MAPS: Preserving Vision-Language Representations via Module-Wise Proximity Scheduling for Better Vision-Language-Action Generalization

Chengyue Huang[1]*, Mellon M. Zhang[2]*, Robert Azarcon[1], Glen Chou[2], Zsolt Kira[1]
Robotics Perception and Learning Lab[1], Trustworthy Robotics Lab[2]
Work available at https://arxiv.org/pdf/2511.19878
Accepted to CVPR 2026

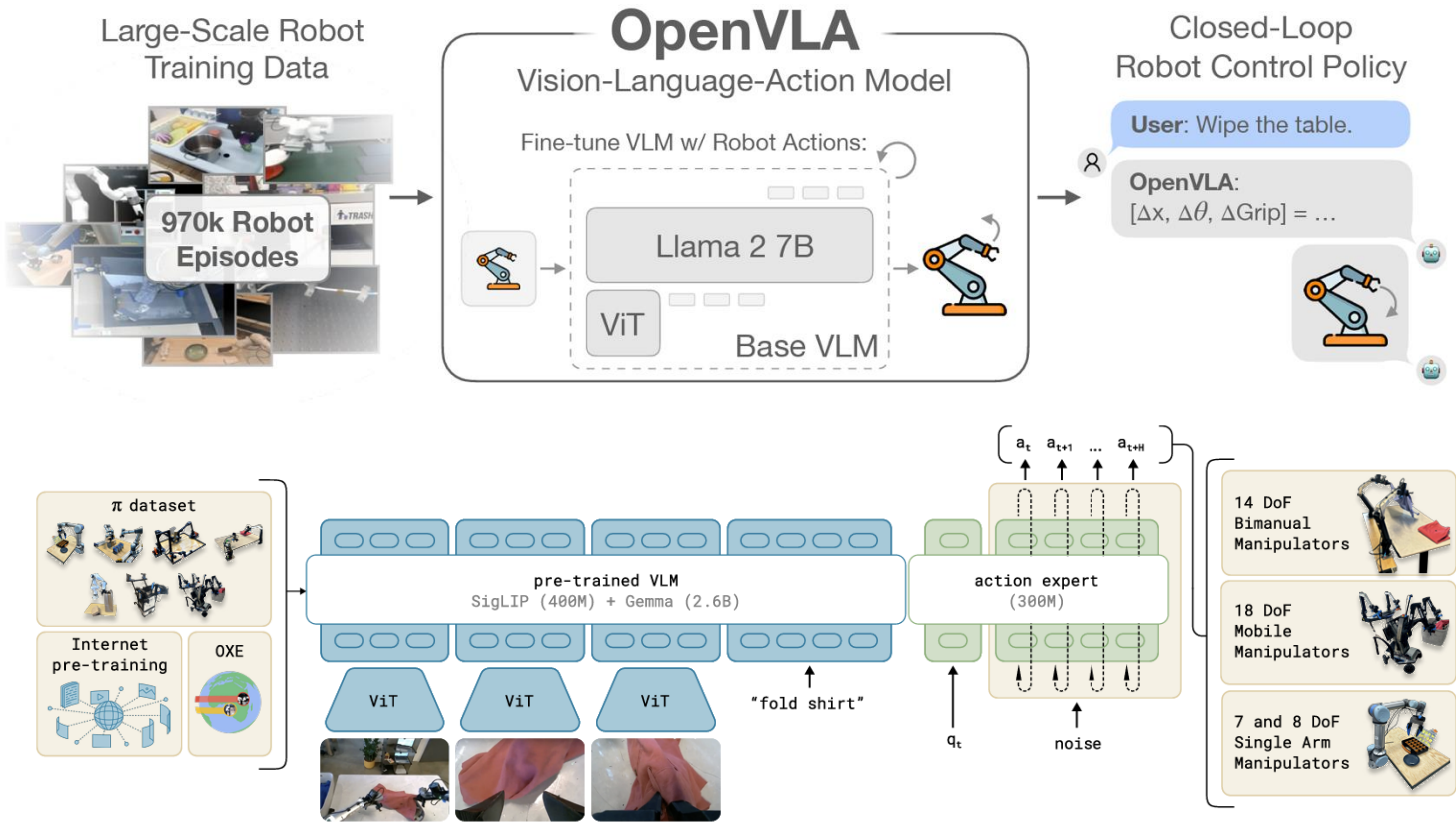
Background: Vision-Language-Action Models

Given input image(s) and language instruction, output:

- Raw actions for N-DoF robot embodiment

Typically consists of:

- Vision encoders (SigLIP, DINOv2, etc.)
- Language backbone (Qwen, Gemma, etc.)
- Action tokenizer (discretized symbol tuning, VQVAE, etc.)



Kim, Moo Jin, et al. "Openvla: An open-source vision-language-action model." *arXiv preprint arXiv:2406.09246* (2024).
Black, Kevin, et al. "π_0: A Vision-Language-Action Flow Model for General Robot Control." arXiv preprint arXiv:2410.24164 (2024).

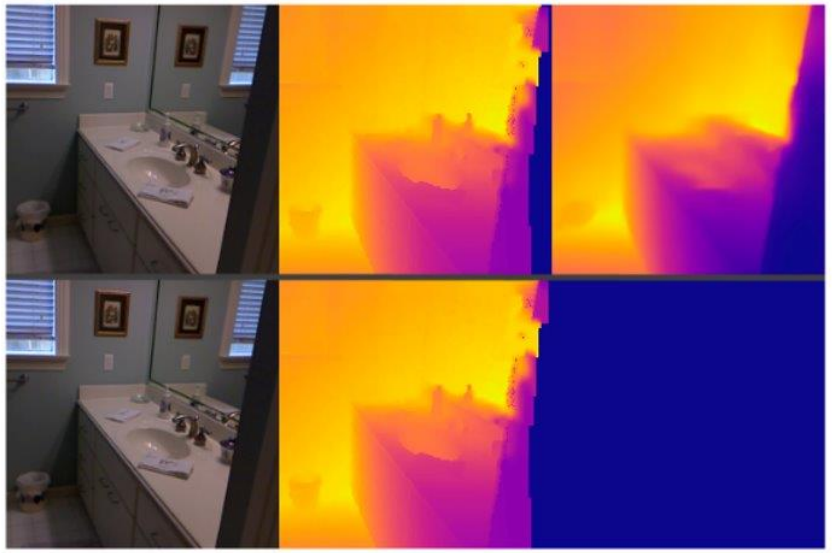
Catastrophic Forgetting in VLAs

VLA models are great when in-distribution (ID) but perform terribly when out-of-distribution (OOD).

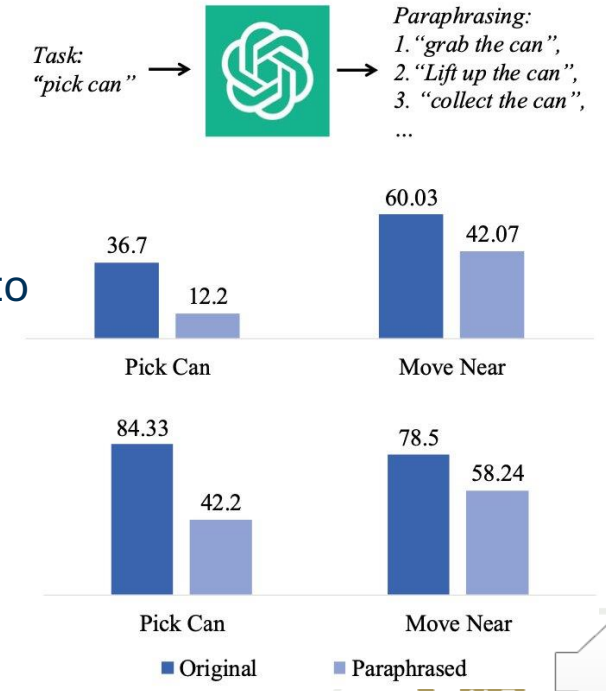
This is a data mismatch problem: small robotics finetuning induces spurious correlations and decreases generalizability of the VLM components. (see right side)

Naïve solution: just add more robotics data.

Is there a principled way to guide the evolution of VLM weights during VLM-to-VLA transfer?



The original DINOv2 estimates depths (top) correctly while OpenVLA's DINOv2 (bottom) performs poorly due to catastrophic forgetting during OpenVLA training.



Dey, Sombit, et al. "Revla: Reverting visual domain limitation of robotic foundation models." 2025 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2025.
Grover, Shresth, et al. "Enhancing generalization in vision-language-action models by preserving pretrained representations." arXiv preprint arXiv:2509.11417 (2025).

The Freezing Paradigm

A common approach to controlled VLA evolution is freezing.

Table 1. Comparison of freezing or robust fine-tuning (RFT) different parts of VLA on SimplerEnv's ID and OOD tasks (details in Sec. 5.3). Early L and Last L denote early and the last language layers. ❄ and 🔥 represent freeze and full fine-tune.

Modules				ID					OOD			
DINOv2	SigLIP	Early L	Last L	T1: Spoon	T2: Carrot	T3: Stack	T4: Eggplant	Avg. ID	Visual	Novel Object	Novel Category	Avg. OOD
MiniVLA-OFT†				22.0	30.0	0.0	2.0	13.5	12.7	2.0	12.0	8.9
❄	❄	🔥	🔥	22.0	24.0	0.0	34.0	20.0	20.0	13.3	10.7	15.0
🔥	🔥	❄	❄	0.0	0.0	0.0	0.0	0.0	1.3	0.0	2.7	1.2
❄	❄	❄	❄	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
❄	🔥	🔥	🔥	42.0	32.0	6.0	42.0	30.5	32.7	51.3	11.3	33.6
🔥	❄	🔥	🔥	12.0	10.0	12.0	90.0	31.0	29.3	35.3	22.7	29.7
🔥	🔥	❄	🔥	0.0	0.0	0.0	0.0	0.0	4.0	0.7	2.0	2.2
RFT				4.0	8.0	0.0	94.0	26.5	6.7	20.0	14.0	13.6

Table 2. Comparison of freezing different parts of VLA on LIBERO. Early L and Last L denote early and the last language layers. ❄ and 🔥 represent freeze and full fine-tune.

Modules				ID	OOD				
DINOv2	SigLIP	Early L	Last L	LIBERO-90	-Spatial	-Object	-Goal	-Long	**Avg. OOD**
VLA-Adapter†				83.0	2.0	5.0	1.0	2.0	2.5
❄	❄	🔥	🔥	92.0	0.0	3.0	0.0	6.0	2.3
🔥	🔥	❄	❄	29.0	1.0	4.0	1.0	0.0	1.5
❄	❄	❄	❄	19.0	0.0	0.0	1.0	0.0	0.3
❄	🔥	🔥	🔥	89.0	0.0	6.0	0.0	6.0	3.0
🔥	❄	🔥	🔥	73.0	1.0	0.0	0.0	3.0	1.0
🔥	🔥	❄	🔥	91.0	0.0	4.0	6.0	8.0	4.5
RFT				88.0	1.0	2.0	0.0	6.0	2.3

Finding 1: Language adaptation is crucial

Finding 2: Freezing vision encoders improves performance

Finding 3: Late language layers drives task performance

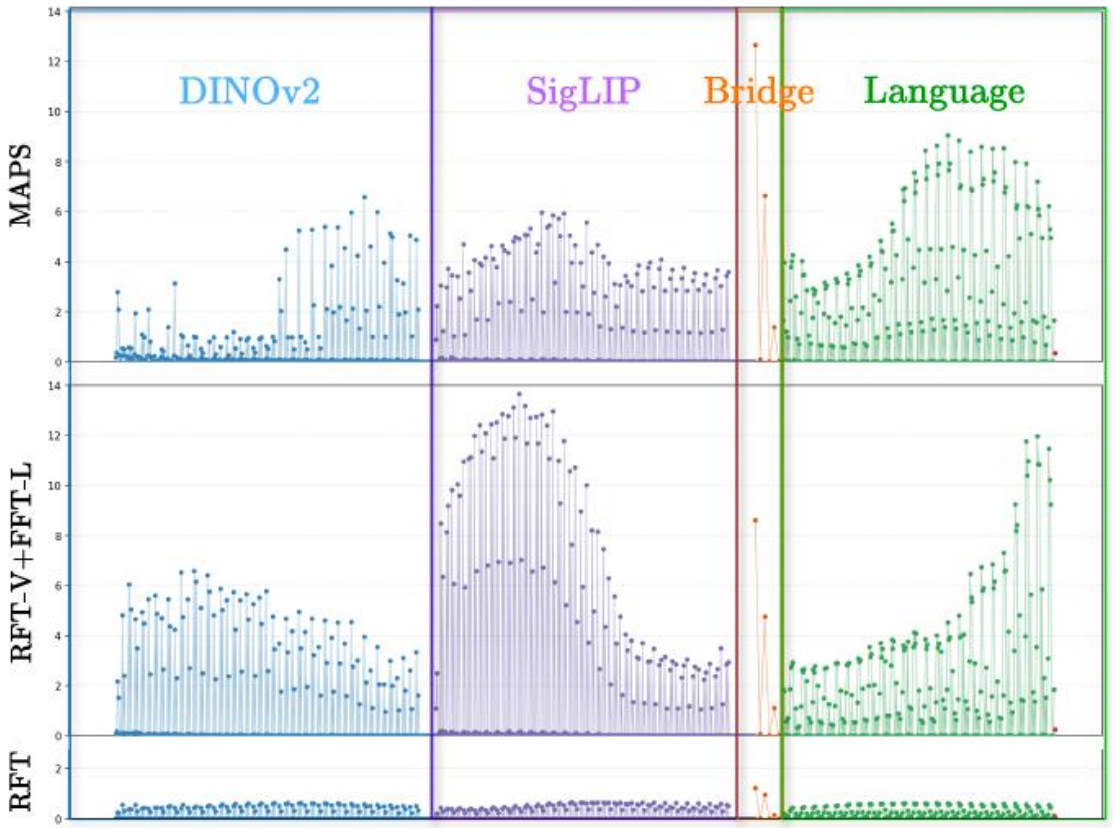
Finding 4: Preserving DINOv2 matters more than SigLIP

Overall priority: DINOv2 > SigLIP > Early language > Late language

Freezing effects are not universally consistent! Hard regularization has strong inductive bias and does not allow flexibility.

From Hard to Soft Regularization: Robust Finetuning (RFT)

- RFT lowers task loss while keeping parameter drift minimal (regularize on $\left|\left|\theta - \theta_0\right|\right|_2^2$).
- Naively applying RFT directly on the action finetuning offers only modest improvements over the baseline.
- Potential reason: RFT techniques impose a single hyperparameter constraint across the entire model, meaning that layers either constrict closer towards or expand away from the pretrained weights at a uniform rate. (right figure)
- **Different components should evolve at different rates.**
- **Model freezing offers high modularity at the cost of heavy inductive bias, RFT offers lighter inductive bias but has no modularity**



Tian, Junjiao, Chengyue Huang, and Zsolt Kira. "Rethinking weight decay for robust fine-tuning of foundation models." Advances in Neural Information Processing Systems 37 (2024): 22418-22440.

MAPS: Module-Wise Proximity Scheduling

- Maps enables strong preservation where pretrained structure is crucial (e.g., early vision and early language layers) while allowing more aggressive adaptation where grounding to actions is most beneficial (e.g., higher-level language layers and the action head).
- Linearly decay proximity hyperparameter λ from $\lambda_{\max}$ to 0 from early to late layers.
- Full fine-tune action head, proprioception projector, etc (modules that are initialized from scratch)

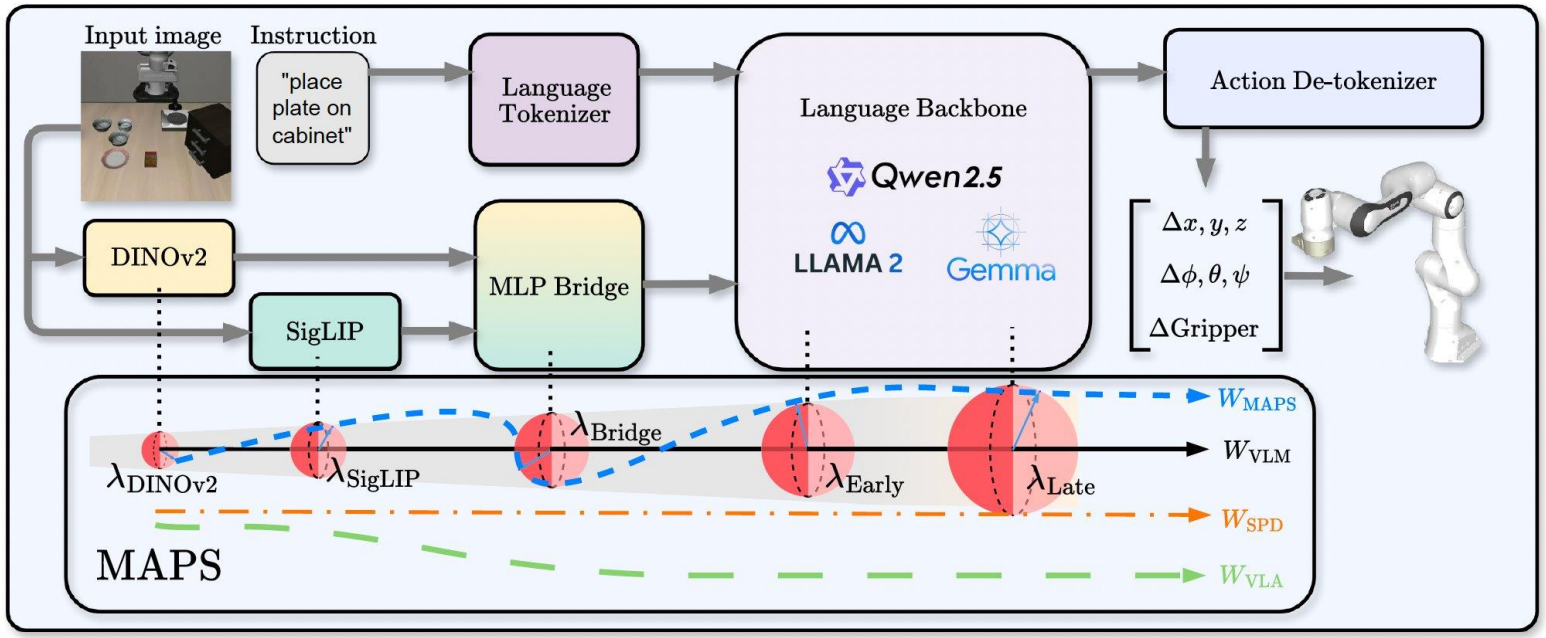


Figure 1. **Module-Wise Proximity Scheduling (MAPS)**. MAPS is applied on pretrained VLM components during the action finetuning stage. MAPS (blue dash line) enforces strong preservation on early vision layers while progressively relaxing constraints toward higher-level language layers. In contrast, vanilla finetuning (green dash line) distorts the VLM representation completely away from its pretrained weights (black solid line), and uniform SPD (orange dash/dot line) applies the same constraint everywhere.

Experiment Setup

- Settings:
  - CALVIN ABC-->D
  - LIBERO
  - SimplerEnv
  - Real Franka Emika Panda 7DoF
- Backbones:
  - MiniVLA-VQ (0.5B)
  - MiniVLA-OFT
  - VLA-Adapter (0.5B)
  - OpenVLA-OFT (7B)

We finetune directly from VLM without VLA pretraining.

- 4 tasks, 150 demos for each task
- In-Distribution Tasks
  - Coke ID: grab the coke can and lift it up,
  - Blocks ID: stack the blue block on the green block,
  - Cups ID: stack the orange cup onto the green cup,
  - Laptop ID: close the laptop lid fully.
- Out-of-Distribution Tasks
  - Coke OOD: coke can is replaced with red bull can;
  - Blocks OOD: blue block is elevated by placing an additional block underneath it;
  - Cups OOD: orange and green cups are replaced with yellow and blue cups, respectively;
  - Laptop OOD: Alienware laptop is replaced with a MacBook.

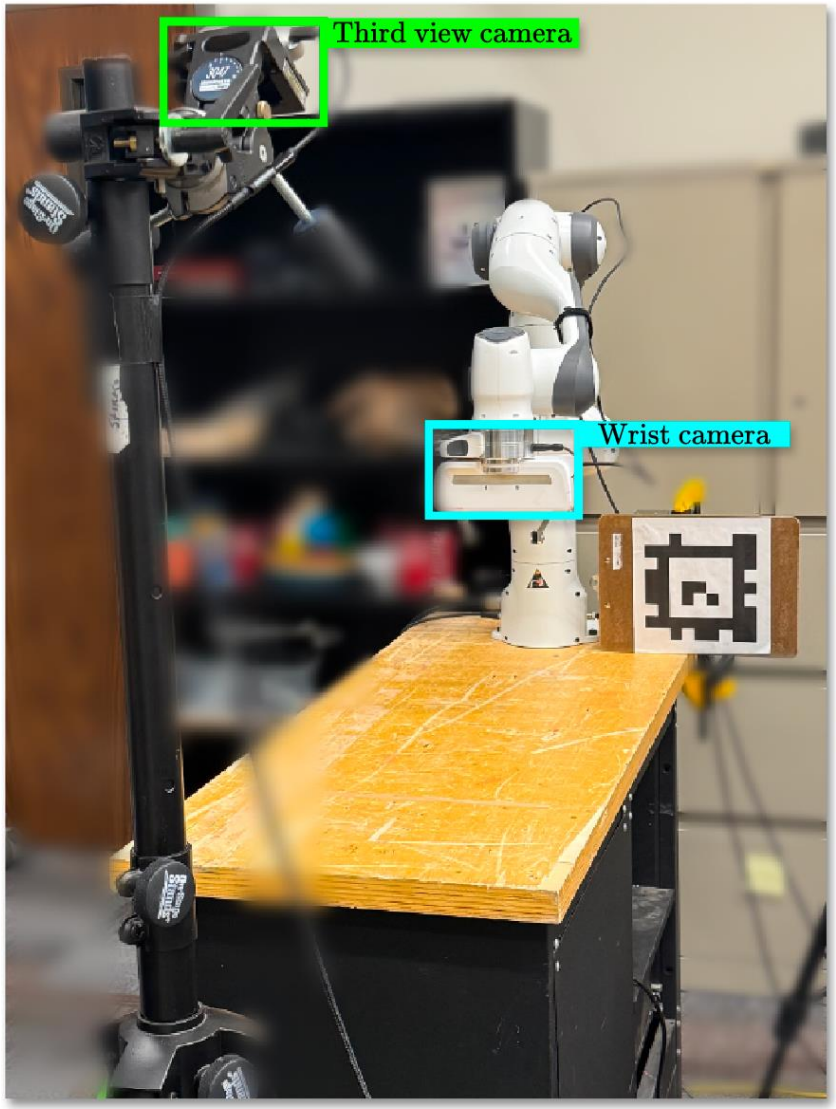


Figure 6. Franka Emika Panda Arm Experiment Setup.

Simulation Experiment 1: LIBERO

- ID: LIBERO-90
- OOD:
  - LIBERO-Spatial,
  - LIBERO-Object
  - LIBERO-Goal
  - LIBERO-Long

Table 3. **LIBERO Performance.** * models borrowed from [17]. † models pretrained from scratch on LIBERO-90. ‡ models fine-tuned on LIBERO-90.

Model	ID (%)	OOD (%)				
	LIBERO-90	-Spatial	-Object	-Goal	-Long	**Avg. OOD**
OpenVLA	61.4	-	-	-	-	-
miniVLA*	62.0	0.0	0.0	0.0	1.0	0.25
miniVLA-VQ*	77.0	0.0	0.0	3.0	1.0	1.0
miniVLA-hist.*	82.0	0.0	1.0	2.0	7.0	2.5
miniVLA-wrist*	82.1	0.0	0.0	0.0	0.0	0.0
SpatialVLA‡	46.2	0.0	0.0	0.67	1.33	0.5
MiniVLA-VQ†	79.0	0.0	0.0	0.0	0.0	0.0
+MAPS (*Ours*)	82.0	0.0	1.0	8.0	10.0	4.75
Δ	+3.0	+0.0	+1.0	+8.0	+10.0	+4.75
MiniVLA-OFT†	91.0	0.0	8.0	7.0	4.0	4.75
+MAPS (*Ours*)	91.0	6.0	14.0	0.0	8.0	7.0
Δ	+0.0	+6.0	+6.0	-7.0	+4.0	+2.25
VLA-Adapter†	83.0	2.0	5.0	1.0	2.0	2.5
+MAPS (*Ours*)	88.0	0.0	7.0	6.0	6.0	4.75
Δ	+5.0	-2.0	+2.0	+5.0	+4.0	+2.25
OpenVLA-OFT†	92.0	3.0	9.0	3.0	5.0	5.0
+MAPS (*Ours*)	90.0	0.0	11.0	1.0	10.0	5.5
Δ	-2.0	-3.0	+2.0	-2.0	+5.0	+0.5

Simulation Experiment 2: CALVIN

- Models are trained on splits A–C and evaluated on split D.
- Metrics:
  - Tasks completed in a row: success rate for completing exactly k consecutive tasks (k = 1–5)
  - Average Length: mean number of tasks completed per five-instruction sequence across all episodes

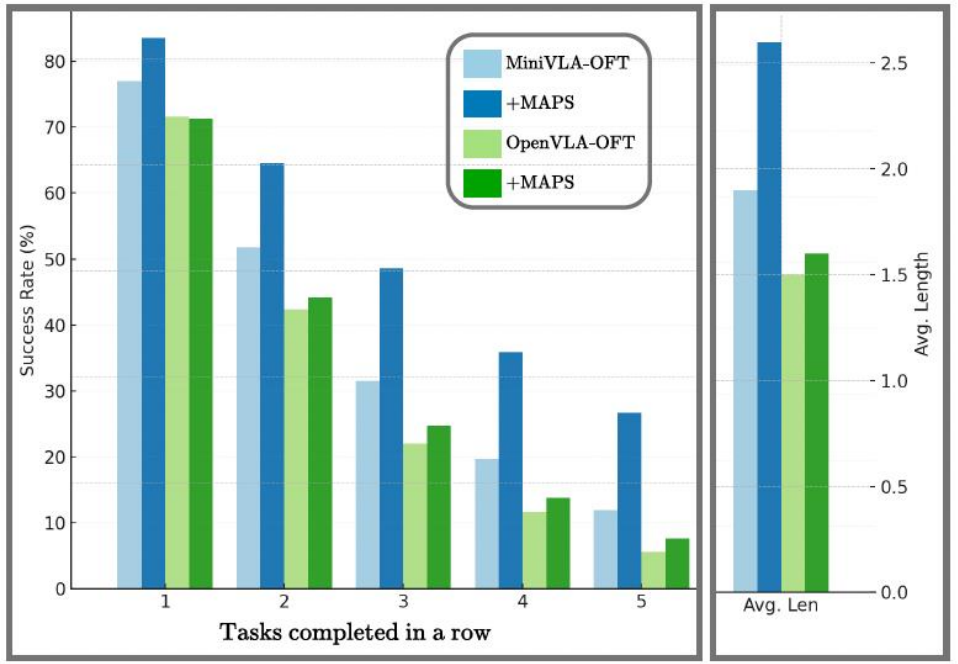


Figure 4. MAPS improves success rates across all action horizons (25% for MiniVLA-OFT and 3% for OpenVLA-OFT) and average length (+0.7 for MiniVLA-OFT and +0.1 for OpenVLA-OFT).

Simulation Experiment 3: SimplerEnv

Table 4. **SimplerEnv Results**. ID includes 4 tasks following SimplerEnv-Bridge setup. OOD evaluation suites include Visual, Novel Object, and Novel Category. * model checkpoints borrowed from original papers. † models pretrained from scratch on SimplerEnv.

Model	ID					OOD			
	T1: Spoon	T2: Carrot	T3: Stack	T4: Eggplant	Avg. ID	Visual	Novel Object	Novel Category	Avg. OOD
RT-1-X [2]	4.2	0.0	0.0	0.0	1.1	0.0	4.0	6.1	3.4
Octo [52, 53]	12.5	41.7	15.8	0.0	17.5	12.6	10.8	8.4	10.6
π_0 [4]	52.5	87.9	83.8	52.5	69.2	71.4	30.2	21.0	40.9
MiniVLA-VQ † [17]	52.0	24.0	42.0	58.0	44.0	37.3	34.0	27.3	32.9
+MAPS (*Ours*)	44.0	30.0	76.0	20.0	42.5	48.0	26.0	29.3	34.4
Δ	-8.0	+6.0	+34.0	-38.0	-1.5	+10.7	-8.0	+2.0	+1.5
MiniVLA-OFT†	22.0	30.0	0.0	2.0	13.5	12.7	2.0	12.0	8.9
+MAPS (*Ours*)	34.0	38.0	0.0	48.0	30.0	32.0	48.7	26.7	35.8
Δ	+12.0	+8.0	+0.0	+46.0	+16.5	+19.3	+46.7	+14.7	+26.9
OpenVLA-OFT† [19]	16.0	18.0	0.0	58.0	23.0	16.0	9.3	0.7	8.7
+MAPS (*Ours*)	2.0	16.0	0.0	74.0	23.0	18.7	20.0	12.7	17.1
Δ	-14.0	-2.0	+0.0	+16.0	+0.0	+2.7	+10.7	+12.0	+8.4

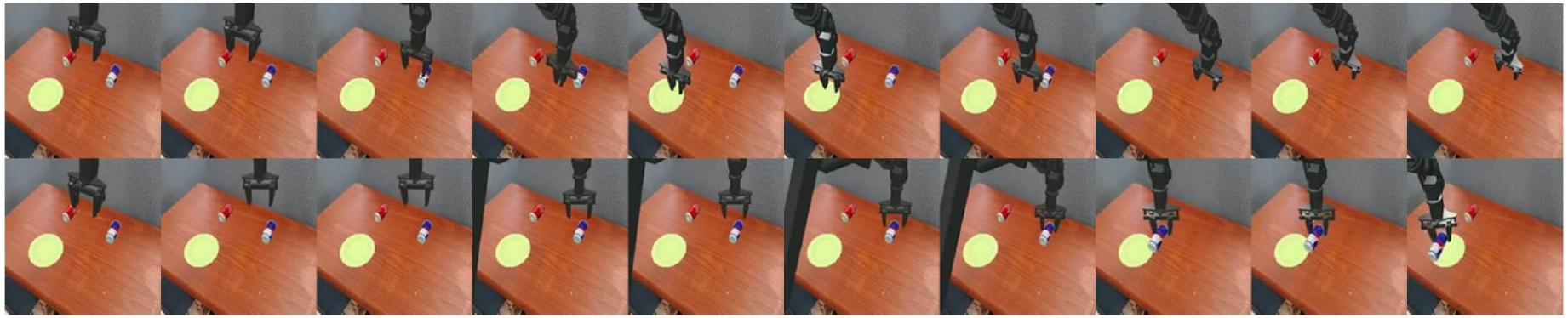


Figure 5. SimplerEnv qualitative example (new object: Red Bull can). Top: Full fine-tuning fails to complete the task. Bottom: MAPS successfully places the can on the plate.

Real Experiment: Franka Arm

- Avg ID performance +32.5%, Avg OOD performance +30.0%.
- Intermediate task success rates improve across ID/OOD and Easy/Hard settings
- Avg number of tries per success significantly lower across most tasks

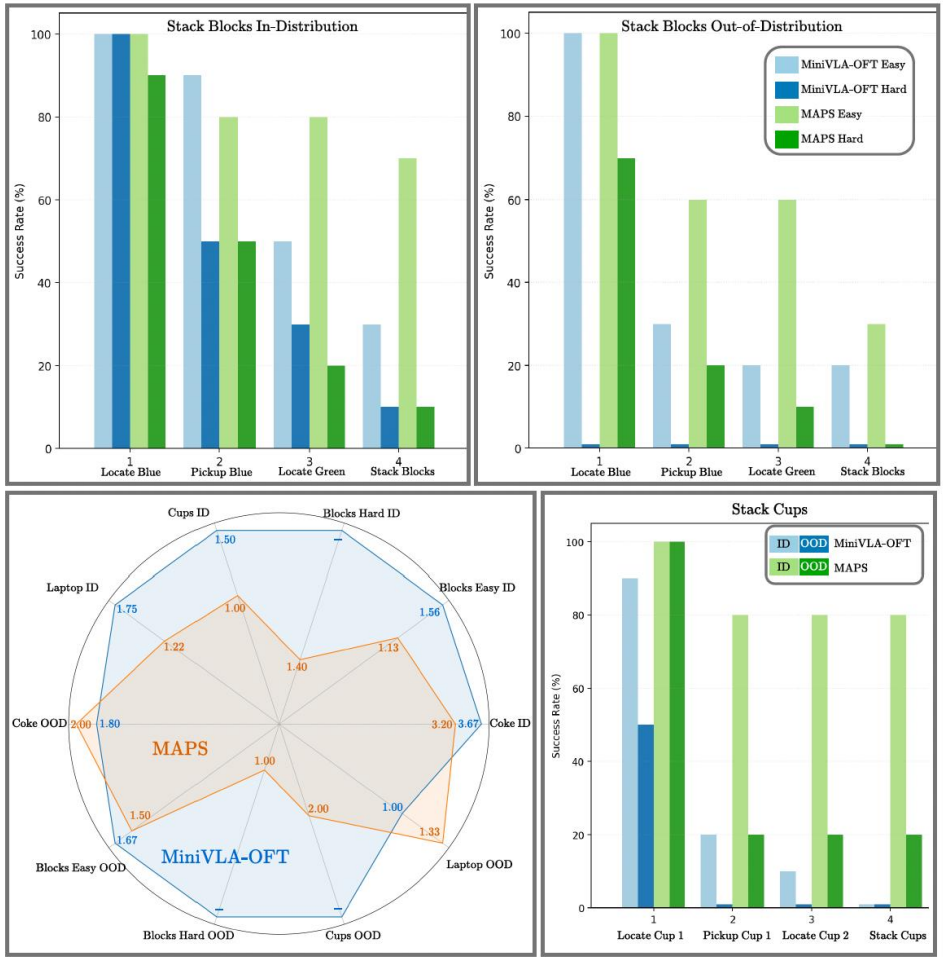


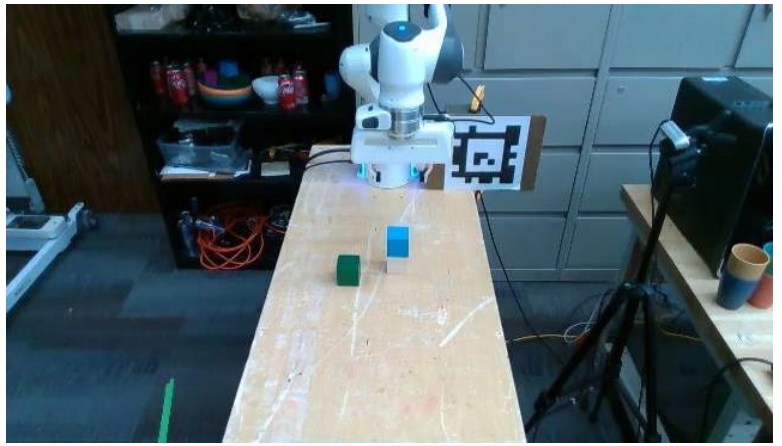
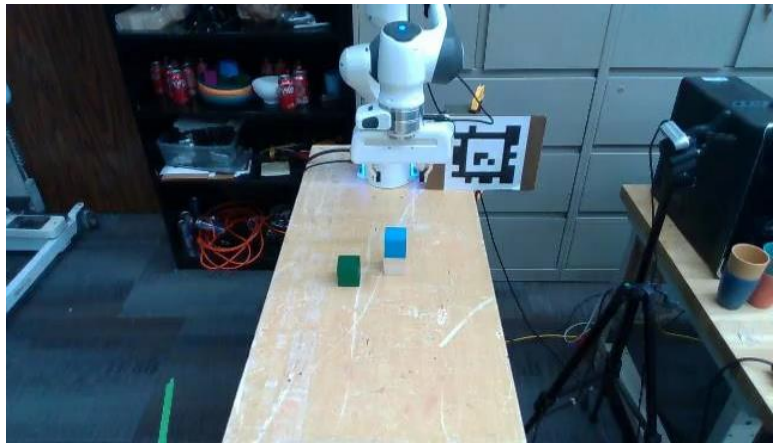
Figure 7. **Franka Detailed Results. Top Left:** Success rates for the Blocks ID Easy/Hard tasks. **Top Right:** Success rates for the Blocks OOD Easy/Hard tasks. **Bottom Left:** Average number of tries per success across all Franka tasks. **Bottom Right:** Success rates for the Cups ID/OOD tasks.

Table 6. **Franka Results.**

Model	ID					OOD				
	T1: Coke	T2: Block	T3: Cup	T4: Laptop	Avg. ID	T1: Coke	T2: Block	T3: Cup	T4: Laptop	Avg. OOD
MiniVLA-OFT†	30.0	30.0	0.0	100.0	40.0	50.0	20.0	0.0	30.0	22.5
+MAPS (*Ours*)	50.0	70.0	80.0	90.0	72.5	100.0	30.0	20.0	60.0	52.5
Δ	+20.0	+40.0	+80.0	-10.0	+32.5	+50.0	+10.0	+20.0	+30.0	+30.0

Qualitative Examples

MAPS enables graceful recovery from initial failure, more careful execution and generalizability to unseen scenarios. All by preserving the abilities of the base VLM.

	Baseline	MAPS
Cups ID		
Blocks OOD		

Thank You!

Website Link

Paper Link