz, Michael Watson, Qi Wu, and Le Zhang. Alpasim: A modular, lightweight, and data-driven research simulator for autonomous driving, October 2025b. URL https://github.com/NVlabs/alpasim.

NVIDIA, Yan Wang, Wenjie Luo, Junjie Bai, Yulong Cao, Tong Che, Ke Chen, Yuxiao Chen, Jenna Diamond, Yifan Ding, Wenhao Ding, et al. Alpamayo-R1: Bridging reasoning and action prediction for generalizable autonomous driving in the long tail. *arXiv preprint arXiv:2511.00088*, 2025c.

NVIDIA, :, Aditi, Niket Agarwal, Arslan Ali, Jon Allen, Martin Antolini, Adeline Aubame, Alisson Azzolini, Junjie Bai, Maciej Bala, Yogesh Balaji, Josh Bapst, Aarti Basant, Mukesh Beladiya, Mohammad Qazim Bhat, Zaid Pervaiz Bhat, Dan Blick, Vanni Brighella, Han Cai, Tiffany Cai, Eric Cameracci, Jiaxin Cao, Yulong Cao, Mark Carlson, Carlos Casanova, Ting-Yun Chang, Yan Chang, Yu-Wei Chao, Prithvijit Chattopadhyay, Roshan Chaudhari, Chieh-Yun Chen, Junyu

Chen, Ke Chen, Qizhi Chen, Wenkai Chen, Xiaotong Chen, Yu Chen, An-Chieh Cheng, Click Cheng, Xiu Chia, Jeana Choi, Chaeyeon Chung, Wenyan Cong, Yin Cui, Magdalena Dadela, Nalin Dadhich, Wenliang Dai, Joyjit Daw, Alperen Degirmenci, Rodrigo Vieira Del Monte, Robert Denomme, Sameer Dharur, Marco Di Lucca, Ke Ding, Wenhao Ding, Yifan Ding, Yuzhu Dong, Nicole Drumheller, Yilun Du, Aigul Dzhumamuratova, Aleksandr Efitorov, Hamid Eghbalzadeh, Naomi Eigbe, Imad El Hanafi, Hassan Eslami, Benedikt Falk, Jiaojiao Fan, Jim Fan, Amol Fasale, Sergiy Fefilatyev, Liang Feng, Francesco Ferroni, Sanja Fidler, Xiao Fu, Vikram Fugro, Prashant Gaikwad, TJ Galda, Katelyn Gao, Yihuai Gao, Wenhang Ge, Sreyan Ghosh, Arushi Goel, Vivek Goel, Akash Gokul, Rama Govindaraju, Jinwei Gu, Miguel Guerrero, Elfie Guo, Aryaman Gupta, Siddharth Gururani, Hugo Hadfield, Song Han, Ankur Handa, Zekun Hao, Mohammad Harrim, Ali Hassani, Nathan Hayes-Roth, Yufan He, Chris Helvig, Cyrus Hogg, Madison Huang, Michael Huang, Sophia Huang, Yufan Huang, Jacob Huffman, DeLesley Hutchins, Suneel Indupuru, Boris Ivanovic, Arihant Jain, Joel Jang, Ryan Ji, Yanan Jian, Dongfu Jiang, Jingyi Jin, Atharva Joshi, Nikhilesh Joshi, Pranjali Joshi, Andy Ju, Jaehun Jung, Weiwei Kang, Scott Kassekert, Jan Kautz, Ashna Khetan, Julia Kiczka, Slawek Kierat, Gwanghyun Kim, Kuno Kim, Sunny Kim, Kezhi Kong, Xin Kong, Zhifeng Kong, Tomasz Kornuta, Egor Krivov, Hui Kuang, Saurav Kumar, Chia-Wen Kuo, George Kurian, Wojciech Kutak, JF Lafleche, Himangshu Lahkar, Omar Laymoun, Jayjun Lee, Sanggil Lee, Gabriele Leone, Boyi Li, Freya Li, Jiajun Li, Jinfeng Li, Ling Li, Pengcheng Li, Shangru Li, Tingle Li, Xiaolong Li, Xuan Li, Zhaoshuo Li, Zhiqi Li, Hao Liang, Maosheng Liao, Chen-Hsuan Lin, Tsung-Yi Lin, Ming-Yu Liu, Sifei Liu, Zihan Liu, Hai Loc Lu, Xiangyu Lu, Alice Luo, Ruipu Luo, Wenjie Luo, Jiangran Lyu, Martin Ding Ma, Nic Ma, Qianli Ma, Dawid Majchrowski, Louis Marcoux, Miguel Martin, Qing Miao, Ashkan Mirzaei, Shreyas Misra, Kaichun Mo, Durra Mohsin, Hyejin Moon, Pawel Morkisz, Saeid Motiian, Kirill Motkov, Seungjun Nah, Yashraj Narang, Deepak Narayanan, Thabang Ngazimbi, Julian Ouyang, Shubham Pachori, David Page, Yatian Pang, Sehwi Park, Mahesh Patekar, Mostofa Patwary, Marco Pavone, Trung Pham, Wei Ping, Soha Pouya, Shrimai Prabhumoye, Varun Praveen, Delin Qu, Hesam Rabeti, Morteza Ramezanali, Marilyn Reeb, Xuanchi Ren, Kristen Rumley, Wojciech Rymer, Jun Saito, Yeongho Seol, John Shao, Piyush Shekdar, Tianwei Shen, Humphrey Shi, Min Shi, Stella Shi, Kevin Shih, Mohammad Shoeybi, Mateusz Sieniawski, Shuran Song, Alexander Sotelo, Amir Sotoodeh, Sunil Srinivasa, Vignesh Srinivasakumar, Bartosz Stefaniak, Rahul Heinrich Steiger, Shangkun Sun, Jiaxiang Tang, Shitao Tang, Yangyang Tang, Yue Tang, Tolou Tavakkoli, Kayley Ting, Krzysztof Tomala, Wei-Cheng Tseng, Jibin Varghese, Sergei Vasilev, Thomas Volk, Raju Wagwani, Roger Waleffe, Andrew Z. Wang, Boxiang Wang, Haoxiang Wang, Qiao Wang, Shihao Wang, Shijie Wang, Ting-Chun Wang, Yan Wang, Yu Wang, Rohit Watve, David Wehr, Fangyin Wei, Xinshuo Weng, Jay Zhangjie Wu, Kedi Wu, Hongchi Xia, Summer Xiao, Tianjun Xiao, Kevin Xie, Daguang Xu, Jiashu Xu, Mengyao Xu, Ruqing Xu, Xingqian Xu, Yao Xu, Dinghao Yang, Dong Yang, Hans Yang, Xiaodong Yang, Xuning Yang, Yichu Yang, Yurong You, Zhiding Yu, Hao Yuan, Simon Yuen, Xiaohui Zeng, Pengcuo Zeren, Cindy Zha, Haotian Zhang, Jenny Zhang, Jing Zhang, Liangkai Zhang, Paris Zhang, Shun Zhang, Xuanmeng Zhang, Zhizheng Zhang, Ann Zhao, Yilin Zhao, Yuliya Zhautouskaya, Charles Zhou, Fengzhe Zhou, Shilin Zhu, Yuke Zhu, Dima Zhylko, and Artur Zolkowski. Cosmos 3: Omnimodal world models for physical ai, 2026. URL https://arxiv.org/abs/2606.02800.

NVIDIA Corporation. PhysicalAI-Autonomous-Vehicles. Hugging Face dataset, 2026. URL https://huggingface.co/datasets/nvidia/PhysicalAI-Autonomous-Vehicles. Version 26.09. Accessed September 23, 2026.

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. *arXiv preprint arXiv:2501.09747*, 2025.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

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

Junjie Wen, Yichen Zhu, Jinming Li, Zhibin Tang, Chaomin Shen, and Feifei Feng. Dexvla: Vision-language model with plug-in diffusion expert for general robot control, 2025. URL https://arxiv.org/abs/2502.05855.

Kechun Xu, Zhenjie Zhu, Anzhe Chen, Rong Xiong, and Yue Wang. Apt: Action expert pretraining improves instruction generalization of vision-language-action policies, 2026. URL https://arxiv.org/abs/2606.12366.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Ryan Yu, Pushi Zhang, Starrick Liu, Brae Liu, Miracle Kang, Shalfun Li, Lights Shi, Ellie Ma, Ping Yang, Chris Pan, Jerry Chen, Dongxiu Liu, Rain Sun, Miles Guo, Byron Zhang, Hugo Zhou, Zach Xu, Vincent Chen, Harrison Huang, James Wang, Dance Kuzi, Andy Zhai, Hang Su, Roy Gan, Lucy Liang, Hao Wang, and Qian Wang. Wall-oss-0.5 technical report, 2026. URL https://arxiv.org/abs/2605.30877.

Haoqi Yuan, Zhixuan Liang, Anzhe Chen, Ye Wang, Haoyang Li, Pei Lin, Yiyang Huang, Zixing Lei, Tong Zhang, Jiazhao Zhang, Jie Zhang, Jingyang Fan, Gengze Zhou, Qihang Peng, Chenxu Lv, Xiaoyue Chen, An Yang, Fei Huang, Junyang Lin, Dayiheng Liu, Jingren Zhou, Chenfei Wu, and Xiong-Hui Chen. Qwen-robotmanip technical report: Alignment unlocks scale for robotic manipulation foundation models, 2026. URL https://arxiv.org/abs/2606.17846.

Lihan Zha, Asher J. Hancock, Mingtong Zhang, Tenny Yin, Yixuan Huang, Dhruv Shah, Allen Z. Ren, and Anirudha Majumdar. Lap: Language-action pre-training enables zero-shot cross-embodiment transfer, 2026. URL https://arxiv.org/abs/2602.10556.

Chunting Zhou, Lili Yu, Arun Babu, Kushal Tirumala, Michihiro Yasunaga, Leonid Shamis, Jacob Kahn, Xuezhe Ma, Luke Zettlemoyer, and Omer Levy. Transfusion: Predict the next token and diffuse images with one multi-modal model, 2024. URL https://arxiv.org/abs/2408.11039.

Xin Zhou, Zongchuang Zhao, Zhibo Yang, Mingsheng Li, Humen Zhong, Shuai Bai, Du Chu, Ruizhe Chen, Zhaohai Li, Jun Tang, Qiuyue Wang, Mingkun Yang, Jiazhao Zhang, Dayiheng Liu, Dingkang Liang, and Xiang Bai. Qwen-drive-1.0: An initial step towards a vision-language foundation model for autonomous driving, 2026a. URL https://arxiv.org/abs/2609.00111.

Xingcheng Zhou, Xuyuan Han, Feng Yang, Yunpu Ma, Volker Tresp, and Alois Knoll. Opendrivevla: Towards end-to-end autonomous driving with large vision language action model, 2025a. URL https://arxiv.org/abs/2503.23463.

Zewei Zhou, Tianhui Cai, Seth Z. Zhao, Yun Zhang, Zhiyu Huang, Bolei Zhou, and Jiaqi Ma. Autovla: A vision-language-action model for end-to-end autonomous driving with adaptive reasoning and reinforcement fine-tuning, 2025b. URL https://arxiv.org/abs/2506.13757.

Zewei Zhou, Ruining Yang, Xuewei, Qi, Yiluan Guo, Sherry X. Chen, Tao Feng, Kateryna Pistunova, Yishan Shen, Lili Su, and Jiaqi Ma. Spanvla: Efficient action bridging and learning from negative-recovery samples for vision-language-action model, 2026b. URL https://arxiv.org/abs/2604.19710.

A Implementation and Training Details

This section reports the architecture, data mixtures, optimizer settings, and training schedules for the driving and manipulation experiments.

A.1 Driving model and action representation

The driving policy uses a Qwen3-VL-2B-Instruct (Yang et al., 2025) backbone and a non-causal flow-matching action expert. At each layer, the expert attends to the K/V representations of the corresponding VLM layer. Insulation detaches these representations before the expert reads them, leaving their forward values unchanged. Under coupling, the K/V representations remain attached and the flow loss can update the VLM.

The text tower and action expert use 128-dimensional attention heads. The vision tower outputs 2,048-dimensional features from 16×16 patches with spatial merge size two. The action expert uses a two-layer waypoint input projection with width 512 and 20 Fourier features, followed by a linear output projection. The VLM context length is 16,384 tokens.

The expert represents the 6.4 s trajectory as 64 acceleration–curvature control pairs sampled at 10 Hz and converts them to pose waypoints using discretized unicycle dynamics. The VLM receives quantized ego-history tokens and, when action-token supervision is enabled, predicts a 128-token discretization of the same future control sequence.

When future-action CE is disabled, the 128 future value-token targets are masked, but their teacher-forced input tokens remain in the sequence. Past-trajectory values and the future-trajectory boundary markers remain supervised. The CE denominator is the number of supervised positions before masking, so removing future-action CE does not increase the weight of the retained autoregressive targets. The expert-facing K/V cache ends immediately after the `<future_start>` marker: it contains neither future value-token states nor the `<future_end>` state. Consequently, the expert cannot read the teacher-forced future values.

Table 5: **Driving optimization settings.** Shared across conditions within each training history.

Setting	Value
Optimizer	Fused AdamW
(β_1, β_2)	$(0.9, 0.95)$
Adam ϵ	10^{-8}
Weight decay	0.1
Vision LR	10^{-5}
VLM text/head LR	10^{-4}
Expert LR	4×10^{-4}
LR schedule	Cosine, 400k steps
Minimum LR factor	0.01
Global batch	128
Gradient clipping	1.0
Model precision	Bfloat16
Optimizer master	Float32

More precisely, let $\mathcal{F}$ contain future-trajectory value-token target positions, let $\mathcal{O}$ contain the other supervised positions, and let ℓ_i be next-token CE at position i. For $p = 1$ when action-token supervision is on and $p = 0$ when it is off, the implemented loss is

$$\mathcal{L}_{\mathrm{CE}}^{(p)} = \frac{\sum_{i \in \mathcal{O}} \ell_i + p \sum_{i \in \mathcal{F}} \ell_i}{|\mathcal{O}| + |\mathcal{F}|}. \tag{3}$$

The non-action CE contribution therefore has the same scale under both settings. The flow and CE objectives each have unit weight on trajectory-bearing batches.

A.2 Driving flow matching and optimization

The driving flow path is $a_t = ta + (1-t)\epsilon$, with target vector field $a - \epsilon$ and $\epsilon \sim \mathcal{N}(0, I)$. We draw $b \sim \mathrm{Beta}(1.5, 1.0)$ and set $t = 0.999(1 - b)$. At inference, the expert starts from Gaussian noise and uses 10 Euler steps. Classifier-free guidance is disabled.

Continous action-prediction batches have sampling probability 48.5%, and vision–language batches have probability $\approx 51.5\%$. Training examples with one, two, four, or seven cameras are sampled with relative weights 1:2:3:4; in evaluation, we always use four RGB images. The expert receives a

flow update only on action-bearing batches, whereas the VLM receives an autoregressive update on every joint-training step. Table 5 reports the common optimization settings.

The concurrent conditions start from identical VLM and randomly initialized expert weights. Expert-First conditions share a 100k-step expert checkpoint trained on trajectory data with the VLM frozen, then receive 200k joint-training steps. VLM-First conditions train the VLM alone for 100k steps, either with or without future-action CE, before attaching experts with a shared fresh initialization for 200k joint-training steps. The VLM-First action-token-off warm-up uses the vision–language mixture without action-bearing batches. Comparisons within each history have matched training schedules and update opportunities.

Table 6 gives the expected component-level gradient exposure at the 300k-total-step endpoints and the intermediate VLM-First checkpoint. The Expert-First warm-up uses only trajectory data, so every warm-up step updates the expert. In VLM-First training, action-token supervision also changes the first-stage data mixture: the tokens-off backbone is trained on vision–language data alone, while the tokens-on backbone sees trajectory-prediction data. The four conditions share the same joint-training mixture after the expert is introduced.

Table 6: **Expected component updates.** Endpoints have 300k total steps

History	Schedule	VLM updates	Expert updates
Concurrent	300k joint steps	300k	≈ 145.5k
Expert-First	100k expert-only + 200k joint	200k	≈ 197k
VLM-First	100k VLM-only + 200k joint	300k	≈ 97k

A.3 Manipulation implementation and training

The manipulation policy pairs a $\pi_{0.5}$-style PaliGemma VLM with a flow-matching action expert (Table 7). Observations contain an external RGB image and a left-wrist RGB image, each resized to 224×224, together with seven robot joint values and one gripper value. The state vector is padded to 32 dimensions. The expert predicts a 16-step action chunk in a 32-dimensional padded space; the first eight output dimensions encode seven joint values and one gripper value. During training, joint positions are expressed as deltas, while the gripper value remains absolute.

Table 7: **Manipulation model dimensions.** Attention heads are query/key-value counts.

Component	Backbone	Layers	Width	MLP	Heads (Q/KV)
VLM Backbone	Gemma 2B	18	2,048	16,384	8/1
Action expert	Gemma 300M	18	1,024	4,096	8/1

As in driving, insulation stops the expert's gradient at the backbone K/V interface while preserving the forward representations. In FAST-supervised robot examples, the autoregressive branch predicts a FAST action postfix. Without FAST supervision, the postfix is absent and robot examples contribute no action-token CE; vision–language examples retain their language-answer CE in every condition. The expert is prevented from attending to FAST-token keys. This implementation differs from the driving intervention, where masked future-action value tokens remain in the training sequence. Manipulation CE is normalized by the number of active target positions per example, clipped to at least one, and the flow loss is applied only to robot-action examples.

Each training batch draws a binomially distributed number of vision–language examples with probability one half; the remaining examples come from the robot stream. The streams therefore contribute equally in expectation, although individual batches need not contain exactly half of each type. The robot stream samples 90% DROID data and 10% PolaRiS data. Within the vision–language stream, task weights are 35% VQAv2 question answering, 20% COCO captioning, 20% COCO localization, 15% Visual Genome question answering, and 10% Visual Genome region tasks. A deterministic 2% hash split is held out for validation; robot examples are split by episode, and COCO examples use image identifiers where available to prevent cross-task image overlap.

Table 8: **Manipulation optimization settings.** Shared across conditions through 400k steps.

Setting	Value
Optimizer	AdamW
(β_1, β_2)	$(0.9, 0.95)$
Adam ϵ	10^{-8}
Weight decay	10^{-10}
Peak/terminal LR	5×10^{-5}
Warm-up	1,000 steps
Global batch	128
Gradient clipping	1.0
EMA decay	0.99

Table 8 records the manipulation training settings. Inference uses 10 Euler steps from Gaussian noise.

B Evaluation and Statistical Details

B.1 Open-loop driving

The Public Golden evaluation dataset contains 1,434 examples; the out-of-distribution (OOD) evaluation contains 273 examples. Both evaluations use four cameras, four temporal frames per camera, and a 16-waypoint ego-trajectory history. The policy generates a chain of thought before sampling six continuous future trajectories. Chain-of-thought generation uses temperature 0.6, top-p 0.98, and a maximum of 256 generated tokens, which we found realized the best performance. All driving results in this paper use this inference mode and the same dataset and prompt configuration across training histories.

For each candidate trajectory, ADE is the mean Euclidean distance in the ground plane between predicted and ground-truth waypoints over the 64-step horizon, measured in meters. Minimum ADE selects the lowest-ADE candidate among the six using ground truth. For open-loop bootstrap confidence intervals, we resample base-clip clusters with replacement and retain every example belonging to each sampled clip. We apply the same sampled clips, with the same multiplicities, to all four conditions before recomputing condition means and paired differences. We use 10,000 such resamples and report the 2.5th and 97.5th percentiles of each bootstrap distribution as the 95% interval.

C Additional Manipulation Results

We evaluate the four manipulation conditions at training step 400k in three PolaRiS DROID environments (Jain et al., 2025): BlockStackKitchen, FoodBussing, and PanClean. PolaRiS places reconstructions of real scenes in a physics simulator, allowing policies to be compared from matched initial conditions. Each environment uses one instruction, held fixed across conditions:

- **BlockStackKitchen:** "Place and stack the blocks on top of the green tray"
- **FoodBussing:** "Put all the foods in the bowl"
- **PanClean:** "Use the yellow sponge to scrub the blue handle frying pan"

Evaluation protocol. Each condition is evaluated on the same initial conditions in every environment, which vary per episode, pairing the 50 episodes by scene across conditions. Episodes run for the default 450-step horizon without stopping early on success. The policy receives exterior and wrist RGB images, each resized with padding to 224×224, and predicts seven joint positions and a gripper command. Importantly, we match the training-time inputs in the simulator to prevent distribution shift and give an accurate assessment of policy performance. Actions are executed open-loop in eight-step chunks. We use one training seed per condition.

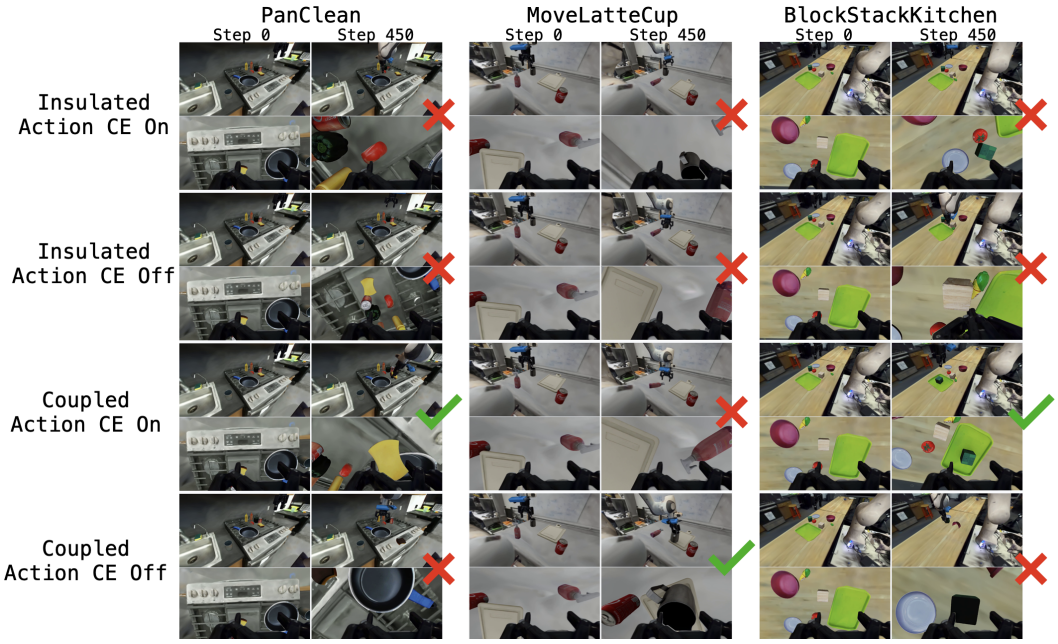

Figure 4: **Closed-loop manipulation examples.** Rollouts of the four concurrently trained policies in `PanClean` (left), `MoveLatteCup` (center), and `BlockStackKitchen` (right). Check marks indicate task success and crosses indicate failure. Among these examples, only the coupled policies succeed.

Scoring. Normalized task progress is the fraction of an environment's rubric criteria satisfied. Each criterion latches once satisfied; criteria with dependencies count only after the required criteria have been satisfied. Progress therefore cannot decrease, and success requires all criteria. Table 9 specifies the conditions and dependencies. For PanClean, the rubric scores placement of the sponge on the pan; it does not separately score a scrubbing motion. Figure 4 showcases qualitative demonstrations of the rollouts in simulation.

Table 9: **PolaRiS rubric criteria.** Overlap is measured between horizontal bounding boxes. Placement requires the gripper to be open.

#	Criterion	Condition	Requires
PanClean			
1	Reach sponge	End effector within 0.2 m of sponge	—
2	Lift sponge	Sponge height > 0.09 m	1
3	Sponge on pan	At least 80% overlap with pan; gripper open	2
FoodBussing			
1	Reach ice cream	End effector within 0.2 m	—
2	Reach grapes	End effector within 0.2 m	—
3	Lift ice cream	More than 0.06 m above rest height	1
4	Lift grapes	More than 0.06 m above rest height	2
5	Ice cream in bowl	At least 80% overlap with bowl; gripper open	3
6	Grapes in bowl	At least 80% overlap with bowl; gripper open	4
BlockStackKitchen			
1	Reach green cube	End effector within 0.2 m	—
2	Reach wood cube	End effector within 0.2 m	—
3	Lift green cube	Cube height > 0.09 m	1
4	Lift wood cube	Cube height > 0.09 m	2
5	Green cube on tray	At least 80% overlap with tray; gripper open	3
6	Wood cube on tray	At least 80% overlap with tray; gripper open	4
7	Green cube on wood cube	At least 50% overlap with wood cube; gripper open	5, 6

D Additional Open-Loop Driving Results

In the main report, we utilized ADE as our metric; here, we report minADE.

D.1 Minimum ADE

Minimum ADE uses the ground truth to choose the best of six sampled trajectories. At the 300k endpoints, its routing pattern is consistent with the selected-trajectory ADE results: removing action-token supervision is more costly under insulation than under coupling across the concurrent, VLM-first, and expert-first histories on both evaluation datasets. Moreover, concurrent training yielded the best performance (see Figure 5).

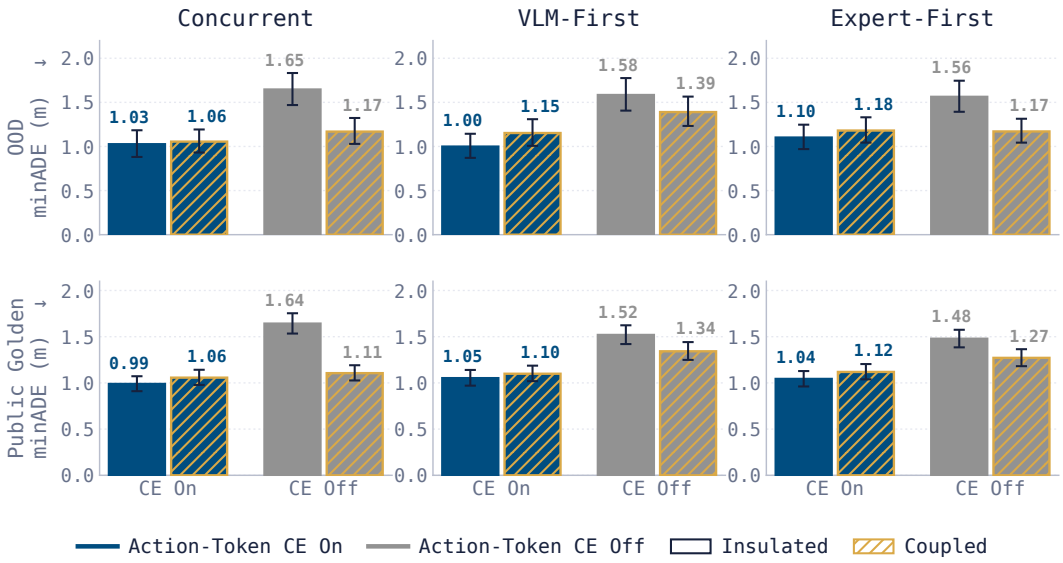

Figure 5: **Open-loop driving across training histories.** Selected-trajectory minADE on the Public Golden dataset and OOD evaluation dataset. Bars show condition means, and error bars show 95% clip-cluster bootstrap confidence intervals for those means. Lower is better.

E Additional Closed-Loop Driving Results

Tables 10, 11, and 12 report the full AlpaSim results. All conditions use the same 827 scenes, with five stochastic rollouts averaged within each scene. AlpaSim terminates or truncates after safety events and censors trajectories at 4 m from the ground-truth trajectory. Arrows indicate the preferred direction, and the best value in each row is bolded. We moreover provide a qualitative demonstration of policy performance in Figure 6.

Table 10: **Concurrent training: AlpaSim metrics.**

	Insulated		Coupled	
Metric	**Tokens on**	**Tokens off**	**Tokens on**	**Tokens off**
Progress ↑	0.600	0.425	**0.703**	0.601
Distance traveled (m) ↑	116.68	78.63	**139.59**	114.78
Duration fraction ↑	0.590	0.363	**0.634**	0.541
Collision, any ↓	0.225	0.311	**0.164**	0.211
Collision, at fault ↓	**0.087**	0.187	0.090	0.127
Off-road ↓	0.161	0.304	**0.136**	0.184
Off-road or at-fault collision ↓	0.248	0.491	**0.226**	0.311
Distance to ground truth (m) ↓	**2.378**	2.471	2.700	2.732
Plan deviation ↓	0.380	0.508	**0.321**	0.378
Wrong lane ↓	0.201	**0.170**	0.230	0.206

Coupling increases route progress without action-token supervision in both staged histories: from 0.538 to 0.604 after expert-first training and from 0.555 to 0.587 after VLM-first training. With action-token supervision, coupling increases progress from 0.597 to 0.625 after expert-first training, while the VLM-first values are nearly identical (0.648 insulated and 0.646 coupled). The highest observed route progress across all three histories is 0.703 under concurrent coupled training with action-token supervision (Table 10).

Table 11: **Expert-first training: AlpaSim metrics.**

	Insulated		Coupled	
Metric	**Tokens on**	**Tokens off**	**Tokens on**	**Tokens off**
Progress ↑	0.597	0.538	**0.625**	0.604
Distance traveled (m) ↑	115.25	104.28	**119.83**	116.13
Duration fraction ↑	**0.599**	0.461	0.561	0.553
Collision, any ↓	0.230	0.305	**0.165**	0.232
Collision, at fault ↓	**0.061**	0.182	0.097	0.139
Off-road ↓	0.159	0.247	**0.134**	0.182
Off-road or at-fault collision ↓	**0.220**	0.429	0.231	0.321
Distance to ground truth (m) ↓	**2.152**	2.569	2.850	2.514
Plan deviation ↓	**0.346**	0.549	0.353	0.413
Wrong lane ↓	**0.190**	0.202	0.288	0.222

Table 12: **VLM-first training: AlpaSim metrics.**

	Insulated		Coupled	
Metric	**Tokens on**	**Tokens off**	**Tokens on**	**Tokens off**
Progress ↑	**0.648**	0.555	0.646	0.587
Distance traveled (m) ↑	128.59	104.96	**129.10**	113.02
Duration fraction ↑	**0.624**	0.508	0.588	0.512
Collision, any ↓	0.208	0.264	0.204	**0.175**
Collision, at fault ↓	**0.076**	0.141	0.117	0.106
Off-road ↓	**0.126**	0.215	0.166	0.199
Off-road or at-fault collision ↓	**0.203**	0.355	0.282	0.305
Distance to ground truth (m) ↓	2.515	**2.512**	2.675	2.829
Plan deviation ↓	**0.360**	0.442	0.412	0.418
Wrong lane ↓	**0.228**	0.244	0.238	0.237

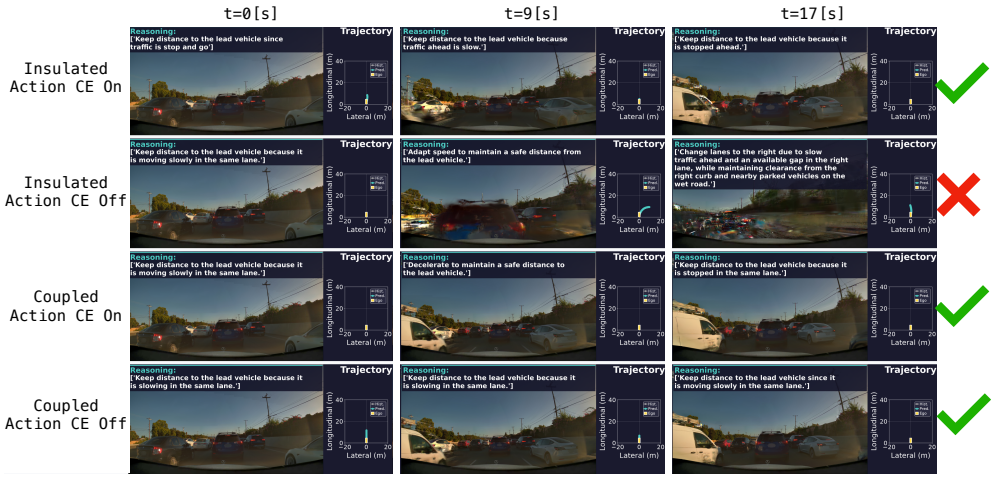

Figure 6: **Concurrent-training AlpaSim case study.** Four conditions in the same traffic scene. The insulated rollout without action-token supervision has an at-fault collision. The displayed rollouts for the other three conditions succeed. This example illustrates a failure mode behind the aggregate results in Table 10.