CTRL-STEER: Closed-Loop Neuron Activation Control in Vision-Language-Action Models

Abhijith Babu
Florida International University
ababu012@fiu.edu

Ramneet Kaur
SRI International
ramneet.kaur@sri.com

Nathaniel D. Bastian
United States Military Academy
nathaniel.bastian@westpoint.edu

Olivera Kotevska
Oak Ridge National Laboratory
kotevskao@ornl.gov

Susmit Jha
SRI International
susmitjha@berkeley.edu

Yanzhao Wu
Florida International University
yawu@fiu.edu

Sumit Kumar Jha
University of Florida
sumit.jha@ufl.edu

Anirban Roy
SRI International
anirban.roy@sri.com

Abstract

Vision-Language-Action (VLA) models enable test-time behavioral steering via neuron-level interventions, but existing methods use fixed strengths and operate in open loop. This static modulation fails under evolving task dynamics, leading to overcorrection, oscillations, and reduced task success—especially for temporal attributes like speed. We propose CTRL-STEER, a control-theoretic framework that casts activation steering as closed-loop feedback with adaptive, time-varying interventions. Instead of assuming neurons encode temporal concepts, we steer along motion-aligned residual directions and regulate intervention magnitude via feedback. We instantiate this with both PID and reinforcement learning controllers that jointly optimize concept adherence and task success. Experiments on fine-tuned OpenVLA policies across four LIBERO suites show improved stability and a better steering–success trade-off over baselines, without retraining the base model.

1. Introduction

Vision-Language-Action (VLA) models [5, 18] enable end-to-end robotic control by jointly processing visual observations and language instructions to generate low-level actions. Typically, a pretrained Vision-Language Model (VLM) serves as a perceptual-semantic backbone, paired with an autoregressive action decoder [32]. Early work showed that pretrained VLMs can be adapted for robotic control via tokenized actions [32], while recent models such as OpenVLA and π_0 integrate large pretrained backbones like LLaMA [5, 18, 26] with robot interaction data. Despite strong semantic generalization [5, 7, 15], these models struggle with physical attributes such as speed, spatial constraints, and fine-grained configurations [4, 8, 14, 17, 24, 28, 30], which require precise continuous control.

We address this gap with an inference-time steering framework that augments execution with control objectives without retraining, avoiding costly data collection [13, 23, 27, 29, 31]. Prior work on activation-level steering [10, 11, 22], identifies concept-aligned neurons or directions and applies fixed scaling [12, 16, 21, 33]. However, such open-loop interventions ignore evolving task dynamics; along with polysemanticity [6, 9], they often lead to overcorrection and degraded task success.

We instead introduce a control-theoretic view of activation steering, treating neuron modulation as a feedback-regulated process rather than a static perturbation. Drawing from classical closed-loop control in robotics [1, 3], we propose **CTRL-STEER**, which dynamically adjusts concept-aligned neuron activations based on execution feedback.

We first identify motion-relevant neurons by projecting feed-forward value vectors into vocabulary space and selecting those aligned with the control features, extending [12] from single features to concept-level steering. Instead of fixed coefficients, we compute time-varying interventions. A PID controller [2] adjusts neuron scaling based on tracking error, providing a simple mechanism for stabilizing steering. To capture longer-horizon effects, we further introduce an RL controller trained with PPO [19, 25], which learns a nonlinear policy to jointly optimize steering and task success.

Our approach enables control over continuous attributes

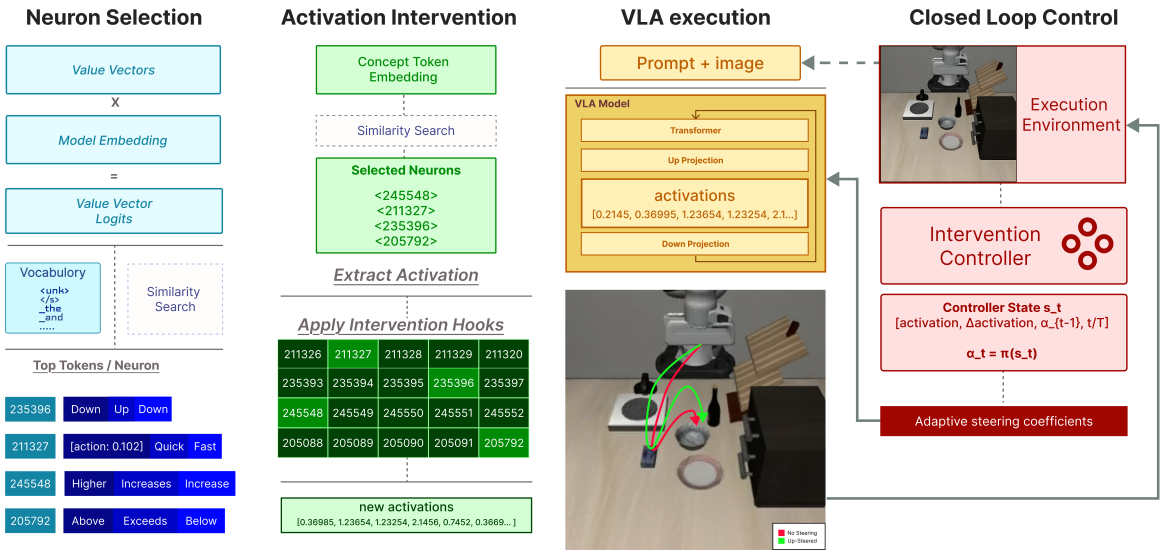

Figure 1. We present CTRL-STEER for controlled steering of VLA models. First, we identify feature-aligned neurons by projecting FFN value vectors to the token space and selecting neurons whose semantic embeddings align with the features. During inference, these neurons are steered through activation intervention. The proposed closed-loop controller learn to adaptively regulate the steering based on the feedback from the environment and reward from task completion. This enables dynamic control of robot behavior while maintaining a high task success rate during VLA execution.

while preserving task performance. PID-based steering maintains success near baseline (71.37%) compared to a collapse under static steering (1.8%). RL further improves success to 73.88% (height) and 76.12% (speed), demonstrating that closed-loop regulation resolves the trade-off between concept enforcement and task completion.

We make the following contributions.

1. **Closed-Loop Steering.** We introduce a control-theoretic formulation of activation steering in VLA models, enabling dynamic, feedback-driven regulation of concept-aligned neurons.
2. **RL-based Control.** We develop an RL controller that adaptively modulates neuron activations to jointly optimize steering objectives and task success.
3. **Empirical Validation.** On LIBERO, PID-based steering preserves success close to baseline (71.38%) while static steering drops to 27.38% (height) and 1.88% (speed). RL further improves success to 73.88% (height) and 76.12% (speed) while maintaining desired steering.

2. Approach

We study how interpretable internal concepts in VLA models can control robot behavior. As shown in Fig. 1, CTRL-STEER first identifies neurons aligned with motion-related concepts and then regulates them online with feedback control to steer behavior while preserving task success.

2.1. Interpretable Concept Identification

We build on mechanistic interpretability for VLA models [10, 12]. In each transformer FFN layer, the output can be written as a sum of neuron activations and value vectors:

$$\mathrm{FFN}^{\ell}(r^{\ell}) = \sum_{i=1}^{d_m} m_i^{\ell} v_i^{\ell}. \tag{1}$$

Following [12], we project each neuron's value vector into vocabulary space and construct a semantic embedding

$$sem_i^{\ell} = \sum_{w \in \mathrm{TopK}(p_i^{\ell})} p_i^{\ell}(w)\, e(w), \tag{2}$$

where p_i^{ℓ} is the probability distribution of tokens after projection. This captures the dominant concept promoted by that neuron.

Unlike prior work [12], we identify neurons for a broader motion concept using a set of representative tokens, e.g., $\{\texttt{up}, \texttt{down}, \texttt{left}, \texttt{right}, \texttt{forward}, \texttt{backward}\}$. We retrieve the candidate intervention set by k-NN matching:

$$\mathcal{S} = \bigcup_{w \in \mathcal{T}_{\mathcal{C}}} \mathrm{kNN}_k\big(e(w)\,;\, \{sem_i^{\ell}\}_{\ell,i}\big)\,. \tag{3}$$

Because neurons can be polysemantic [9], we manually filter the candidates and retain the 10 most consistent neurons.

This interpretability step provides the link between internal concept representations and controllable robot behavior.

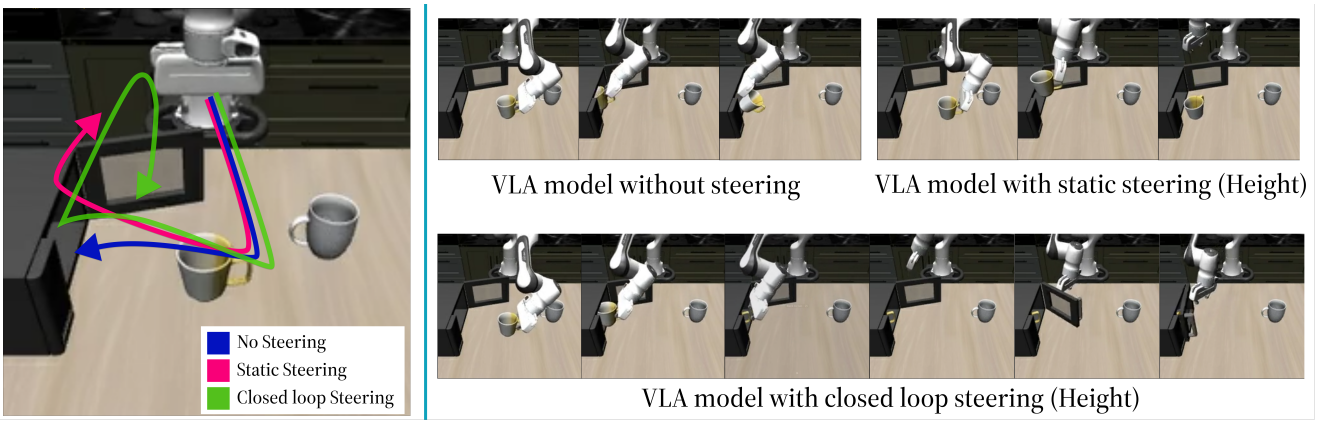

Figure 2. **Left**: Example of steering the height concept for the task: *Put the yellow and white mug in the microwave and close it.* The goal is to steer the arm to the desired direction, e.g., a larger height. **Right**: i. The unsteered VLA model follows a low trajectory and fails to place the mug correctly. ii. Static steering of VLA drastically increases the trajectory height and collides with the microwave top. iii. CTRL-STEER - our closed-loop RL-based steering dynamically adjusts neuron coefficients, achieving sufficient vertical clearance for successful placement while enabling correct door closure.

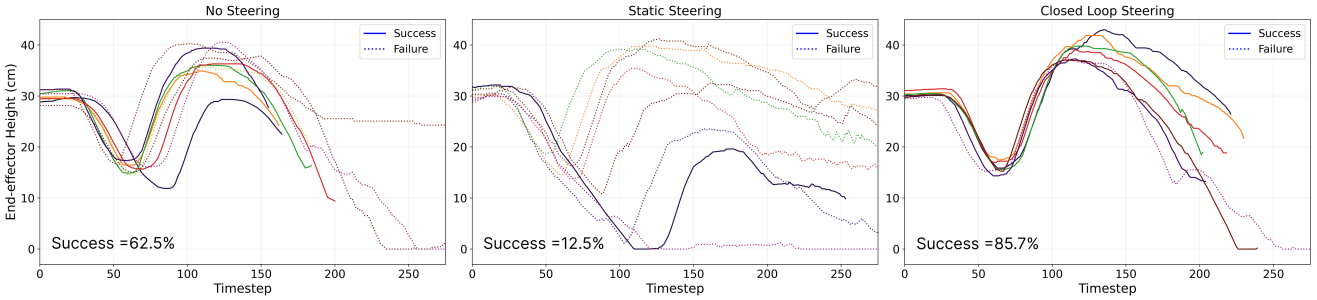

Figure 3. We present the effect of steering height for the task *'pick up the book and place it in the back compartment of the caddy'*. Unsteered model keeps the end-effector lower. The static steered model increases the height, but fails to achieve a high task success rate. Closed-loop steering keeps the end-effector higher while completing the task.

However, for temporal concepts such as speed, identifying aligned neurons is not sufficient: the intervention strength must also vary over time as the task evolves.

2.2. Closed-Loop Activation Steering

Prior methods use a fixed intervention coefficient [12]:

$$\tilde{m}_i^\ell = \begin{cases} \alpha & \text{if } (\ell, i) \in \mathcal{S}, \\ m_i^\ell & \text{otherwise.} \end{cases} \tag{4}$$

This open-loop strategy enforces a concept, but harms task completion because it ignores changing task dynamics.

We cast steering as a feedback control problem. Let c^t denote the concept value at time t and define error as

$$e^t = c^* - c^t, \tag{5}$$

where c^* is the desired concept target. Rather than applying a fixed coefficient, we compute time-varying neuron interventions α^t over the selected neurons.

PID control. We first use a PID controller [2]:

$$\alpha_{PID}^t = K_P e^t + K_I \sum_{\tau=0}^{t} e^\tau + K_D(e^t - e^{t-1}), \tag{6}$$

which adapts steering based on instantaneous, accumulated, and changing error. This yields a simple closed-loop mechanism for stabilizing concept control. We bound interventions to $[0, 20]$ to avoid overly strong perturbations.

RL control. PID is reactive and does not explicitly optimize long-horizon task outcomes. We therefore introduce an RL controller that jointly optimizes steering and task success. The policy takes as input

$$s_t = [a_t, \Delta a_t, \alpha^{t-1}, t/T], \tag{7}$$

and predicts neuron-level interventions

$$\alpha_{RL}^t = \pi_\theta(s_t). \tag{8}$$

Table 1. Steering of end-effector height averaged across LIBERO task suites. We compare unsteered OpenVLA, static steering ($C \in \{5, 10, 20\}$), and CTRL-STEER (PID, RL). Height (m) denotes average end-effector height, AAT captures Area Above Threshold (active motion), and SR (%) is task success rate.

Method	Height (m)	AAT	SR (%)
OpenVLA [18]	0.792 ± 0.096	96.200 ± 57.110	71.38
C=5 [12]	0.791 ± 0.095	91.457 ± 53.947	70.75
C=10 [12]	0.789 ± 0.095	101.594 ± 79.486	71.25
C=20 [12]	0.756 ± 0.087	106.732 ± 80.042	27.38
CTRL-STEER(PID)	0.791 ± 0.094	97.677 ± 73.516	71.00
CTRL-STEER(RL)	0.787 ± 0.096	96.070 ± 65.048	**73.88**

Table 2. Temporal steering of end-effector speed averaged across LIBERO task suites. We compare unsteered OpenVLA, static steering ($C \in \{5, 10, 20\}$), and CTRL-STEER (PID, RL). Speed (cm/s) denotes average end-effector speed, SAT captures Speed Above Threshold (active motion), and SR (%) is task success rate.

Method	Speed (cm/s)	SAT (cm/s)	SR (%)
OpenVLA	14.238 ± 3.270	3.008 ± 0.923	71.38
C=5	14.404 ± 3.157	3.027 ± 1.017	71.62
C=10	14.140 ± 3.248	3.000 ± 0.953	57.88
C=20	12.308 ± 1.966	4.186 ± 1.486	1.88
CTRL-STEER(PID)	14.260 ± 3.256	2.999 ± 0.942	72.50
CTRL-STEER(RL)	14.207 ± 3.308	2.950 ± 0.971	**76.12**

The reward balances concept control and task success:

$$r_t = r_{steer}(t) + \lambda \cdot r_{task}. \tag{9}$$

We train task-specific policies initialized from PID trajectories, allowing coordinated and nonlinear modulation of concept-aligned neurons.

3. Experiments

We evaluate on all four LIBERO task suites [20]: LIBERO-Goal, LIBERO-Object, LIBERO-Spatial, and LIBERO-Long (libero-10 subset). These tasks span compositional reasoning, spatial relations, and long-horizon execution, making them suitable for evaluating both state-based and temporal steering. We use OpenVLA [18] (7B, LLaMA-2 backbone [26]), fine-tuned per LIBERO task suite. We evaluate (i) *height* (state-based) and (ii) *speed* (temporal).

For height: mean height, 95^{th} percentile, and AAT (Area Above Threshold) [12]. For speed: mean speed and SAT (Speed Above Threshold) [12].

3.1. Results

Height Steering. Let $c^t = h^t$ and $c^* = 2h_0$. For RL, we use h^t as reward with trade-off $\lambda \in \{100, 200, 500, 1000\}$. Mean height is sensitive to low-motion phases, while max height depends on initialization; thus we report 95^{th} percentile and AAT for sustained elevation. As shown in Tab. 1, static steering increases height but degrades success, especially at high coefficients ($C = 20$). In contrast, CTRL-STEER maintains comparitive performance while achieving comparable elevation as shown in Fig. 3.

Speed Steering. We set $c^t = s^t$ and $c^* = 30$ cm/s. Since average speed is dominated by stationary phases, we use SAT defined as $\text{mean}(s^t \mid s^t > s_{thr})$ with $s_{thr} = 20$ cm/s. Results in Tab. 2 show that static steering with large coefficients significantly harms task success, while CTRL-STEER achieves stable speed modulation with minimal degradation.

Analysis. Static steering exhibits a clear trade-off: increasing intervention strength improves concept expression but severely reduces success. With $C = 20$, average success drops from **71.37%** (unsteered) to **27.37%** (height) and **1.8%** (speed). In contrast, closed-loop control maintains performance: PID achieves **71%** (height) and **72.5%** (speed), while RL further improves to **73.88%** and **76.12%**, respectively, surpassing the baseline.

Steering–Success Trade-off. CTRL-STEER adaptively regulates intervention strength, reducing it when strict concept enforcement conflicts with task completion. For example, in the microwave task (Fig. 2), static steering causes collisions due to excessive height, whereas CTRL-STEER attenuates intervention during insertion, preserving success. Similar behavior is observed for speed, where excessive motion destabilizes interactions; adaptive control mitigates this, unlike static steering.

RL without PID Initialization. Without PID warm-start, RL performance drops: task success rate and steered concept metrics reduced while training RL policies without PID initialization, underperforming both baseline and other methods. This highlights the importance of PID initialization for stable and efficient learning.

4. Conclusion

We present a closed-loop activation steering framework for VLA models that enables controllable behavior while preserving task success. By formulating steering as feedback control and adapting neuron activations via PID and RL, we improve the steering–success trade-off across LIBERO, with RL surpassing the baseline. Limitations include task-specific RL training, partial manual neuron selection, and limited concept coverage. Future work includes task-agnostic control, automated disentanglement, and multi-concept steering.

Acknowledgment

The authors acknowledge partial support from NSF (IIS-2331908 and OAC-2530965), DARPA (HR00112420004, HR00112490420, and HR00112490424), DoE (DE-SC0024576), and NAIRR Pilot (NAIRR250261).

References

[1] KJ Åström and RM Murray. Analysis and design of feedback systems princeton, 2008. 1

[2] Karl J Astrom. Pid controllers: theory, design, and tuning. *The international society of measurement and control*, 1995. 1, 3

[3] Karl Johan Åström and Richard Murray. *Feedback systems: an introduction for scientists and engineers*. Princeton university press, 2021. 1

[4] Jose Barreiros, Andrew Beaulieu, Aditya Bhat, Rick Cory, Eric Cousineau, Hongkai Dai, Ching-Hsin Fang, Kunimatsu Hashimoto, Muhammad Zubair Irshad, Masha Itkina, et al. A careful examination of large behavior models for multitask dexterous manipulation. *arXiv preprint arXiv:2507.05331*, 2025. 1

[5] 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. 1

[6] Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. https://transformer-circuits.pub/2023/monosemantic-features/index.html. 1

[7] Jiahang Cao, Yize Huang, Hanzhong Guo, Rui Zhang, Mu Nan, Weijian Mai, Jiaxu Wang, Hao Cheng, Jingkai Sun, Gang Han, et al. Compose your policies! improving diffusion-based or flow-based robot policies via test-time distribution-level composition. *arXiv preprint arXiv:2510.01068*, 2025. 1

[8] 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. *The International Journal of Robotics Research*, 44 (10-11):1684–1704, 2025. 1

[9] Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. *arXiv e-prints*, pages arXiv–2309, 2023. 1, 2

[10] Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, 2021. 1, 2

[11] Mor Geva, Avi Caciularu, Kevin Wang, and Yoav Goldberg. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. In *Proceedings of the 2022 conference on empirical methods in natural language processing*, pages 30–45, 2022. 1

[12] Bear Häon, Kaylene Caswell Stocking, Ian Chuang, and Claire Tomlin. Mechanistic interpretability for steering vision-language-action models. In *Conference on Robot Learning*, pages 2743–2762. PMLR, 2025. 1, 2, 3, 4

[13] et al. Huo. Gigabrain-0: A world model-powered vision-language-action model. *arXiv preprint arXiv:2510.19430*, 2025. 1

[14] Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: Robot manipulation with multimodal prompts. 2023. 1

[15] Kento Kawaharazuka, Jihoon Oh, Jun Yamada, Ingmar Posner, and Yuke Zhu. Vision-language-action models for robotics: A review towards real-world applications. *IEEE Access*, 2025. 1

[16] Momin Ahmad Khan, Novak Boskov, Fatima M Anwar, and Manzoor A Khan. Controlling vision–language–action policies through sparse latent directions. In *Mechanistic Interpretability Workshop at NeurIPS 2025*. 1

[17] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024. 1

[18] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024. 1, 4

[19] Oliver Kroemer, Scott Niekum, and George Konidaris. A review of robot learning for manipulation: Challenges, representations, and algorithms. *Journal of Machine Learning Research*, 22(30):1–82, 2021. 1

[20] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023. 4

[21] Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372, 2022. 1

[22] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2(11):e7, 2017. 1

[23] Younghyo Park. Towards scalable robot learning without physical robots. Master's thesis, Massachusetts Institute of Technology, 2025. 1

[24] Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, et al. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022. 1

[25] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 1

[26] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov,

Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 1, 4

[27] Andrew Wagenmaker, Mitsuhiko Nakamoto, Yunchu Zhang, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering your diffusion policy with latent space reinforcement learning. *arXiv preprint arXiv:2506.15799*, 2025. 1

[28] Homer Rich Walke, Kevin Black, Tony Z Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, et al. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning*, pages 1723–1736. PMLR, 2023. 1

[29] Xiu Yuan, Tongzhou Mu, Stone Tao, Yunhao Fang, Mengke Zhang, and Hao Su. Policy decorator: Model-agnostic online refinement for large policy model. *arXiv preprint arXiv:2412.13630*, 2024. 1

[30] et al. Zhao. Up-vla: A unified understanding and prediction model for embodied agent. In *International Conference on Machine Learning (ICML)*, 2025. 1

[31] Zhengbang Zhu, Ziyan Li, Xiu Yuan, Hanbo Zhang, Yuxiao Liu, Chongjie Zhang, Yong Yu, Weinan Zhang, and Minghuan Liu. Unified latent steering and residual refinement for online improvement of diffusion policy models. 2025. 1

[32] Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR, 2023. 1

[33] Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, et al. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*, 2023. 1

A. RL Training without PID Initialization

Table 3. Performance of RL controllers trained without PID initialization for speed steering across all four LIBERO task suites. Compared to the RL+PID controllers used in the main paper, training RL without PID initialization results in lower task success rates (SR), demonstrating the importance of PID initialization for starting with a stable RL training.

TASK SUITE	METHOD	SAT (cm/s)	SR (%)
LIBERO LONG	RL + PID	2.535 ± 1.210	**66.50**
	RL (no PID)	2.669 ± 1.292	57.00
LIBERO GOAL	RL + PID	3.488 ± 1.388	**83.00**
	RL (no PID)	3.657 ± 1.306	78.00
LIBERO OBJECT	RL + PID	2.171 ± 0.466	**76.50**
	RL (no PID)	2.124 ± 0.434	72.00
LIBERO SPATIAL	RL + PID	3.606 ± 0.820	**78.50**
	RL (no PID)	3.606 ± 0.759	77.00

In this section, we provide detailed results across all LIBERO task suites to support our claim in the main paper on "degraded task success and steering performance for RL training initialized without PID data".

In our default training setup, the RL controller is initialized using steering coefficients generated by the PID controller, which provides a stable starting point for learning. To evaluate the importance of this initialization, we also trained RL policies from random initialization, without using PID-generated trajectories.

Tab. 3 reports the detailed results for speed steering on all four LIBERO task suites: Long, Goal, Object, and Spatial. Across all suites, removing PID initialization consistently leads to lower task success rates while providing no improvement in steering metrics. These results support the conclusion in the main paper that PID initialization stabilizes RL training and improves the trade-off between concept steering and task completion.

B. Time Complexity Analysis

Tab. 4 shows the average time complexity and peak GPU usage for different approaches. CTRL-STEER imposes low overhead on both time complexity and GPU usage, making it a suitable approach over retraining or finetuning the model.

Fig. 4 and Fig. 5 shows the breakdown of the time overhead in controlled steering. The results show that the major computational cost comes from VLA forward pass, and the overhead created by the controller is negligible.

Table 4. Average Inference Cost for All Approaches (No steering, Static steering with $C = 5, 10, 20$, and CTRL-STEER with PID and RL)

Task	time / step (s)	Peak GPU (GB)
OpenVLA	0.1869 ± 0.0039	14.2587 ± 0.0003
C=5	0.1961 ± 0.0124	14.2587 ± 0.0003
C=10	0.1967 ± 0.0079	14.2587 ± 0.0003
C=20	0.1974 ± 0.0060	14.2587 ± 0.0003
CTRL-STEER (PID)	0.2021 ± 0.0076	14.2587 ± 0.0003
CTRL-STEER (RL)	0.2094 ± 0.0077	14.2588 ± 0.0003

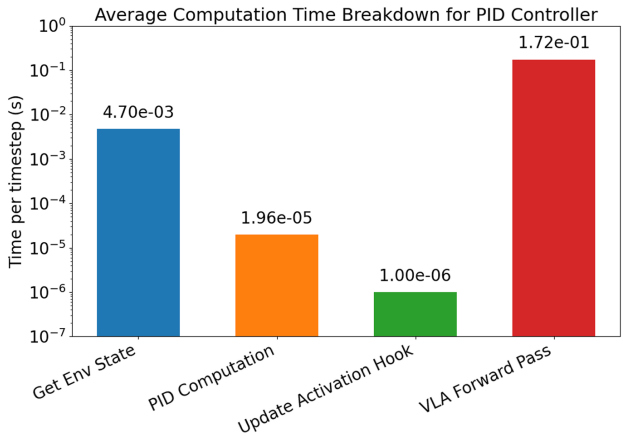

Figure 4. Breakdown of the **average** per-timestep computational cost for CTRL-STEER with only PID. The VLA forward pass dominates the runtime in both cases, while the additional computations required for controlled steering contribute a very low overhead.

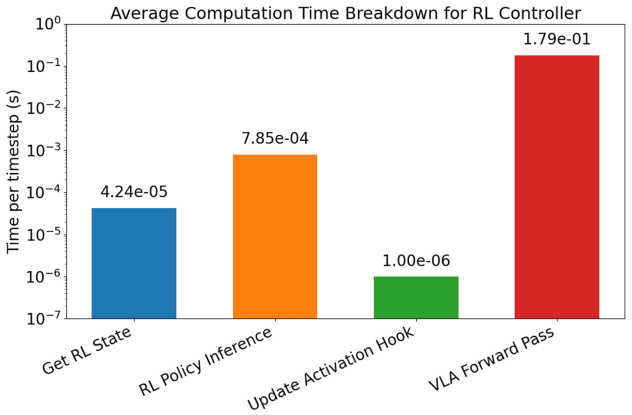

Figure 5. Breakdown of the **average** per-timestep computational cost for CTRL-STEER with RL+PID. The VLA forward pass dominates the runtime in both cases, while the additional computations required for controlled steering contribute a very low overhead.