

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

Supplementary Material

7. Implementation Details

Backbone details. MiniVLA [17] pairs a DINOv2-SigLIP visual stack with a Qwen2.5-0.5B [6] language model; the VQ variant replaces the typical uniform discretized action bins with a vector-quantized VAE [18], enabling multi-step action chunking from a single forward pass. OpenVLA-OFT [19] instead uses a LLaMA-2 7B [7] backbone and a parallel decoding head that directly regresses continuous actions via an ℓ_1 loss, fully avoiding discretization. By implementing this OFT-style action head on the MiniVLA backbone, we also create a lightweight MiniVLA-OFT variant, allowing us to test our method with both small and large LMs while holding other components fixed. Finally, VLA-Adapter [53] introduces a lightweight action head with Bridge Attention, enabling efficient and low-cost VLA training.

Algorithm. The full algorithm of MAPS is shown in Algorithm 1. The algorithm extends standard Adam optimization with a progressive Module-Wise Proximity Scheduling regularization strategy applied across an ordered stack of model layers. Beginning with the lowest-level representation (DINOv2) and proceeding sequentially toward the highest-level language layers, the method assigns each layer a monotonically decreasing proximal weight λ_k , with the strongest regularization applied to early layers and no regularization applied to the final layer. For each layer, the algorithm performs standard Adam updates by computing stochastic gradients, maintaining exponential moving averages of first- and second-order moments, and applying bias correction before taking an adaptive step in parameter space. After each tentative Adam update $\tilde{\theta}_t$, the method evaluates a MAPS consistency signal c_t , defined as the negative inner product between the current negative gradient $-g_t^T$ and the displacement from the initialization $\theta_{t-1} - \theta_0$. A negative value indicates movement that contradicts the expected descent direction relative to the original initialization and triggers a correction: the updated parameters are pulled back toward the initialization by an amount proportional to both the layer-specific proximal weight λ_k and the deviation ratio r_t . If the consistency signal is non-negative, the Adam update is accepted unchanged. By iterating this procedure until convergence at each layer before progressing to the next, the algorithm preserves the stability of foundational layers and components (DINOv2 and SigLIP), encourages smooth and initialization-aligned updates in intermediate modules, and allows full flexibility in the final language components.

Algorithm 1: Adam with MAPS.

Given: ordered layer set

$\mathcal{L} = (\text{DINOv2}, \text{SigLIP}, \text{Bridge}, \text{Language})$

Hyperparameters: $\alpha, \beta_1, \beta_2, \epsilon, \lambda_{\max}$

Pre-computation: For $k = 1, \dots, |\mathcal{L}|$ define

$$\lambda_k := \lambda_{\max} \left(1 - \frac{k-1}{|\mathcal{L}|-1} \right),$$

so that $\lambda_1 = \lambda_{\max}$ (DINOv2) and $\lambda_{|\mathcal{L}|} = 0$ (Language).

Initialization: $m_0 \leftarrow 0, v_0 \leftarrow 0, t \leftarrow 0, c_0 \leftarrow 0$

For $k = 1, \dots, |\mathcal{L}|$ (**from DINOv2 to Language**)
do

while θ_t not converged **do**

$t \leftarrow t + 1$

$g_t \leftarrow \nabla_{\theta} \tilde{\mathcal{L}}(\theta_{t-1})$

$m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1) g_t$

$v_t \leftarrow \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$

Bias correction:

$$\hat{m}_t \leftarrow \frac{m_t}{1 - \beta_1^t}, \quad \hat{v}_t \leftarrow \frac{v_t}{1 - \beta_2^t}$$

Adam update:

$$\tilde{\theta}_t \leftarrow \theta_{t-1} - \frac{\alpha \hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

$$c_t \leftarrow -g_t^T (\theta_{t-1} - \theta_0)$$

If $c_t < 0$ **then**

$$\theta_t \leftarrow \tilde{\theta}_t - \lambda_k r_t (\tilde{\theta}_t - \theta_0)$$

else

$$\theta_t \leftarrow \tilde{\theta}_t$$

end if

end while

end for

Model and benchmark configurations. In Tab. 6, we summarize the training configurations used across different models and benchmarks. Specifically, we report the optimal learning rate, total training steps, global batch size, usage of the wrist camera and proprioceptive state, whether image augmentation is applied, whether LoRA is used (and, if so, the chosen rank), the GPU setup, total training time, and the value of λ_{\max} used in MAPS.

Table 6. Configurations for Each Model and Benchmark.

Benchmark	Model	lr	steps	batch size	use wrist	use proprio	use img aug	lora	GPUs	Training Time	λ_{\max}
LIBERO	MiniVLA-VQ	1e-5	50000	16*8	no	no	False	False	8*A40	24h	0.5
	MiniVLA-OFT	5e-5	50000	8*8	yes	yes	True	False	8*A40	19h	3.2
	VLA-Adapter	1e-4	50000	8*8	yes	yes	True	False	8*A40	22h	0.5
	OpenVLA-OFT	5e-5	50000	4*16	yes	yes	True	r=32	16*A40	28h	1.0
CALVIN	MiniVLA-OFT	5e-5	40000	8*8	yes	yes	True	False	8*A40	19h	2.5
	OpenVLA-OFT	1e-4	35000	4*16	yes	yes	True	r=32	16*A40	28h	1.5
SimplerEnv	MiniVLA-VQ	1e-5	50000	32*4	no	no	False	False	4*H200	10h	0.5
	MiniVLA-OFT	5e-5	100000	8*8	no	no	True	False	8*A40	15h	3.0
	OpenVLA-OFT	2e-4	100000	16*4	no	no	True	r=32	4*H200	21h	0.8
Franka	MiniVLA-OFT	5e-5	100000	8*8	yes	no	True	False	8*A40	14h	1.5

8. Performance on Real-World Franka Emika Panda Robot

8.1. Experiment Setting

Task suite. We conduct real-world tabletop manipulation experiments using a single Franka Emika Panda robotic arm. We form a pretraining corpora of 600 demonstrations across 4 different ID training tasks following [55], which are:

1. Coke ID: grab the coke can and lift it up,
2. Blocks ID: stack the blue block on the green block,
3. Cups ID: stack the orange cup onto the green cup,
4. Laptop ID: close the laptop lid fully.

To assess generalization, we evaluate the trained policies on the 4 ID tasks as well as on 4 corresponding OOD variants. The OOD tasks introduce controlled distribution shifts while preserving the underlying manipulation objectives. Specifically:

1. Coke OOD: coke can is replaced with red bull can;
2. Blocks OOD: blue block is elevated by placing an additional block underneath it;
3. Cups OOD: orange and green cups are replaced with yellow and blue cups, respectively;
4. Laptop OOD: Alienware laptop is replaced with a MacBook.

Overall, we design the OOD tasks to systematically probe performance under diverse distribution shifts. Coke OOD introduces novel instruction, class (Red Bull can), object geometry, and color. Blocks OOD tests novel elevation (elevated blue block). Cups OOD tests novel instruction and color. Laptop OOD tests novel class, color, object geometry, and altered physics (joint stiffness).

Hardware Setup. We use a single Franka Emika Panda robotic arm and two RealSense D435 cameras, one mounted in a third perspective and one mounted as a wrist camera (Fig. 6). The cameras capture images in 1920×1080 resolution, which is resized and center-cropped to construct 224×224 resolution inputs for training and inference.

Experiment Setup. We evaluate each task setting using

ten rollouts across two object–location configurations, with five rollouts per configuration. All evaluations are executed in the real-world setup using the same Franka system as in training with a max limit of 500 timesteps. For the multi-object Blocks and Cups tasks, the second configuration is created by swapping the positions of the objects. For the Coke task, we use “near” and “far” layouts, placing the can either within the gripper’s immediate reach or farther away. For the Laptop task, the two configurations place the laptop at an acute 60° angle, oriented either toward or away from the camera. To enable a more fine-grained task analysis, we further introduce a Hard setting for the Blocks task, where objects are positioned farther from the robot arm (Fig. 7) while still relying on configurations that simply swap object positions. We also define a new out-of-distribution (OOD) variant for this Hard setting by replacing the blue block with a red one. This results in a substantially more challenging OOD condition than the original Easy OOD, as both the visual appearance and semantic attributes differ from the pretrained setup.

8.2. Fine-grained task analysis.

Beyond reporting aggregate success rates, we also provide a more fine-grained analysis of MAPS in Fig. 8, differentiating non-trivial competency on intermediate steps. To this end, we decompose the multi-step Task 2: Block (Fig. 8 Top) and Task 3: Cup (Fig. 8 Bottom Right) into a sequence of subtasks: (1) locating object 1, (2) picking up object 1, (3) locating object 2, and (4) stacking object 1 on object 2. We record the success rates on each of these subtasks for a closer look at potential failure modes. Finally, we also track how many attempts each method needs to complete a rollout (Fig. 8 Bottom Left). For Task 1 and Task 4, which have no intermediate subtasks, this value is the number of tries until the task succeeds. For Tasks 2 and 3, which include multiple subtasks, we record attempts only for the grasping subtask, as it is the only stage where the policy can retry.

MAPS’s fine-grained evaluations show that its biggest advantages come from more reliable grasps, more ac-

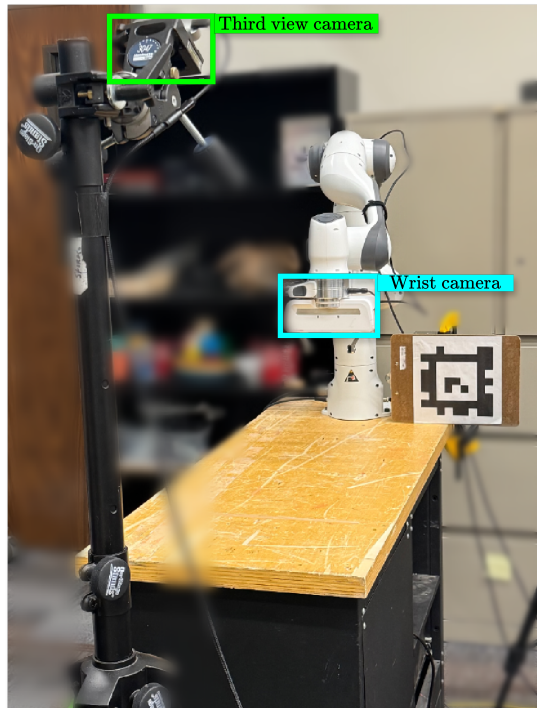


Figure 6. Franka Emika Panda Arm Experiment Setup.

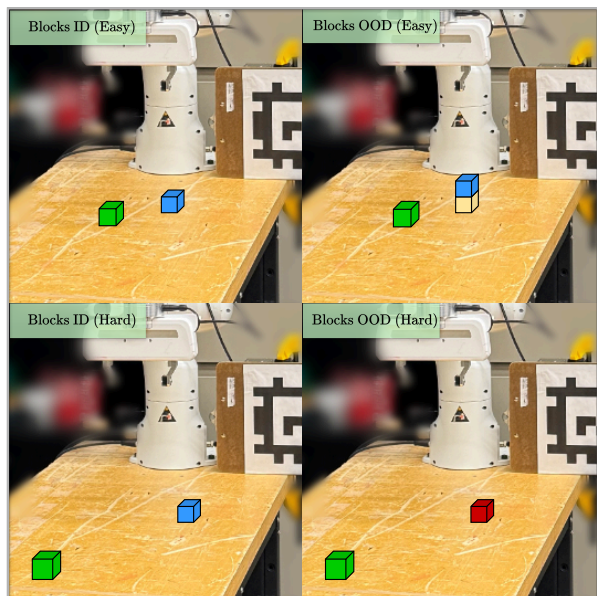


Figure 7. **Task 2 (Block) settings.** To further differentiate between MAPS and the baseline MiniVLA-OFT, we construct a more challenging “Hard” variant of T2: Block by introducing more geometric and vision variations. Blocks ID Hard (bottom left) increases the separation between the two blocks to probe long range planning and task execution, whereas Blocks OOD Hard (bottom right) additionally introduces an unseen new color block to the task.

curate intermediate localization, and more stable multi-step execution across all object categories. In Blocks ID

(Easy), MAPS strengthens later-stage performance, improving second-block localization (80% vs. 50%), more than doubling final placements (70% vs. 30%), and reducing grasp attempts (1.13 vs. 1.56). Similar gains appear in Blocks OOD (Easy), boosting first-block grasping (60% vs. 30%), second-block localization (60% vs. 20%), and final placement (30% vs. 20%), indicating robustness to modest distribution shift. Under Hard settings, both models degrade, but MAPS still reduces repeated grasps on Blocks ID (1.4 vs. 4.2). In Blocks OOD (Hard), where object 1 is fully OOD, the baseline fails entirely while MAPS maintains non-trivial competence – 70% first-block localization, 20% initial grasps, and 10% second-block localization – showing resilience to combined geometric and appearance shifts. For cup stacking, MAPS substantially improves every sub-stage: grasp-first success (80% vs. 20%), second-object localization (80% vs. 10%), final stacking (80% vs. 0%), and grasp attempts (1.0 vs. 1.5) in Cups ID; and in Cups OOD, the baseline fails beyond initial localization while MAPS successfully grasps, relocates, and stacks (2 trials each), demonstrating clear generalization. For laptop closing, both models locate and push the lid, but MAPS excels on the harder metric of fully closing: in Laptop ID it remains near-perfect (90% vs. 100%) with fewer attempts (1.22 vs. 1.75), and in Laptop OOD it doubles the fully closed rate (60% vs. 30%), showing better control accuracy under appearance change. For coke can lifting, MAPS improves success in both ID (50% vs. 30%) and OOD (100% vs. 50%), with slightly fewer attempts in ID (3.20 vs. 3.67); its perfect OOD score reflects strong appearance invariance. Overall, MAPS not only boosts final task success but consistently improves intermediate sub-task reliability across both ID and OOD conditions.

8.3. Qualitative Results

We show the qualitative results of ID and OOD tasks in Figs. 10 and 11, with raw video recordings of these roll-outs included on our project site. Across both settings, MAPS and the baseline exhibit clear and consistent behavioral differences. The baseline policies frequently stall, hesitate, or follow incorrect trajectories, resulting in missed grasps, poor approach angles, unstable placements, or premature object drops. These failures are visible in Cup ID, Block ID, and Coke ID (Fig. 10), where the baseline often grasps imprecisely, retries multiple times (Block ID), tips objects over (Coke ID), or performs poor localization that causes early releases (Cup ID). In contrast, MAPS executes smoother, more deliberate, and more stable motion plans. It firmly grasps objects, recalibrates effectively to locate secondary targets, and demonstrates careful multi-step execution – such as reopening and reclosing the gripper when the orange cup initially fails to stack into the green cup – without letting objects fall.

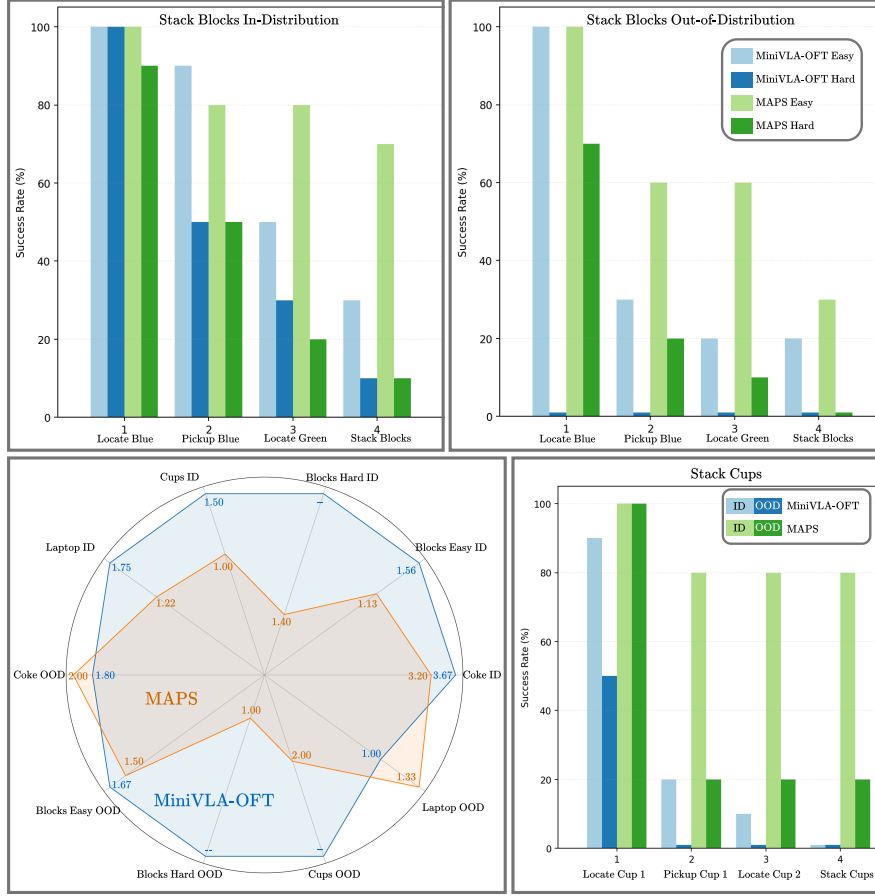


Figure 8. **Franka Detailed Results.** **Top Left:** Success rates for the Blocks ID Easy/Hard tasks. **Top Right:** Success rates for the Blocks OOD Easy/Hard tasks. **Bottom Left:** Average number of tries per success across all Franka tasks. **Bottom Right:** Success rates for the Cups ID/OOD tasks.

These qualitative differences become even more pronounced under distribution shifts (Fig. 11). MAPS consistently adapts to altered object appearances and dynamics, whereas the baseline tends to rigidly reuse its ID trajectories and fails to compensate for changes. In Laptop OOD, replacing the Alienware laptop with a MacBook Pro introduces greater joint friction; the baseline follows its original trajectory and leaves the MacBook partially open (10–30%), while MAPS adopts a more conservative path and applies sufficient, consistent force to close it fully. A similar pattern appears in Block OOD, where the first block’s elevation is altered. MAPS successfully adjusts its grasp strategy, while the baseline struggles and persists with its ID motion. Taken together, these visual rollouts show that MAPS not only corrects common baseline failure modes but also exhibits markedly stronger robustness, carefulness, and adaptability in both familiar and shifted environments.

8.4. Results Discussion

We discuss some questions regarding our qualitative and quantitative results.

(1) Why are the performance gains on our real world ID evaluations so much more pronounced compared to our simulation ID results in the main paper? One of our central hypotheses for the large ID performance gap on Franka is the limited size of our pretraining dataset – only 600 demonstrations – which likely leads to substantial overfitting in the baseline model. Both our qualitative observations and quantitative OOD results already indicate that the baseline MiniVLA-OFT is highly sensitive to spurious correlations. We believe the baseline has overfit so strongly to the demonstration distribution that minor natural variations which should remain well within the ID regime become challenging for it to handle. In contrast, the regularization introduced by MAPS mitigates this overfitting, enabling the model to generalize more robustly.

(2) Why, in general, are these success rates the way that

they are? Across our tasks, a consistent trend emerges: grasping is challenging, and grasping round objects is especially difficult. In contrast, the Laptop task is comparatively trivial because it only requires the robot arm to push the object along a trajectory rather than perform a true grasp. The Block task is easier than Cup because the block’s square geometry provides stable grasping affordances. This explains why the baseline and MAPS perform similarly on Block grasping, whereas MAPS is substantially better on Cup grasping. As shown in Fig. 8, the baseline’s primary failure point on Blocks (ID) is in locating the second object, while on Cups (ID) the main dropoff occurs at the very first grasp. The Coke task also involves grasping a cylindrical object, but it is more straightforward than Cup because only a single grasp is required. Taken together, these observations suggest the following expected difficulty ordering: Cup > Block > Coke > Laptop. This is also consistent with our observed results.

(3) Why does MAPS perform worse on Laptop ID compared to baseline MiniVLA-OFT? On the Laptop ID/OOD tasks, we observed a recurring failure mode shared by both MAPS and the baseline, stemming partly from the task’s design. In configurations where the laptop screen faces the robot, once the arm pushes the lid down past a certain angle, the third-view camera can no longer reliably determine whether the laptop is fully closed. Because the learned policies typically lift the arm after pushing to avoid contacting the table, small partial closings also become indistinguishable from the wrist-view camera. The single MAPS ID failure arose from this ambiguity, and we observed similar issues in the OOD setting for both methods. Although one could debate how to interpret “success” for such borderline cases, we chose to evaluate results strictly and consistently. Overall, the Laptop task appears too saturated and simple to reveal meaningful differences in the ID setting – making a 10% change negligible – whereas the 30% gap in the OOD setting is far more consequential.

9. Ablation Studies

We design our ablation studies to rigorously evaluate our design choices and compare against prior approaches. Specifically, we compare different projection strengths and schedulers, as well as compare against other methods for preserving pretrained representations.

Comparison with different schedulers. We ablate the choice of scheduler and projection strength values in Table 7 on the LIBERO benchmark. Specifically, we compare against regular robust-finetuning (Constant scheduler), cosine scheduler and linear scheduler on projection strength values of 0.5, 2, and 3. We find that in general, project-strength modulation offers ID and OOD benefits compared to regular robust-finetuning. We also find that the linear scheduler offers most stable ID performance and greatest

gains in OOD, with the best configuration being a linear scheduler with projection strength $v = 0.5$.

Table 7. **Scheduler Comparison:** Linear (MAPS, Ours) vs. Cosine (and Constant) across projection-strength values.

Scheduler / Value	ID		OOD			
	LIBERO-90	-Spatial	-Object	-Goal	-Long	Avg. OOD
Constant ($v=0.5$)	80	0	0	0	6	1.5
Constant ($v=2$)	67	0	0	0	4	1
Constant ($v=3$)	51	0	0	0	0	0
Cosine ($v=0.5$)	81	0	2	0	6	2
Cosine ($v=2$)	85	0	7	3	3	3.25
Cosine ($v=3$)	87	0	9	1	2	3
Linear ($v=0.5$)	88	0	7	6	6	4.75
Linear ($v=2$)	84	2	6	0	4	3
Linear ($v=3$)	88	1	2	0	6	2.25

Table 8. **Dual Encoders vs. MAPS** on LIBERO using MiniVLA-VQ.

Method	LIBERO-90 (ID)	Avg. OOD	Params
MiniVLA-VQ (baseline) [53]	79	0.0	1.5B
+Dual-Encoder [9]	75	2.75	1.5B
+Dual-Encoder (new DINOv2+SigLIP)	79	3.75	2.5B
+MAPS (Ours)	82	4.75	1.5B

Comparison with Dual Encoders. The comparison against different dual-encoder variants is presented in Tab. 8. Notably, MAPS achieves the strongest performance among all compared methods while remaining the most compute-efficient. Dual-encoder approaches require instantiating a second trainable copy of the frozen vision encoder, which effectively doubles the parameter count of the vision stack. We implement this as the **Dual-Encoder (new DINOv2+SigLIP)** variant. A more efficient variant is proposed by [9], in which one of the vision encoders is frozen while the other is kept trainable. We implement this as the **Dual-Encoder** variant. As shown in the table, neither dual-encoder variant matches the ID or OOD performance of MAPS. **Dual-Encoder** performs worst, and **Dual-Encoder (new DINOv2+SigLIP)** is slightly better but incurs substantially higher architectural cost. MAPS offers the best performance without additional cost.

Comparison with Weight Interpolation. Weight interpolation methods such as [8] also introduces additional overhead: it depends on having a finetuned set of VLA weights available for interpolation, which means one training run to obtain those weights and another to perform the interpolation process.

Comparison with Bi-Level Optimization. Bi-level optimization approaches that hyper-optimize the constraints be-

Table 9. SimplerEnv results on NORA.

Model	Avg. ID	Visual	Novel Object	Novel Category	Avg. OOD
NORA	35.68	37.50	15.30	8.33	20.38
+MAPS (Ours)	36.43	36.10	40.27	15.27	30.54

tween the fine-tuned and pre-trained weights, such as [15], incur roughly twice the training time. Under this setting, the resulting policy attains 79% ID success and 4% OOD success using VLA-Adapter, and 0% ID and OOD success using MiniVLA-VQ, both of which fall short of MAPS while requiring significantly higher computational cost.

10. Additional Results

Tab. 9 evaluates MAPS on NORA [54] which consists of a Qwen2.5-VL [6] visual-language backbone with FAST action tokenizer [29], in contrast to the OpenVLA family which uses DINOv2 and SigLIP vision encoders with LLaMA language decoder. We train NORA on 8 A40 GPUs with a global batch size of 64 for 16k steps, taking approximately 10 hours. MAPS improves performance by +10% on OOD settings and +1% on ID settings in SimplerEnv. This confirms the architecture-agnosticism of our method.

11. Intuition Discussion

We provide *additional intuition* by analyzing gradient norms from early to late layers (Fig. 9). We observe a clear hierarchical trend with a monotonic and approximately linear increase in gradient magnitude across layers/modules, indicating parameters with larger gradients require greater adaptation, which aligns with the linearly decaying constraint used in MAPS.

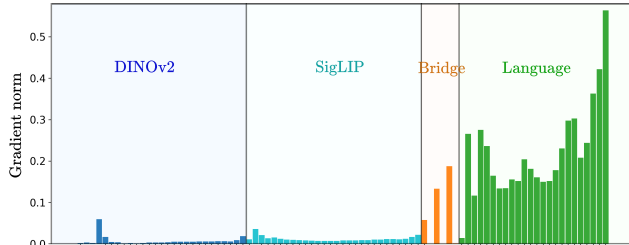
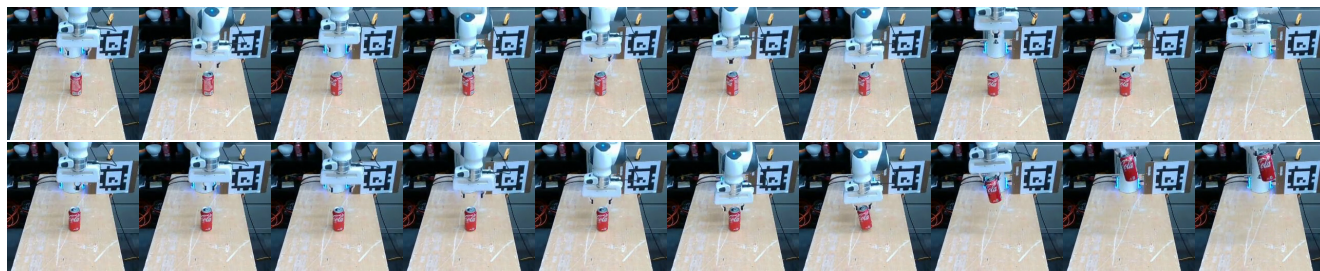
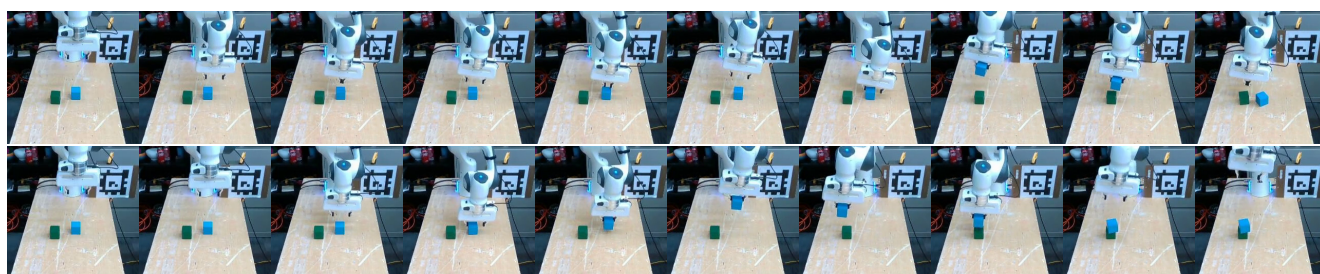


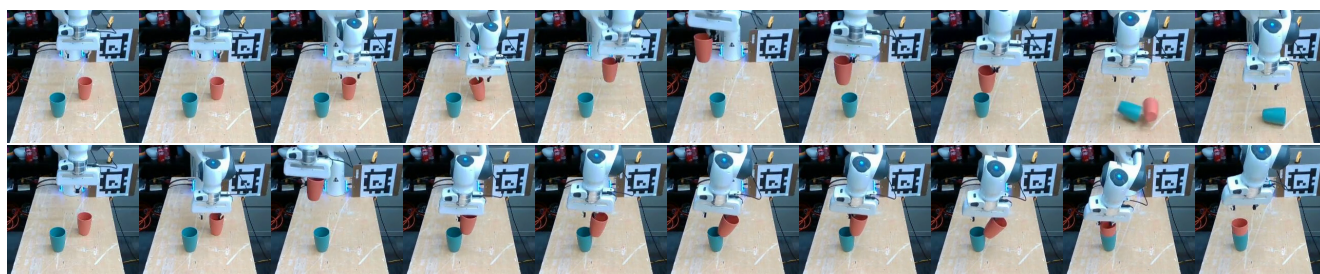
Figure 9. Gradient norm of MiniVLA-OFT on SimplerEnv averaged over first 50 steps.



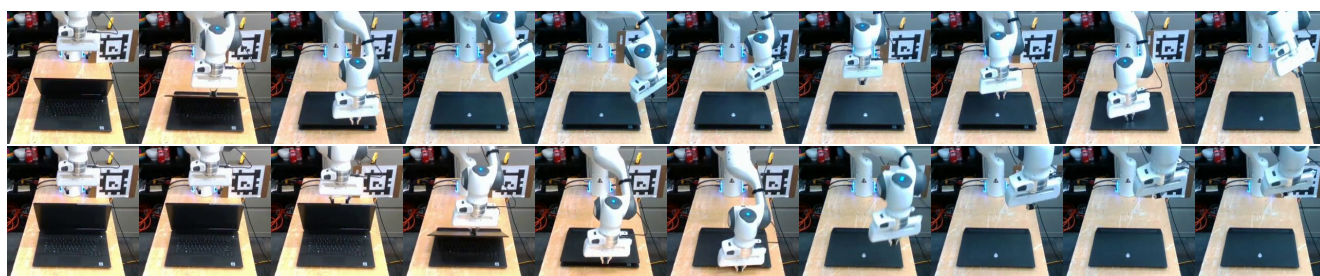
(1) Grab the Coke Can and Lift It Up



(2) Stack the Blue Block on the Green Block

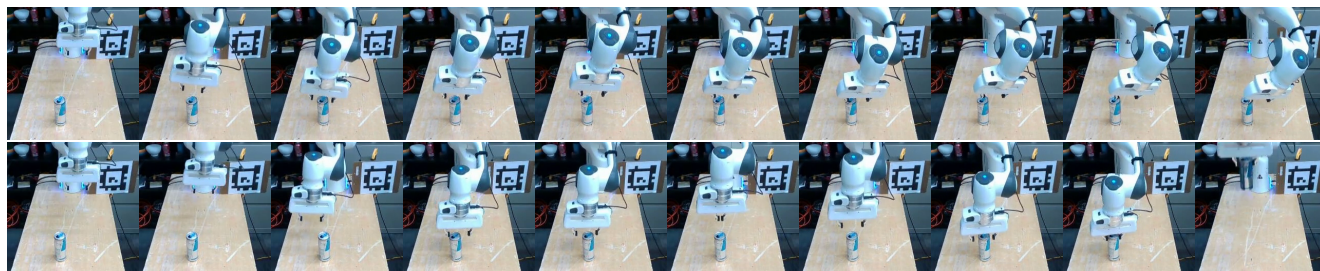


(3) Stack the Orange Cup onto the Green Cup

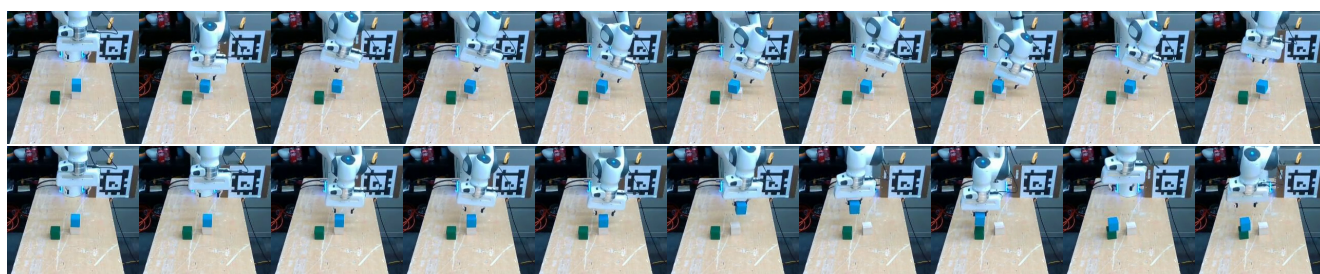


(4) Close the Laptop Lid Fully

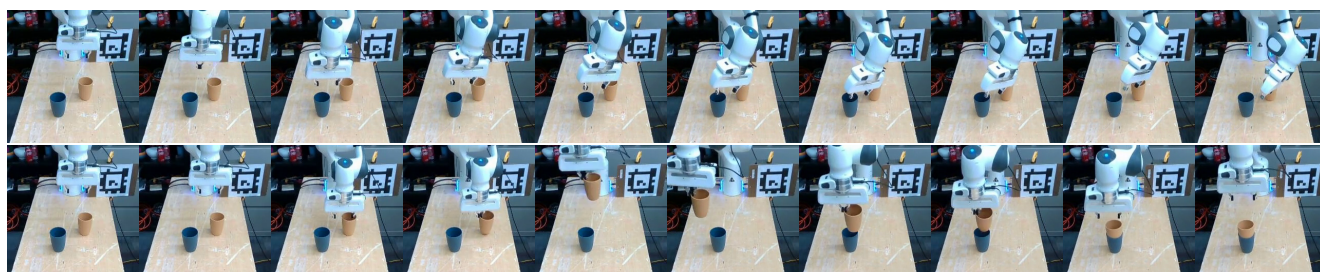
Figure 10. ID tasks: baseline (top) vs. MAPS (bottom).



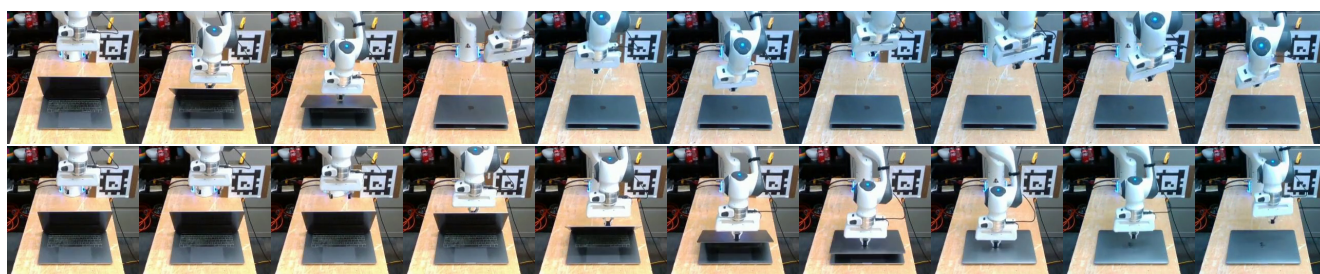
(5) Grab the Red Bull Can and Lift It Up



(6) Stack the Blue Block on the Green Block



(7) Stack the Yellow Cup onto the Blue Cup



(8) Close the Laptop Lid Fully

Figure 11. OOD tasks: baseline (top) vs. MAPS (bottom).