MicroVLA: Edge-Deployable Vision Language Action at 10M Parameters

Ngseo Kim[1], Junghyun Kim[1], Gi-Cheon Kang[2,†], Youngjae Yu[1,†], Byoung-Tak Zhang[1,3,†]
[1]Seoul National University [2]Ajou University [3]Tommoro Robotics

***Abstract*—Deploying vision-language-action (VLA) models on edge controllers in production manipulation cells remains challenging under tight memory and control-rate budgets. A key bottleneck is that contemporary VLAs span hundreds of millions to several billion parameters — orders of magnitude above what an embedded controller can host. To this end, we study two aspects: (1) how compact a VLA can be made by a coordinated reduction across all three branches, and (2) how to recover the policy quality that a sub-15 M-parameter trunk loses under plain behavior cloning. Specifically, we introduce MicroVLA, a 12.4 M-parameter VLA that pairs a static text-embedding lookup with a small distilled vision encoder and a single-forward regression head emitting a multi-step action chunk. We train MicroVLA with action-L1 behavior cloning augmented by an auxiliary self-supervised contrastive loss over the fused conditioning. On the full LIBERO benchmark (2000 closed-loop episodes), MicroVLA reaches 90.2% success, with a forward latency of 13.1 ms on a desktop GPU and 75 ms on a CPU-only forward — the smallest published VLA above 90% LIBERO Avg, at one-to-two orders of magnitude fewer deployed parameters than the next-smallest competitor, and an order-of-magnitude lower forward latency than 7 B-class VLAs on identical hardware.**

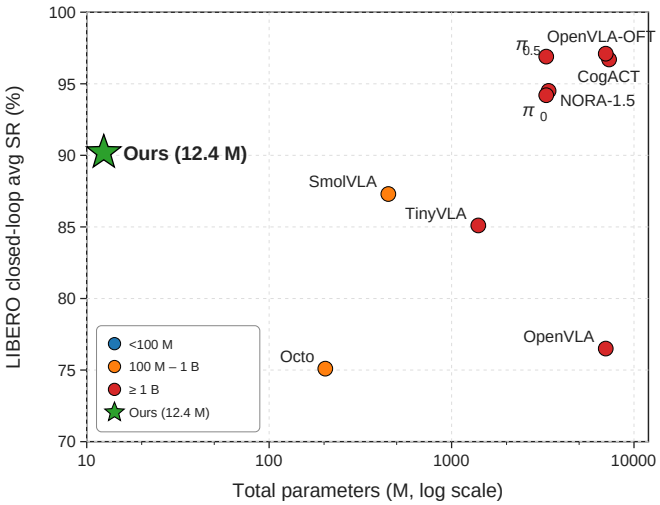

Fig. 1. LIBERO closed-loop average SR vs. total parameter count (log scale).

I. Introduction

Contemporary vision–language–action (VLA) models reach strong closed-loop performance on standard manipulation benchmarks [1, 6, 14], but at hundreds of millions to several billion parameters and inference budgets sized for desktop or server-class accelerators. Production manipulation cells, in contrast, often target embedded controllers with on the order of megabytes of on-device memory, tight control-rate budgets, and update cycles measured in days rather than seconds. Even the most compact published VLAs ship hundreds of megabytes of weights and take hundreds of milliseconds per forward pass on commodity GPUs — numbers that sit far above what an embedded controller can reasonably absorb. The remaining gap in production therefore appears to be **more about deployment footprint and per-step latency than about modelling capacity**. We close most of that gap on the model side. The lever is a coordinated reduction across all three branches: the language backbone is replaced by a static sentence-embedding lookup [16] — the noun-phrase task descriptions standard in manipulation benchmarks do not require an LLM — the vision branch by a tiny multi-teacher distilled encoder [13], and the action head by a single-shot regression that emits a multi-step action chunk in one forward. At this scale plain behavior cloning trails larger VLAs by a noticeable margin; we recover the gap with an auxiliary InfoNCE contrastive term over the fused conditioning [15], and report a controlled 2×2 ablation isolating which axis of the contrast — positive type and augmentation — carries the recovery. The resulting recipe — **MicroVLA**, 12.4 M deployed parameters — reaches 90.2% closed-loop success on the full LIBERO benchmark [10], runs in 13.1 ms on a desktop GPU and 75 ms on a CPU-only forward on the same machine, and is an order of magnitude faster than 7 B-class VLAs on identical hardware. **Our contributions:** (i) a three-branch shrinking recipe two-to-three orders of magnitude below contemporary VLAs while preserving benchmark performance (Figure 1); (ii) same-hardware GPU and CPU deployment numbers measured by us; (iii) a controlled 2×2 SSL ablation suggesting that augmentation, rather than the contrast itself, is the dominant axis at this scale; and (iv) a same-recipe scaled-up control (137 M) bounding how much of the gap is a capacity effect rather than a BC-objective limitation. The recipe targets the L2P research questions on industrial speed (Q1), representations supporting repeatability (Q2), and evaluation under realistic hardware constraints (Q3).

II. Related Work

A. Modern VLAs

Modern VLAs inherit a pretrained vision–language backbone and finetune an action head on robot demonstration data, reaching strong closed-loop performance at hundreds of millions to billions of parameters: OpenVLA [6] (7 B) and its optimized variant OpenVLA-OFT [7], π_0 [1] (3.3 B) and the followup $\pi_{0.5}$ [12], SmolVLA [14] (450 M), CogACT [9] (7.3 B), and NORA-1.5 [4] (3.4 B). TinyVLA [18] (1.4 B) and Octo [11] (203 M) target the deployment gap by shrinking the decoder LLM or the policy transformer, but still ship hundreds of megabytes to gigabytes at inference — two-to-three orders of magnitude above an embedded controller's budget.

B. Auxiliary self-supervised losses

ATC [15] popularised a temporal-InfoNCE auxiliary loss for reinforcement learning; recent VLA-side work [8, 17] revisits contrastive regularisation for vision–language policies.

C. Static sentence embeddings

Model2Vec [16] distills sentence-transformer embeddings into a static lookup, standard in NLP retrieval. Such lookups replace the inference-time transformer forward with a constant-time table read, making them a natural fit for downstream policies that can absorb a fixed sentence-level representation without per-token attention.

III. Method

A. Shrinking the model

VLA parameter counts are dominated by the language branch (L $>$ V $>$ A). Standard manipulation benchmarks, however, use predominantly noun-phrase task descriptions ("pick up the black bowl and put it on the plate") with little compositional or referential ambiguity. We exploit this asymmetry along three axes. **Static language embedding:** We replace the LLM with POTION-base [16], a 2.0 M-parameter static embedding table distilled from a sentence-transformer teacher. Instructions are mean-pooled over token-level embeddings to a 64-d vector; no transformer over text runs in the forward path. **Tiny multi-teacher vision backbone:** The vision branch is Theia-tiny [13], a ViT-tiny (5.5 M parameters) trained by multi-teacher distillation from DINOv2, CLIP, and DepthAnything. Aligning pretraining with robotics-relevant visual structure allows a small encoder to match much larger baselines. We share Theia weights between the agentview and wrist streams. **Single-shot regression action head:** The action head is a 5-block MLP-ResNet that maps the 1600-d fusion output (FiLM-conditioned vision tokens pooled through 8 Perceiver latent queries [5]) to a 16-step, 7-DoF action chunk in one forward pass. We avoid transformer decoders [19] (no measurable gain at chunk length 16) and diffusion samplers [3] (5–10$\times$ inference forwards); a one-step rectified-flow head is a natural alternative we leave for future work.

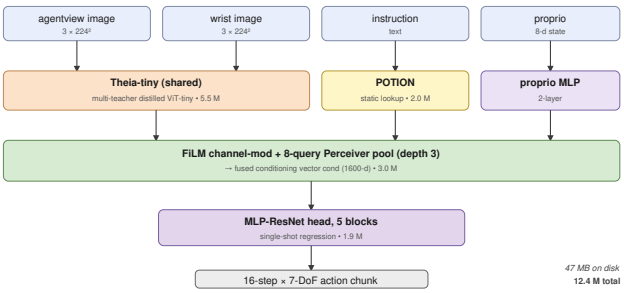

Fig. 2. End-to-end MicroVLA architecture (12.4 M deployed, 47 MB on disk in fp32).

B. Plain BC is not enough

The minimal model of Figure 2 trained only with masked L_1 action regression reaches 77.55% closed-loop SR on the full LIBERO benchmark — 9.8 pp below SmolVLA (87.3) and 16.6 pp below π_0 (94.2). **Big-BC control:** To check whether this is a BC-objective failure or a capacity failure, we scale the vision branch to Theia-base (85.8 M, ViT-B backbone) and deepen the head to 12 blocks of width 1024, keeping the BC objective untouched. The resulting 137 M-parameter model reaches **95.50 %** closed-loop SR on the same benchmark (`spatial` 96.4 / `object` 99.0 / `goal` 97.0 / `long` 89.6). The 12.4 M model's deficit therefore appears to be a representational under-fit rather than a BC failure: the trunk has the right shape but receives insufficient gradient signal to shape its features under the small training-data budget of LIBERO. A natural remedy is to add a stronger gradient signal at the representation level. We try two: data augmentation alone, and an auxiliary self-supervised loss.

C. Auxiliary self-supervised loss

We augment the action loss with an auxiliary InfoNCE term applied to the fused conditioning vector cond (1600-d):

$$\mathcal{L} \;=\; \mathcal{L}_{\text{action}} \;+\; \lambda\, \mathcal{L}_{\text{InfoNCE}}(\mathrm{cond}_1,\, \mathrm{cond}_2)\,, \qquad (1)$$

where $(\mathrm{cond}_1, \mathrm{cond}_2)$ is a positive pair. Two axes of positive-mining and an augmentation switch define a 2×2 grid. **Positive type:** (i) *Temporal positives*: a second frame from the same episode at a different timestep (the ATC [15] setting). (ii) *Same-frame positives*: two independent views of the *same* frame (the SimCLR [2] setting). **Augmentation switch:** Independently of positive type, augmentation can be *on* (random resized crop scale 0.85–1.0, brightness/contrast jitter, applied to agentview and, for same-frame pairs, also wrist) or *off*. Table I reports the full grid alongside the BC, BC+aug, and 137 M capacity controls. The temporal positives row gives ATC (aug) and TCN (no aug); the same-frame row gives SimCLR (aug). All rows share the training recipe: b=128, chunk 16, 400 k steps on mixed 4-suite data. Three observations: (i) *augmentation alone is not enough* — BC + aug recovers only 3.5 pp over plain BC (77.6 $\rightarrow$ 81.1), and still leaves a 9.1 pp gap to ATC; (ii) *temporal contrast without augmentation under-performs* — TCN sits 19.8 pp

TABLE I
SSL PAIR-MODE ABLATION AND CAPACITY CONTROL. CLOSED-LOOP SR IS THE FULL LIBERO BENCHMARK (n=2000 EPISODES).

Variant	Positive type	Augment	Params	SR (%)
no SSL (plain BC)	—	no	12.4 M	77.6
BC + aug only	—	yes	12.4 M	81.1
big plain BC	—	no	137 M	95.5
SimCLR-style	same-frame	yes	12.4 M	84.4
TCN	temporal	no	12.4 M	70.4
ATC	temporal	yes	12.4 M	**90.2**

TABLE II
LIBERO CLOSED-LOOP SUCCESS RATE (%). BOLD MARKS THE PER-COLUMN BEST. PARAMETER COUNTS INCLUDE THE TEXT ENCODER.

Model	Params	Sp.	Obj.	Goal	Long	Avg
Octo [11] (incl. T5 enc.)	203 M	78.9	85.7	84.6	51.1	75.1
OpenVLA [6]	7 B	84.7	88.4	79.2	53.7	76.5
TinyVLA [18]	1.4 B	89.9	95.1	95.0	60.4	85.1
SmolVLA [14]	450 M	90.0	96.0	92.0	71.0	87.3
π_0 [1]	3.3 B	96.8	**98.8**	95.8	85.2	94.2
NORA-1.5 [4]	3.4 B	97.3	96.4	94.5	89.6	94.5
CogACT [9]	7.3 B	98.4	98.4	96.4	93.4	96.7
$\pi_{0.5}$ [12]	3.3 B	**98.8**	98.2	**98.0**	92.4	96.9
OpenVLA-OFT [7]	7 B	97.6	98.4	97.9	**94.5**	**97.1**
Ours	**12.4 M**	91.0	93.8	92.2	83.8	90.2

below ATC despite a comparable action-loss curve during training, suggesting that augmentation is the dominant axis of the ATC recipe; (iii) *the 12.4 M deficit is a capacity effect, not a BC failure* — the same recipe scaled to 137 M with pure BC reaches 95.5 %, 5.3 pp above ATC at an order of magnitude more parameters.

IV. RESULTS

Table II reports closed-loop SR on the full LIBERO benchmark [10] (50 ep/task × 10 tasks per suite; Avg is the 2000-episode mean), with the most relevant size-class baselines. Parameter counts include the text encoder (e.g., Octo's 203 M folds in its T5). The 137 M big-BC capacity control of Table I lands at 95.5 % Avg — omitted here to keep the published-baseline comparison clean. Table III reports deployment-side numbers: single forward at b=1 on an RTX 3090 (competitors at bf16, ours at fp32) for the GPU column, and a single forward in fp32 for the CPU column. π_0 uses the LeRobot port `lerobot/pi0_base` (4.0 B with the port's normalization layers; their paper reports 3.3 B). **Summary:** MicroVLA is the **smallest published VLA above 90 % LIBERO Avg** — 274× smaller than the next-smallest competitor (NORA-1.5 at 3.4 B), 2.9 pp above SmolVLA (87.3 %) at $1/36$ its parameters, and 18× faster than OpenVLA-7B on the same RTX 3090. On a CPU-only forward (8 threads, Threadripper 2990WX), the same model takes **75 ms**, whereas OpenVLA takes 13.0 s on the same machine (∼173× slower) and SmolVLA takes 4.44 s (∼59× slower). After amortization, the per-step cost drops to ∼9.4 ms — within a real-time manipulation budget even without a discrete GPU. Larger recent VLAs (OpenVLA-OFT, $\pi_{0.5}$, CogACT, NORA-1.5) reach higher Avg, but at 250–550× our parameter count. What the data suggest is less a new architectural primitive than a coordinated compression — a shrunk language branch, a shrunk vision branch, a single-shot head — recovered by an SSL aux loss whose principal ingredient is augmentation rather than the contrast itself.

TABLE III
INFERENCE-SIDE COST.
(RTX 3090 GPU b=1; THREADRIPPER 2990WX CPU, 8 THREADS.)

Model	Ckpt	VRAM (b=1)	GPU	CPU
OpenVLA [6]	14.1 GB	14.07 GB	238 ms	13.0 s
SmolVLA [14]	1.0 GB	0.90 GB	290 ms	4.44 s
π_0 [1]	13.0 GB	15.25 GB	357 ms	19.1 s
TinyVLA [18]	2.46 GB	2.51 GB	88.6 ms	3.09 s
Ours (12.4 M)	**47 MB**	**51 MB**	**13.1 ms**	**75.2 ms**

V. CONCLUSION

We report an end-to-end VLA recipe that fits a production-side compute budget — **12.4 M deployed parameters, 47 MB on disk, 13.1 ms GPU forward, 75 ms CPU forward** — and a controlled SSL ablation suggesting that *augmentation* is the dominant axis of the auxiliary loss at this scale. The model is the smallest published VLA above 90 % LIBERO Avg (90.2 %), stays deterministic and single-shot at inference, and fits within a real-time manipulation budget on CPU. The immediate next step is a CPU real-robot port; longer-term directions include calibrated int8 deployment and richer proprioception-aware contrastive objectives.

LIMITATIONS

No real-cell validation. The deployment numbers in Table III are measured against synthetic inputs on a single workstation; industrial deployment will require end-to-end closed-loop measurements. *Static text.* POTION is sufficient for LIBERO-style noun-phrase task descriptions but would degrade on compositional spatial language ("put X between Y and Z"). *No int8 deployment.* A torchao or TensorRT int8 path would shrink the checkpoint further but is not run here.

AI USAGE STATEMENT

A large language model assisted with copy-editing across all sections, drafted the initial LaTeX scaffolding for tables and figures, and helped author training and evaluation code. All experimental design, numerical results, and final scientific claims were authored and verified by the (anonymous) authors.

ACKNOWLEDGMENT

This work was partly supported by grants from the IITP (RS-2026-25522885, Development of a World Foundation Model for Training and Development of Physical AI Systems/10%, RS-2022-II220951-LBA/10%, RS-2022-II220953-PICA/10%, RS-2026-25553157-MIACC/10%), the NRF (RS-2024-00353991-SPARC/15%, RS-2023-00274280-HEI/15%), the KEIT (RS-2025-25453780/15%), and the KIAT (RS-2025-25460896/15%), funded by the Korean government.

REFERENCES

[1] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[2] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*, 2020.

[3] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems (RSS)*, 2023.

[4] Chia-Yu Hung, Aaron Hua, Navonil Majumder, Lidong Bing, and Soujanya Poria. NORA-1.5: Direct preference optimization for robust vision-language-action models. *arXiv preprint arXiv:2511.14659*, 2025.

[5] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *International Conference on Machine Learning (ICML)*, 2021.

[6] 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. In *Conference on Robot Learning (CoRL)*, 2024.

[7] Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

[8] Taeyoung Kim, Jimin Lee, Myungkyu Koo, Dongyoung Kim, Kyungmin Lee, Changyeon Kim, Younggyo Seo, and Jinwoo Shin. Contrastive representation regularization for vision-language-action models. *arXiv preprint arXiv:2510.01711*, 2025.

[9] 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, Jiang Bian, Dahua Zhang, Yu Liu, and Baining Guo. CogACT: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024.

[10] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[11] Octo Model Team. Octo: An open-source generalist robot policy. In *Robotics: Science and Systems (RSS)*, 2024.

[12] Physical Intelligence Team. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

[13] Jinghuan Shang, Karl Schmeckpeper, Brandon B. May, Maria Vittoria Minniti, Tarik Kelestemur, David Watkins, and Laura Herlant. Theia: Distilling diverse vision foundation models for robot learning. In *Conference on Robot Learning (CoRL)*, 2024.

[14] 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. *arXiv preprint arXiv:2506.01844*, 2025.

[15] Adam Stooke, Kimin Lee, Pieter Abbeel, and Michael Laskin. Decoupling representation learning from reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2021.

[16] Stéphan Tulkens and Thomas van Dongen. Model2vec: Distilling sentence transformers into static embeddings. Model2Vec open-source library, 2024.

[17] Junlin Wang and Zhiyun Lin. Grounding bodily awareness in visual representations for efficient policy learning. *arXiv preprint arXiv:2505.18487*, 2025.

[18] Junjie Wen, Yichen Zhu, Jinming Li, Minjie Zhu, Kun Wu, Zhiyuan Xu, Ning Liu, Ran Cheng, Chaomin Shen, Yaxin Peng, Feifei Feng, and Jian Tang. TinyVLA: Towards fast, data-efficient vision-language-action models for robotic manipulation. *IEEE Robotics and Automation Letters (RA-L)*, 2024.

[19] Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Robotics: Science and Systems (RSS)*, 2023.