S^3-VLA: VISUAL SELF-GROUNDING FOR VISION-LANGUAGE-ACTION POLICIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-language-action (VLA) policies have shown strong promise as generalist controllers for robot manipulation, yet their action predictions can remain fragile under changes in camera extrinsics. We identify a systematic failure pattern in which camera perturbations induce directional drift in predicted robot trajectories, revealing that current VLAs may exploit *image-space shortcuts*, such as pixel-coordinate correlations, rather than grounding actions in robot–object geometry. To bridge this gap, we introduce S^3-VLA, a Self-Supervised alignment approach to incentivize *visual Self-grounding* capability in VLA policies. Our central insight is that robot motion provides intrinsic spatial supervision for localizing the robot itself. To implicitly guide the policy to attend to the robot, we formulate an auxiliary grounding objective to recover motion-derived localization targets from its visual representations. Across simulation benchmarks and real-world manipulation tasks, S^3-VLA consistently improves average task success on two leading VLA backbones, $\pi_{0.5}$ and GR00T-N1.6. Under camera-extrinsic perturbations, S^3-VLA increases the average success rate on real-world tasks from 16.4% to 29.4%. These findings highlight the importance of *visual self-grounding* for robust vision-language-action policies.

1 INTRODUCTION

Vision-language-action (VLA) policies have emerged as a foundation for generalist robot manipulation, coupling the semantic priors of vision-language models with action learning from robot experience (Kim et al., 2025; Black et al., 2025b; Bjorck et al., 2025). Recent work broadens this foundation through multi-embodiment training and shared action representations, enabling knowledge to transfer across robots with different morphologies and control spaces (Gemini Robotics Team et al., 2025; Luo et al., 2026; Yang et al., 2026). Generalization now extends beyond robot platforms to manipulation in previously unseen environments (Black et al., 2025a), while richer multimodal task context enables learned skills to be recombined for novel tasks (Physical Intelligence, 2026).

Despite this progress, we identify a systematic *image-space shortcut* in current VLAs. When perturbing camera viewpoints under identical physical scenes and tasks, predicted robot trajectories drift directly with camera-induced pixel displacements, leading to severe execution failures during grasping or placement (Figure 1). Rather than grounding these predictions in robot-centric relations, the policy ties them to camera-specific coordinates. When probing the latent VLM representations, we further find that the model often fails to localize the robot. This suggests a critical missing capability we term *visual self-grounding*: the ability to localize the robot itself in visual observations.

Existing methods strengthen visual perception, yet their objectives still admit *image-space shortcuts*. Geometric encoding and feature alignment (Qu et al., 2025; Li et al., 2026b) enrich scene representations, while modality rebalancing and state-aware learning (Lu et al., 2026; Wang et al., 2026; Kim et al., 2026) promote visual reliance. These objectives leave the correspondence between the robot's physical state and its visible body implicit. Under consistent training views, image-coordinate correlations can explain demonstrated actions without recovering robot–object geometry, leaving imitation accuracy insufficient to establish *visual self-grounding*.

To address this gap, we introduce S^3-VLA, a Self-Supervised alignment approach that incentivizes *visual Self-grounding* in VLA policies. Our central insight is that robot motion provides intrinsic

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: Camera-only perturbations reveal *image-space shortcut* failures while robot–object geometry and physical objective remain fixed. The detailed setup is summarized in Appendix A.

spatial supervision for localizing the robot itself. Unlike kinematic projection (Li et al., 2025c), this supervision requires no camera calibration, making it applicable to uncalibrated real-world demonstrations. We therefore mine robot localization targets from optical flow and frame differences in demonstration videos. To implicitly guide the policy to attend to the robot, we introduce *Self-Grounding Alignment* (SGA), an auxiliary objective that trains a lightweight grounding head to recover these targets from the VLM's visual tokens. The head uses learned queries to predict a grounding distribution over these tokens. Aligning this distribution with the targets complements action learning, encouraging the shared representation to preserve robot localization cues.

We evaluate S^3-VLA on SimplerEnv and Meta-World, and in real-world Franka manipulation under original and perturbed camera views. Across these settings, it consistently improves average task success on two leading VLA backbones, $\pi_{0.5}$ and GR00T-N1.6. It also yields relevance maps that focus more consistently on the robot's visible body. On the real robot, the task success rate under the original camera configuration increases from 44.2% to 53.3%. Under camera-extrinsic perturbations, mean success across the twelve perturbed conditions increases from 16.4% to 29.4%. These results suggest that *visual self-grounding* can make VLA policies more robust to camera shifts.

Our contributions are summarized as follows:

- We identify a systematic failure pattern in VLA policies, showing that current VLAs can rely on *image-space shortcuts* rather than robot–object geometry. We further attribute this behavior to weak *visual self-grounding* capability of VLA policies.
- We introduce S^3-VLA, a self-supervised alignment approach that incentivizes *visual self-grounding* for VLA policies, based on the insight that robot motion provides intrinsic spatial supervision for localizing the robot itself.
- We conduct extensive experiments on SimplerEnv, Meta-World, and real robots. S^3-VLA consistently improves average task success on two leading VLA backbones, GR00T-N1.6 and $\pi_{0.5}$. Under camera-extrinsic perturbations, mean real-world success increases from 16.4% to 29.4%.

2 Related Work

Spatially Grounded VLA Policies. Vision-language-action (VLA) policies transfer vision-language pretraining to robot control, as established by RT-2 and OpenVLA (Zitkovich et al., 2023; Kim et al., 2025). Recent models pursue broader generalization through heterogeneous co-training in $\pi_{0.5}$ (Black et al., 2025a) and multimodal context conditioning in $\pi_{0.7}$ (Physical Intelligence, 2026), while GR00T-N1.7 and Gemini Robotics On-Device 2 support control across robot embodiments (NVIDIA, 2026; Google DeepMind, 2026). Alongside these generalist models, spatial approaches encode 3D scene structure (Zhen et al., 2024; Qu et al., 2025; Li et al., 2026a; Sun et al., 2025) and use projected or synthesized views for manipulation (Li et al., 2025a; Gu et al., 2026). Other methods express motion through visual trajectory prompts (Zheng et al., 2025) or camera-space actions (Zhang et al., 2026).



Figure 2: Self-grounding probes on real-world and simulation scenes. Task prompts tend to concentrate relevance on the manipulated objects, whereas the self-grounding prompt, "Where is the robot itself", does not reliably concentrate relevance on the robot's visible body.

Visual Self-Grounding. Proprioception provides precise state information, yet can allow a policy to predict actions with limited dependence on vision. State-free Policy omits proprioceptive inputs and predicts relative end-effector actions from wrist-camera observations (Zhao et al., 2025). GAP scales down proprioceptive gradients according to estimated motion-transition probabilities (Lu et al., 2026), while ReViP modulates policy features using visually inferred task stages (Li et al., 2026c). ThinkProprio uses robot state and language instructions to select visual tokens before VLM processing (Wang et al., 2026). Proprioceptive inputs specify physical state, while *visual self-grounding* requires localizing the robot itself in visual observations.

Representation Learning for VLA Policies. Beyond input conditioning, auxiliary objectives directly supervise visual representations through reconstruction, feature alignment, or contrastive learning. ROSS reconstructs image latents to retain visual detail (Wang et al., 2025), while REPA and 3DRS align model features with pretrained 2D and 3D representations, respectively (Yu et al., 2025; Huang et al., 2025). This principle extends to robot policies through Spatial Forcing, which aligns VLA visual embeddings with pretrained 3D features (Li et al., 2026b), and RS-CL, which uses distances between robot states as contrastive supervision (Kim et al., 2026). Our approach derives robot localization targets from visual motion in demonstrations, using them to supervise VLA visual tokens through a training-only grounding loss.

3 Does a VLA Know Where It Is?

Under a fixed training viewpoint, demonstrated actions can be predicted without localizing the robot in visual observations. We hypothesize that current VLAs exploit this freedom and bind actions to camera-specific image coordinates. We test this hypothesis behaviorally with camera-extrinsic perturbations in Section 3.1 and representationally with self-grounding probes in Section 3.2.

3.1 VLAs Can Rely on Image-Space Shortcuts

As shown in Figure 1, we shift the wrist or frontal camera of a GR00T-N1.6 policy under the same physical scene and task. Although robot–object geometry is unchanged, predicted robot trajectories drift with camera-induced pixel displacements: horizontal and vertical shifts bias grasping or placement along the corresponding axis, so the gripper misses the cucumber or the cucumber misses the bowl. Table 6 summarizes these failures. Across four real-world tasks, such perturbations reduce the success of GR00T-N1.6 from 44.2% to 16.4%, as shown in Figure 5.

We refer to this behavior as an *image-space shortcut*: the policy maps object pixel locations to robot-frame actions through camera-specific correlations, implicitly absorbing the training camera extrinsics. An alternative that does not hinge on fixed pixel coordinates is to use the robot's visible body as a visual reference, relating objects to the robot within the same image, the principle that makes image-based visual servoing less sensitive to camera calibration (Hutchinson et al., 1996). This alternative, however, requires the policy to know where the robot itself appears.

3.2 Probing Visual Self-Grounding in VLAs

We hypothesize that the *image-space shortcut* is partly due to weak *visual self-grounding*, i.e., the ability to localize the robot itself in visual observations. We probe this capability with prompt-conditioned relevance maps over the VLM's visual tokens, computed as detailed in Appendix E.1 and shown in Figure 2. With task prompts, relevance tends to concentrate on the manipulated objects,

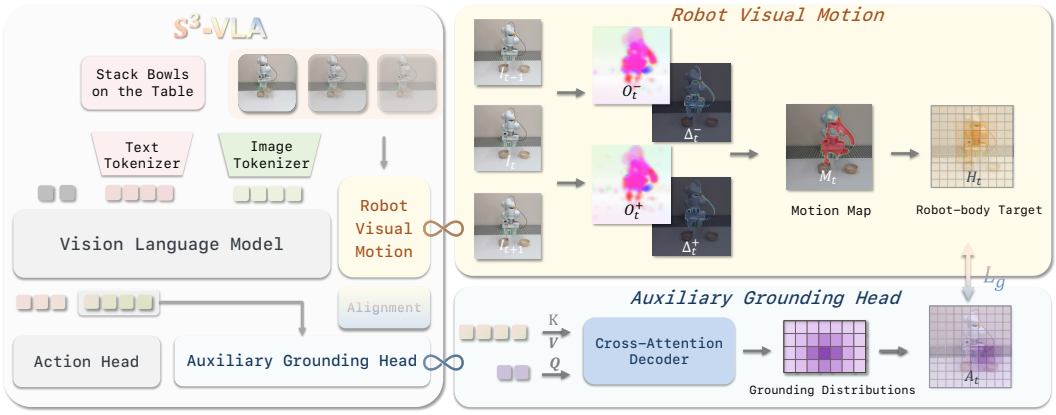


Figure 3: Overview of S^3-VLA. We mine token-level robot localization targets from *Robot Visual Motion* for *Self-Grounding Alignment*, which trains an auxiliary grounding head to recover these targets from the VLM's visual tokens. This auxiliary objective encourages the shared visual representation to preserve robot localization cues during action learning.

whereas with the self-grounding prompt, "Where is the robot itself", it does not reliably concentrate on the robot's visible body. To assess how well these features support robot localization, we train linear probes to predict gripper position, as detailed in Section 5.4. On held-out tasks with shifted camera views, only 61.5% of predictions are accurate to within 5% of the image diagonal.

If the visual tokens carry weak robot localization cues, camera-specific correlations remain an easy way to fit consistent training views. Since the imitation objective does not require self-grounding, these observations motivate explicit supervision of this capability. In Section 4, we introduce S^3-VLA to provide such supervision through robot localization targets that require no camera calibration and are used only during training.

4 S^3-VLA

We propose S^3-VLA, a self-supervised alignment approach that incentivizes *visual self-grounding* capability in VLA policies. As shown in Figure 3, S^3-VLA builds on two core ingredients: *Robot Visual Motion*, which provides intrinsic spatial supervision through token-level robot localization targets mined from demonstrations, and *Self-Grounding Alignment* (SGA), an auxiliary objective that trains a lightweight grounding head to recover these targets from the VLM's visual tokens.

4.1 Preliminaries

A VLA policy predicts actions conditioned on an image observation I_t, a language instruction x, and a proprioceptive state s_t. It is trained on demonstrated actions a_t with a backbone-specific objective $\mathcal{L}_{\text{act}}$. S^3-VLA adds auxiliary supervision to the visual tokens used for action prediction.

For the target camera view, let $\mathcal{P}_t$ be the set of valid image tokens after view-specific padding or masking. From the VLM's final output sequence of visual and language tokens, we select the hidden states at image-token positions for this view, yielding visual features $V_t = \{v_t(p) \in \mathbb{R}^d\}_{p \in \mathcal{P}_t}$. Here $u_p \in [0,1]^2$ is the normalized image center of token p. To express robot localization at this token resolution, we use a categorical distribution over $\mathcal{P}_t$ whose probability mass indicates the location of the robot's visible body.

4.2 Mining Robot Visual Motion

To obtain self-grounding supervision from demonstrations, we exploit *Robot Visual Motion* (RVM), the image motion induced by the robot's visible body. We mine this signal offline from front-view videos recorded with a mostly static camera, using visual motion to construct robot localization targets and proprioceptive activity to assess their reliability.

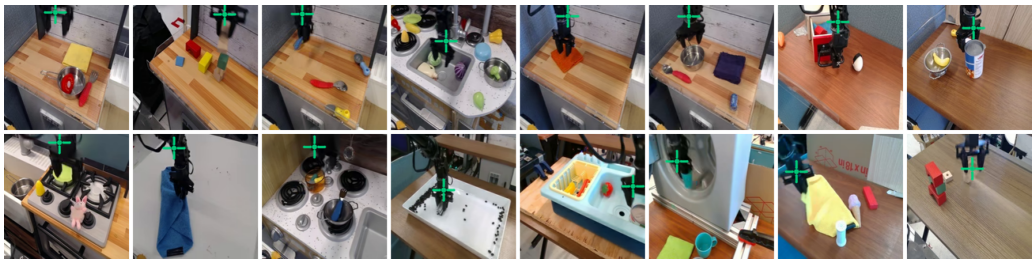


Figure 4: Point pseudo-labels mined from the *Robot Visual Motion* signal for robot localization. Green crosses mark the extracted peak locations.

Motion map construction. We combine appearance changes and motion magnitudes from adjacent frames to construct a dense motion map:

$$M_t(\xi) = \mathcal{N}\big(\lambda_\Delta\big(\Delta_t^-(\xi) + \Delta_t^+(\xi)\big) + \lambda_f\big(O_t^-(\xi) + O_t^+(\xi)\big)\big)\,, \tag{1}$$

where ξ denotes an image pixel, $\Delta_t^\pm$ are grayscale differences with the previous and next frames, and $O_t^\pm$ are the corresponding Farnebäck optical-flow magnitudes (Farnebäck, 2003). The operator $\mathcal{N}$ applies robust normalization followed by light Gaussian smoothing.

Localization target construction. From M_t, we extract a motion peak z_t and estimate its spatial uncertainty from the surrounding high-score region. These estimates define an anisotropic Gaussian on the VLA token grid. When the local support box contains valid tokens, we clip the Gaussian to this box before normalization. Let $\tilde{G}_t(p)$ denote the score at token p. The soft target is

$$H_t(p) = \frac{\tilde{G}_t(p)}{\sum_{q\in\mathcal{P}_t}\tilde{G}_t(q)}. \tag{2}$$

Figure 4 visualizes the point pseudo-labels z_t, while training uses the distributions H_t to account for spatial uncertainty around the mined locations.

Proprioception-guided confidence filtering. We assign each target a confidence score c_t, computed as the geometric mean of three normalized cues: peak confidence, motion-energy confidence, and robot activity. The activity term is derived from end-effector displacement and gripper-state change. Targets with $c_t < \tau_c$ are excluded from the grounding loss. Appendix B.1 details target construction, confidence computation, and parameters.

4.3 Self-Grounding Alignment

To incorporate this spatial supervision into action learning, we introduce *Self-Grounding Alignment* (SGA). A lightweight grounding head learns to recover the mined targets H_t from the VLM's visual tokens by aligning its predicted distribution A_t with H_t. The resulting grounding loss $\mathcal{L}_g$ is optimized jointly with the action objective to encourage robot localization cues in shared visual features.

Grounding head. We add a positional embedding to each extracted visual token, $\hat{v}_t(p) = v_t(p) + e(u_p)$. Following the learned-query design of transformer detectors (Carion et al., 2020), the grounding head uses a shallow cross-attention decoder in which M learned queries attend to the visual tokens $\hat{v}_t(p)$. For each query, we obtain a spatial distribution by applying a softmax to the cosine similarities between its projected feature $q_{t,m}$ and the token keys $k_t(p)$, both L2-normalized:

$$a_{t,m}(p) = \mathrm{softmax}_{p\in\mathcal{P}_t}\left(s\, q_{t,m}^\top k_t(p)\right), \qquad A_t(p) = \frac{1}{M}\sum_{m=1}^{M} a_{t,m}(p). \tag{3}$$

Here s controls the logit scale, and averaging the per-query distributions yields the grounding prediction A_t.

Grounding loss. The grounding loss aligns the predicted distribution A_t with the motion-derived localization target H_t. Let $U_t(p) = 1/|\mathcal{P}_t|$ be the uniform distribution over valid visual tokens. For informative targets, the normalized per-frame grounding loss and its confidence-weighted average over minibatch frames $\mathcal{B}$ are

$$\ell_t^{\mathrm{g}} = \frac{D_{\mathrm{KL}}(H_t \| A_t)}{D_{\mathrm{KL}}(H_t \| U_t)}, \qquad m_t = \mathbf{1}[c_t \geq \tau_c], \\ \mathcal{L}_{\mathrm{g}} = \frac{\sum_{t \in \mathcal{B}} m_t c_t \ell_t^{\mathrm{g}}}{\max(\sum_{t \in \mathcal{B}} m_t c_t, \varepsilon)}. \tag{4}$$

The full training objective then adds this auxiliary term to the original action loss:

$$\mathcal{L} = \mathcal{L}_{\mathrm{act}} + \lambda_g \mathcal{L}_{\mathrm{g}}. \tag{5}$$

The denominator in ℓ_t^{g} rescales the KL divergence by the target's information content relative to a uniform distribution. For informative targets, the loss equals zero when $A_t = H_t$ and one when $A_t = U_t$. Appendix B.2 details the parameterization and numerical safeguards, including ε. The grounding head and the mined targets are used only during training. At inference, the policy uses the same observations, action interface, and inference path as the original VLA.

5 Experiments

We organize the experiments around four questions. *Q1*: Does *visual self-grounding* improve manipulation under the training view? *Q2*: Does it reduce reliance on the *image-space shortcut* under camera perturbations? *Q3*: Does the policy learn to localize the robot, and does this ability transfer across tasks? *Q4*: Do the gains come from supervising the robot's image location?

5.1 Experimental Setup

We evaluate GR00T-N1.6 on seven WidowX tasks from Bridge and six Google Robot tasks from Fractal in SimplerEnv (Li et al., 2025b), and $\pi_{0.5}$ on Meta-World MT50 (Yu et al., 2020). Real-world experiments use a Franka Research 3 robot on four tasks (Figure 10 in Appendix C.2.3), evaluated under the training view and under the Wrist, Front-H, and Front-V camera perturbations. S^3-VLA adds the SGA objective only during fine-tuning, so the backbone, action objective, and inference path remain unchanged. In simulation, we compare against the official action-tuned checkpoint of each backbone. On the real robot, both the baseline and S^3-VLA are fine-tuned from the same GR00T-N1.6 checkpoint for one epoch on 200 trajectories per task. We run 200 rollouts per SimplerEnv task or camera condition, 10 trials per Meta-World task for each of three evaluation seeds, and 30 trials per real-robot task and camera condition. Appendix C provides further details.

5.2 Does Self-Grounding Help under the Training View?

We first address *Q1*. Table 1 shows that S^3-VLA improves success even under the training view, raising mean SimplerEnv success from 60.1% to 74.1% on Bridge and from 71.7% to 76.1% on

Table 1: **Task success rates (%) on SimplerEnv with GR00T-N1.6.** 200 rollouts per task; Avg. is the mean across tasks within each embodiment. Colored values show changes relative to GR00T-N1.6.

Method	Bridge (WidowX)							Avg. success ↑
	Spoon on Towel	Carrot on Plate	Stack Cube	Eggplant in Basket	Eggplant in Sink	Open Drawer	Close Drawer	
GR00T-N1.6	75.5	70.5	4.5	46.5	29.0	95.5	99.5	60.1
S^3-VLA (Ours)	90.5 +15.0	66.5 -4.0	8.0 +3.5	89.0 +42.5	69.0 +40.0	97.5 +2.0	98.5 -1.0	**74.1** +14.0

Method	Fractal (Google Robot)						Avg. success ↑
	Pick Coke Can	Move Near	Open Drawer	Close Drawer	Place in Closed Drawer	Pick Object	
GR00T-N1.6	100.0	98.0	49.0	77.5	10.5	95.0	71.7
S^3-VLA (Ours)	100.0 0.0	96.0 -2.0	67.5 +18.5	80.5 +3.0	16.0 +5.5	96.5 +1.5	**76.1** +4.4

Fractal. Success improves substantially on both eggplant tasks and Fractal Open Drawer, with a modest decline on Carrot on Plate.

S^3-VLA improves task success across VLA backbones. With $\pi_{0.5}$, it raises average success on Meta-World MT50 from 52.8% to 61.7% and improves all four difficulty groups, as shown in Table 2. The gains are largest on Easy and Medium tasks, at +10.0% and +12.8%, and shrink to +3.3% and +0.7% on Hard and Very Hard tasks, which remain challenging for both policies.

Table 2: Meta-World MT50 mean success (%) with $\pi_{0.5}$ over three evaluation seeds. Avg. is the mean across all 50 tasks. V.H. denotes Very Hard.

Method	Easy	Med.	Hard	V.H.	Avg.
$\pi_{0.5}$	65.0	42.7	31.7	32.0	52.8
S^3-VLA	**75.0**	**55.5**	**35.0**	**32.7**	**61.7**
Δ	+10.0	+12.8	+3.3	+0.7	+8.9

5.3 Does Self-Grounding Help under Camera Perturbations?

For *Q2*, we test whether *visual self-grounding* improves robustness to camera shifts. This addresses the *image-space shortcut* observed in Section 3.1, where predicted trajectories drift with camera-induced pixel displacements. We vary only the camera extrinsics on WidowX *Eggplant in sink* and Google Robot *Pick object*. As shown in Table 3, S^3-VLA improves success in 13 of the 14 robot–condition pairs, with WidowX yaw as the only exception, and raises average success from 18.1% to 30.7% on WidowX and from 78.9% to 82.4% on Google Robot. Our adaptation of Spatial Forcing (Li et al., 2026b), which aligns visual features with a 3D teacher, reaches only 19.7% on WidowX, as detailed in Appendix D.4. This gap suggests that enriching scene geometry leaves the robot's correspondence to its visible body implicit, whereas S^3-VLA supervises it directly.

Table 3: Task success (%) with GR00T-N1.6 in SimplerEnv under camera perturbations. S^3-VLA results are averaged over three fine-tuning seeds. Colored values show changes relative to GR00T-N1.6. Appendix C.2.2 gives the camera transformations and per-condition standard deviations.

Robot	Method	Yaw	Pitch	Roll	Forward	Right	Up	Composite	Avg.
WidowX	GR00T-N1.6	**14.5**	2.0	7.0	48.0	24.0	4.5	27.0	18.1
	S^3-VLA	12.2 -2.3	**4.3** +2.3	**31.3** +24.3	**64.5** +16.5	**44.2** +20.2	**22.8** +18.3	**35.7** +8.7	**30.7** +12.6
Google Robot	GR00T-N1.6	71.0	55.5	93.0	91.0	93.0	92.5	56.5	78.9
	S^3-VLA	**73.3** +2.3	**59.0** +3.5	**93.3** +0.3	**93.8** +2.8	**97.2** +4.2	**94.7** +2.2	**65.3** +8.8	**82.4** +3.5

The same trend holds on the real robot. Figure 5 shows that S^3-VLA raises success from 44.2% to 53.3% under the training view and from 16.4% to 29.4% under camera perturbations, improving 11 of the 12 perturbed conditions. Appendix D.1 reports the trial counts. Under perturbation, the baseline retains 37% of its training-view success, whereas S^3-VLA retains 55%. This result is consistent with the intuition from image-based visual servoing in Section 3.1. A policy that can locate its own body in the image depends less on the pixel-to-action mapping of the training view.

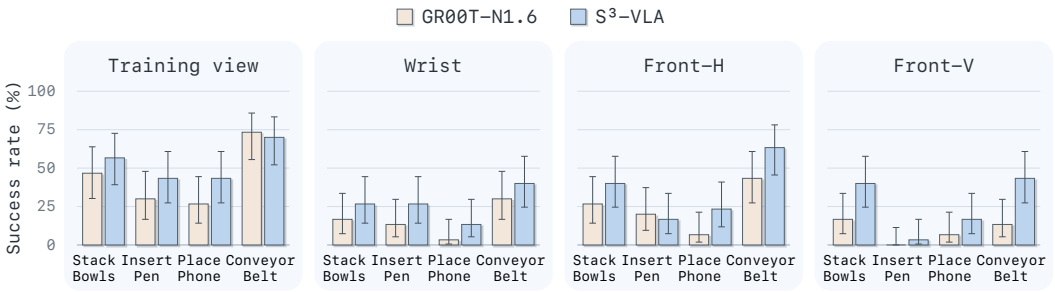


Figure 5: **Real-world success under camera perturbations.** Both methods fine-tune GR00T-N1.6 on the same task data. Each bar uses 30 trials; Wrist, Front-H, and Front-V average two directions with 15 trials each. Error bars are 95% Wilson intervals. Figure 1 and Appendix C.2.3 show the six perturbation setups.

5.4 Does the Policy Learn Where the Robot Is?

Higher task success alone does not show that the policy localizes the robot. To answer *Q3*, we fit linear probes of equal capacity to each policy's frozen visual features at the action-head input, using

simulator-rendered robot and gripper masks from four WidowX tasks. We then evaluate the probes under camera perturbations on new trajectories from seven tasks, including three held out from probe fitting to assess cross-task transfer. Appendix E.2 describes the protocol.

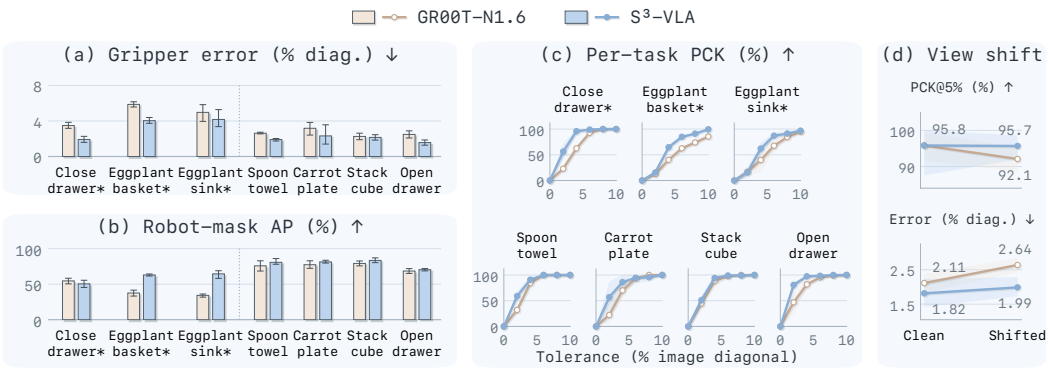


Figure 6: **Robot localization from frozen GR00T-N1.6 visual features.** (a) Gripper-center error (% of image diagonal), (b) robot-mask AP, and (c) the fraction of gripper predictions within each threshold (PCK), averaged over four camera perturbations. ∗ marks tasks excluded from probe fitting. (d) Matched states across training and perturbed views on the four fitting tasks. Error bars and shading show 95% initial-scene bootstrap intervals.

S³-VLA localizes the robot more accurately. This advantage holds in simulation and on the real robot, both quantitatively and qualitatively. In simulation, S³-VLA reduces gripper error on all seven tasks by 27% on average and improves robot-mask AP on six, as shown in Figure 6a and b. The advantage is most evident under camera perturbations. Figure 6d shows that both policies reach 95.8% PCK@5% under the training view of the fitting tasks, where memorizing the robot's typical image position suffices. Once the camera moves, however, the baseline's gripper error grows from 2.11% to 2.64% of the image diagonal, three times the increase for S³-VLA. On the real robot, where simulator masks are unavailable, we apply the relevance probe from Section 3.2. Given the prompt "Where is the robot itself", the baseline spreads relevance over the table and objects, whereas S³-VLA concentrates it on the arm and gripper, as shown in Figure 7. S³-VLA thus mitigates the weak self-grounding diagnosed in Section 3.2 on both simulated and real observations.

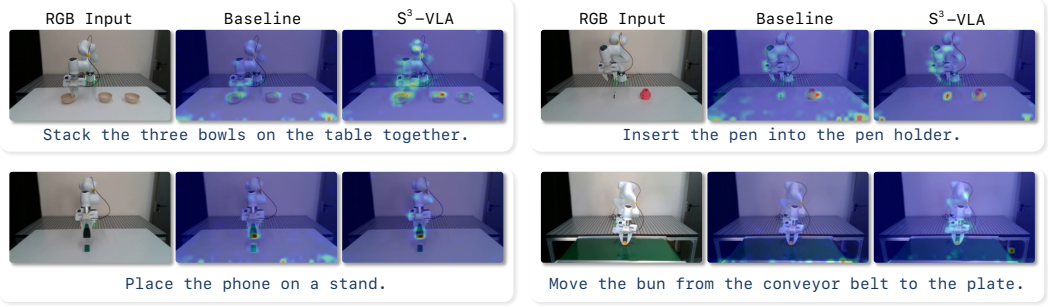


Figure 7: **Real-robot relevance maps** from GR00T-N1.6 policies for the prompt "Where is the robot itself". Appendix E.1 describes how the maps are computed from self-attention and normalized for display.

***Visual self-grounding* transfers across tasks.** Figure 6c shows that the advantage of S³-VLA extends to the three tasks held out from probe fitting, where its PCK curves lie above those of the baseline at nearly all error thresholds. On these tasks, PCK@5% rises from the 61.5% reported in Section 3.2 to 84.3%, whereas action-only fine-tuning without $\mathcal{L}_g$ reaches only 66.3%. A probe fitted on four tasks therefore transfers far better to unseen tasks when the features are trained with SGA. S³-VLA learns robot localization as a task-agnostic capability rather than a task-specific cue, consistent with the design of RVM, whose supervision comes from the robot's own motion and is available in every demonstration regardless of the task.

5.5 What Drives the Gain?

Table 4: Task success (%) in SimplerEnv ablations. Overall weights all 13 tasks equally; colored values show changes relative to the full model.

Method	Bridge	Fractal	Overall
S^3-VLA	**74.1**	**76.1**	**75.0**
w/o image diff $\Delta_t^{\pm}$	64.9	70.7	67.6 -7.4
w/o optical flow $O_t^{\pm}$	63.6	68.2	65.8 -9.2
w/o $\mathcal{L}_{\mathrm{g}}$	63.4	69.8	66.3 -8.7
w/o normalized KL	66.5	70.9	68.5 -6.5

Table 5: Task success (%, mean over three seeds) on *Eggplant in sink* and *Pick object*. The camera perturbations column averages seven conditions.

Method	Training view	Camera perturbations
S^3-VLA	**82.7**	**56.5**
Predict s_t	63.5	48.1
Predict Δs_t	71.0	50.2
w/o $\mathcal{L}_{\mathrm{g}}$	72.5	49.8

Ablations. To answer *Q4*, Table 4 removes or replaces each component of SGA. Removing the grounding loss $\mathcal{L}_{\mathrm{g}}$ lowers overall success from 75.0% to 66.3%, close to the 65.5% of GR00T-N1.6, so action-only fine-tuning does not account for the gain. The construction of the supervision matters as well. Removing either motion cue or replacing the RVM targets and normalized KL loss with the Gaussian cross-entropy variant (Appendix E.4) lowers overall success. Each variant loses most of the gain from SGA, indicating that both the RVM targets and the normalized KL formulation contribute.

Robot-state prediction does not substitute for *visual self-grounding*. State-aware objectives supervise the robot's physical state but leave its correspondence to the visible body implicit. To test whether such supervision suffices, we replace SGA with prediction of the robot state s_t or its change Δs_t, following Appendix D.3. Table 5 shows that neither variant clearly improves over action-only fine-tuning, and predicting s_t even lowers training-view success from 72.5% to 63.5%. These controls suggest that the useful additional supervision is the robot's location in the image, which state prediction does not explicitly provide.

RVM targets primarily identify the robot. A remaining concern is that SGA helps by supervising task objects or other regions with salient motion. In particular, a grasped object moving with the gripper could attract the peaks z_t used to construct RVM targets H_t. To examine this possibility, we use simulator actor masks to locate the retained target points across seven WidowX tasks. Figure 8a shows that 82.0% lie on the robot and only 0.3% on the manipulated object. We then restrict the analysis to grasp frames from five tasks, where robot and object motion are coupled. Figure 8b shows that 75.1% of the retained points still lie on the robot and only 2.0% on the grasped object. RVM thus mainly supervises the robot's image location, even during grasping. Together with the ablations, these results suggest that learning to locate the robot in the image contributes to the gains.

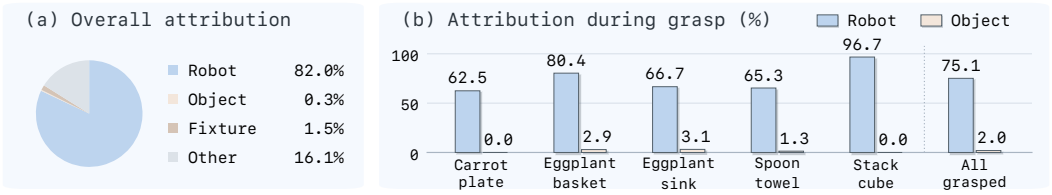


Figure 8: **RVM target points predominantly lie on the robot.** (a) Attribution of target points across seven tasks. (b) Attribution restricted to frames in which the robot grasps an object. Each bar pair reports the percentages of points on the robot and on the grasped object. The final pair aggregates all five tasks, with 75.1% of points on the robot and 2.0% on the object. Appendix E.3 describes the actor-mask audit and grasp-frame selection.

6 Conclusion

This work studies *visual self-grounding* in VLA policies, showing that current policies can rely on *image-space shortcuts* under camera shifts instead of stable robot–object geometry. S^3-VLA mines robot localization targets from visual motion and recovers them from VLA visual tokens through Self-Grounding Alignment with a lightweight grounding head, without manual masks or external grounding data. The approach encourages robot localization while preserving the original action interface, offering a simple way to improve the robustness of VLA policies to camera shifts.

AI USE STATEMENT

We used generative AI tools to assist with manuscript writing and editing, including improvements to wording, grammar, clarity, and organization, as well as LaTeX formatting. The authors reviewed and revised the AI-assisted content to ensure that it accurately reflects the methods and results. The authors take full responsibility for the accuracy and integrity of the final manuscript.

REFERENCES

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. GR00T N1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In *Proceedings of the 9th Conference on Robot Learning*, volume 305, pp. 17–40. PMLR, 2025a.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert 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. In *Proceedings of Robotics: Science and Systems*, 2025b.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pp. 213–229. Springer, 2020.

Gunnar Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Scandinavian Conference on Image Analysis*, pp. 363–370. Springer, 2003.

Gemini Robotics Team, Abbas Abdolmaleki, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Ashwin Balakrishna, Nathan Batchelor, Alex Bewley, Jeff Bingham, et al. Gemini Robotics 1.5: Pushing the frontier of generalist robots with advanced embodied reasoning, thinking, and motion transfer. *arXiv preprint arXiv:2510.03342*, 2025.

Google DeepMind. Gemini Robotics On-Device 2. Model card, 2026. URL https://deepmind.google/models/model-cards/gemini-robotics-on-device-2/.

Songen Gu, Yuhang Zheng, Weize Li, Yupeng Zheng, Yating Feng, Xiang Li, Yilun Chen, Pengfei Li, and Wenchao Ding. VistaBot: View-robust robot manipulation via spatiotemporal-aware view synthesis. *arXiv preprint arXiv:2604.21914*, 2026.

Xiaohu Huang, Jingjing Wu, Qunyi Xie, and Kai Han. 3DRS: MLLMs need 3D-aware representation supervision for scene understanding. In *Advances in Neural Information Processing Systems*, volume 38, pp. 67961–67988. Curran Associates, Inc., 2025.

Seth Hutchinson, Gregory D. Hager, and Peter I. Corke. A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, 12(5):651–670, 1996.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, et al. OpenVLA: An open-source vision-language-action model. In *Proceedings of the 8th Conference on Robot Learning*, volume 270, pp. 2679–2713. PMLR, 2025.

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. In *International Conference on Machine Learning*, 2026.

Chengmeng Li, Junjie Wen, Yaxin Peng, Yan Peng, and Yichen Zhu. PointVLA: Injecting the 3D world into vision-language-action models. *IEEE Robotics and Automation Letters*, 11(3): 2506–2513, 2026a.

Fuhao Li, Wenxuan Song, Han Zhao, Jingbo Wang, Pengxiang Ding, Donglin Wang, Long Zeng, and Haoang Li. Spatial Forcing: Implicit spatial representation alignment for vision-language-action model. In *International Conference on Learning Representations*, 2026b.

Peiyan Li, Yixiang Chen, Hongtao Wu, Xiao Ma, Xiangnan Wu, Yan Huang, Liang Wang, Tao Kong, and Tieniu Tan. BridgeVLA: Input-output alignment for efficient 3D manipulation learning with vision-language models. In *Advances in Neural Information Processing Systems*, volume 38, pp. 63635–63673. Curran Associates, Inc., 2025a.

Xuanlin Li, Kyle Hsu, Jiayuan Gu, Oier Mees, Karl Pertsch, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, et al. Evaluating real-world robot manipulation policies in simulation. In *Proceedings of the 8th Conference on Robot Learning*, volume 270, pp. 3705–3728. PMLR, 2025b.

Yi Li, Yuquan Deng, Jesse Zhang, Joel Jang, Marius Memmel, Caelan Garrett, Fabio Ramos, Dieter Fox, Anqi Li, Abhishek Gupta, and Ankit Goyal. HAMSTER: Hierarchical action models for open-world robot manipulation. In *International Conference on Learning Representations*, 2025c.

Zhuohao Li, Yinghao Li, Jian-Jian Jiang, Lang Zhou, Tianyu Zhang, and Wei-Shi Zheng. ReViP: Reducing false completion in vision-language-action models with vision-proprioception rebalance. *arXiv preprint arXiv:2601.16667*, 2026c.

Jingxian Lu, Wenke Xia, Yuxuan Wu, Zhiwu Lu, and Di Hu. When would vision-proprioception policies fail in robotic manipulation? In *International Conference on Learning Representations*, 2026.

Hao Luo, Ye Wang, Wanpeng Zhang, Sipeng Zheng, Ziheng Xi, Chaoyi Xu, Haiweng Xu, Haoqi Yuan, Chi Zhang, Yiqing Wang, Yicheng Feng, and Zongqing Lu. Being-H0.5: Scaling human-centric robot learning for cross-embodiment generalization. *arXiv preprint arXiv:2601.12993*, 2026.

NVIDIA. GR00T-N1.7-3B. Model card, 2026. URL https://huggingface.co/nvidia/GR00T-N1.7-3B.

Physical Intelligence. $\pi_{0.7}$: a steerable generalist robotic foundation model with emergent capabilities. Technical report, Physical Intelligence, 2026. URL https://www.pi.website/download/pi07.pdf.

Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Jiayuan Gu, Zhigang Wang, Yan Ding, Bin Zhao, Dong Wang, et al. SpatialVLA: Exploring spatial representations for visual-language-action models. In *Proceedings of Robotics: Science and Systems*, 2025.

Lin Sun, Bin Xie, Yingfei Liu, Hao Shi, Tiancai Wang, and Jiale Cao. GeoVLA: Empowering 3D representations in vision-language-action models. *arXiv preprint arXiv:2508.09071*, 2025.

Fangyuan Wang, Peng Zhou, Jiaming Qi, Shipeng Lyu, Chengyang He, David Navarro-Alarcon, and Guodong Guo. Think proprioceptively: State-grounded visual token selection for VLA policies. *arXiv preprint arXiv:2602.06575*, 2026.

Haochen Wang, Anlin Zheng, Yucheng Zhao, Tiancai Wang, Zheng Ge, Xiangyu Zhang, and Zhaoxiang Zhang. Reconstructive visual instruction tuning. In *International Conference on Learning Representations*, 2025.

Senqiao Yang, Chengyao Wang, Yuxin Chen, Zixuan Wang, Longxiang Tang, Haokun Gui, Jinhui Ye, Changsheng Lu, Xiaoyang Wu, Mingkang Zhu, Pengguang Chen, Shu Liu, Zhuotao Tian, Hengshuang Zhao, Bei Yu, and Jiaya Jia. Beyond data scaling: Representation-centric continued pre-training for vision-language-action models. *arXiv preprint arXiv:2608.27550*, 2026.

Chao Yu, Yuanqing Wang, Zhen Guo, Hao Lin, Si Xu, Hongzhi Zang, Quanlu Zhang, Yongji Wu, Chunyang Zhu, Junhao Hu, et al. RLinf: Flexible and efficient large-scale reinforcement learning via macro-to-micro flow transformation. In *20th USENIX Symposium on Operating Systems Design and Implementation (OSDI 26)*, pp. 829–846. USENIX Association, 2026.

Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. In *International Conference on Learning Representations*, 2025.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-World: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning*, pp. 1094–1100. PMLR, 2020.

Tianyi Zhang, Haonan Duan, Haoran Hao, Yu Qiao, Jifeng Dai, and Zhi Hou. Grounding actions in camera space: Observation-centric vision-language-action policy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 18782–18790, 2026.

Juntu Zhao, Wenbo Lu, Di Zhang, Yufeng Liu, Yushen Liang, Tianluo Zhang, Yifeng Cao, Junyuan Xie, Yingdong Hu, Shengjie Wang, et al. Do you need proprioceptive states in visuomotor policies? *arXiv preprint arXiv:2509.18644*, 2025.

Haoyu Zhen, Xiaowen Qiu, Peihao Chen, Jincheng Yang, Xin Yan, Yilun Du, Yining Hong, and Chuang Gan. 3D-VLA: A 3D vision-language-action generative world model. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 61229–61245. PMLR, 2024.

Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daumé, III, Andrey Kolobov, Furong Huang, and Jianwei Yang. TraceVLA: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. In *International Conference on Learning Representations*, 2025.

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*, pp. 2165–2183. PMLR, 2023.

APPENDIX OVERVIEW

This appendix is organized as follows. Appendix A summarizes the camera-perturbation diagnostics, and Appendix B provides the RVM and grounding-loss details. Appendix C describes the experimental setups, tasks, and evaluation protocols. Appendix D reports additional results and ablations, while Appendix E presents the grounding analyses and their protocols. Appendix F discusses limitations.

A SUMMARY OF CAMERA-PERTURBATION EXPERIMENTS

To test whether the policy relies on camera-specific pixel positions, we change only the camera while keeping the task, scene, instruction, and robot–object geometry fixed. Table 6 summarizes the six perturbations, Figure 1 illustrates them, and Appendix C.2.3 reports their magnitudes. Although the physical scene is unchanged, the predicted robot trajectories shift when the robot appears at different image positions. This suggests an *image-space shortcut*: the policy partly associates pixel positions with actions instead of relying on stable robot–object relationships.

Table 6: Summary of camera-perturbation diagnostics. Only the camera configuration is changed.

Camera	**Perturbation**	**Failure**	**Bias**
Wrist camera	Leftward shift	Grasp miss	Right-biased
Wrist camera	Rightward shift	Grasp miss	Left-biased
Frontal camera	Leftward angular shift	Bowl miss	Left-biased
Frontal camera	Rightward angular shift	Bowl miss	Right-biased
Frontal camera	Upward shift	Grasp miss	Upward-biased
Frontal camera	Downward shift	Grasp miss	Downward-biased

B ADDITIONAL METHOD DETAILS

B.1 RVM TARGET MINING DETAILS

RVM is computed offline from each demonstration trajectory before VLA fine-tuning. Each RGB frame is converted to grayscale, and dense optical flow is estimated between adjacent frames with Farnebäck's polynomial expansion method (Farnebäck, 2003). Let $\bar{I}_t$ denote the grayscale image at step t, and let ξ denote an image pixel. The appearance and flow cues used in Eq. (1) are

$$\begin{aligned} \Delta_t^+(\xi) &= |\bar{I}_{t+1}(\xi) - \bar{I}_t(\xi)|, \quad O_t^+(\xi) = \|F_{t\to t+1}(\xi)\|_2, \\ \Delta_t^-(\xi) &= |\bar{I}_t(\xi) - \bar{I}_{t-1}(\xi)|, \quad O_t^-(\xi) = \|F_{t-1\to t}(\xi)\|_2. \end{aligned} \tag{6}$$

We set $\lambda_\Delta = 0.5$ and $\lambda_f = 0.25$ in Eq. (1). Boundary terms that require frames outside the episode are set to zero. The normalization operator $\mathcal{N}$ applies robust score normalization followed by light Gaussian smoothing, which suppresses isolated noisy pixels while preserving coherent motion regions.

After computing the dense motion map M_t, RVM extracts its highest-scoring location as the peak $z_t \in [0,1]^2$. The high-score region around the peak defines an uncertainty $\sigma_t = (\sigma_{t,x}, \sigma_{t,y})$ and a local support box $B_t \subset [0,1]^2$. For token $p \in \mathcal{P}_t$, the anisotropic Gaussian score is

$$G_t(p) = \exp\left[-\frac{1}{2}\left(\left(\frac{u_{p,x} - z_{t,x}}{\sigma_{t,x}}\right)^2 + \left(\frac{u_{p,y} - z_{t,y}}{\sigma_{t,y}}\right)^2\right)\right]. \tag{7}$$

When the support box contains valid tokens, RVM clips the Gaussian to that local support before normalization:

$$\tilde{G}_t(p) = \begin{cases} G_t(p)\,\mathbf{1}[u_p \in B_t], & \sum_{q\in\mathcal{P}_t} G_t(q)\mathbf{1}[u_q \in B_t] > 0, \\ G_t(p), & \text{otherwise}, \end{cases} \qquad H_t(p) = \frac{\tilde{G}_t(p)}{\sum_{q\in\mathcal{P}_t} \tilde{G}_t(q)}. \tag{8}$$

This clipping prevents a broad Gaussian tail from assigning supervision to tokens outside the detected motion support.

The robot-activity cue is computed from end-effector displacement and gripper-state change:

$$r_t = \mathcal{R}_{\text{ep}}(\|e_t - e_{t-1}\|_2) + \beta_g \mathcal{R}_{\text{ep}}(|g_t - g_{t-1}|), \qquad \alpha_t = \mathcal{R}_{\text{ep}}(r_t), \tag{9}$$

where e_t and g_t denote the end-effector position and gripper state, respectively, $\beta_g = 0.5$, and $\mathcal{R}_{\text{ep}}$ denotes robust episode-level normalization.

The target confidence combines three normalized reliability cues. The peak confidence c_t^{peak} is computed from the peak-to-mean ratio of M_t, the motion-energy confidence c_t^{energy} from the average motion energy in M_t, and the robot activity α_t from Eq. (9). Their geometric mean gives

$$c_t = \left(c_t^{\text{peak}}\, c_t^{\text{energy}}\, \alpha_t\right)^{1/3}. \tag{10}$$

Targets with $c_t < \tau_c$ are excluded from the grounding loss. This filtering removes frames with ambiguous visual motion, low motion energy, or little robot activity from grounding supervision. Retained targets contribute in proportion to their confidence c_t, as specified in Eq. (4).

B.2 Grounding Normalization and Loss Details

Query and key normalization. Let $\ell_{t,m}$ denote the input query to the final decoder block and $\hat{v}_t(p)$ the position-augmented image token. The block computes

$$q_{t,m} = \mathcal{N}_2(W_q\, \text{LN}_q(\ell_{t,m})), \qquad k_t(p) = \mathcal{N}_2(W_k\, \text{LN}_k(\hat{v}_t(p))), \tag{11}$$

where W_q, W_k are learned linear projections, LN denotes layer normalization, and $\mathcal{N}_2(x) = x/\max(\|x\|_2, 10^{-12})$ is L2 normalization. The grounding logits are taken before the final block updates its queries.

Concentration of the predicted distribution. For a nonnegative logit scale s and $N = |\mathcal{P}_t|$ valid tokens, each per-query logit lies in $[-s, s]$. Assigning s to one token and $-s$ to all others gives the upper bound

$$a_{t,m}(p) \le \frac{e^{2s}}{e^{2s} + N - 1}, \qquad A_t(p) \le \frac{e^{2s}}{e^{2s} + N - 1}. \tag{12}$$

The same bound holds after averaging the per-query probabilities. The GR00T Bridge and Fractal configurations fix $s = 1$.

Loss normalization and confidence weighting. Equation (4) first excludes frames with no valid image tokens or with $c_t < \tau_c$, then weights each retained frame by c_t. The grounding loss is zero when no frame is retained. For $D_t = D_{\text{KL}}(H_t \| U_t)$ and $\varepsilon = 10^{-6}$, the complete per-frame rule is

$$\ell_t^{\text{g}} = \begin{cases} D_{\text{KL}}(H_t \| A_t)/D_t, & D_t > \varepsilon, \\ 0, & D_t \le \varepsilon. \end{cases} \tag{13}$$

The second branch avoids division by a near-zero uniform-baseline divergence.

C Experimental Setups and Evaluation Protocols

C.1 Evaluation Protocol

SimplerEnv camera perturbations. We independently fine-tune S^3-VLA with seeds 45, 46, and 47. Each checkpoint is evaluated on WidowX *Eggplant in sink* and Google Robot *Pick object*, using 200 rollouts per perturbation. We report the mean and sample standard deviation across seeds. For the Avg. row, we first average the seven perturbations within each seed. The standard-view results in Table 1 use one checkpoint per method and embodiment.

Meta-World. We evaluate each fixed checkpoint with 10 trials per task for each of three environment seeds (42, 43, and 44), giving 30 trials per task. The 50 tasks comprise 28 Easy, 11 Medium, 6 Hard, and 5 Very Hard tasks. Table 2 averages success across the three seeds; the overall mean weights all 50 tasks equally.

Real robot. For each task and method, we conduct 30 trials under each camera condition: Clean, Wrist, Front-H, and Front-V. Each perturbed condition averages two directions with 15 trials each. Across four tasks and two methods, this gives 960 trials. Per method, clean and perturbed success rates pool 120 and 360 trials, respectively. Table 10 reports success counts and 95% Wilson binomial confidence intervals.

C.2 Robot Setups

C.2.1 Simulation Robot Setups

SimplerEnv. We evaluate GR00T-N1.6 on WidowX tasks from Bridge and Google Robot tasks from Fractal in SimplerEnv (Li et al., 2025b). We report task success as the fraction of successful rollouts, using 200 rollouts per standard task. Appendix C.1 describes the multi-seed camera-perturbation protocol.

Using the GR00T codebase (Bjorck et al., 2025), we initialize the WidowX and Google Robot policies from nvidia/GR00T-N1.6-bridge and nvidia/GR00T-N1.6-fractal, respectively. We fine-tune each policy for 500 steps on an NVIDIA A100-SXM4-80GB GPU with an AMD EPYC 7473X 24-core CPU. SGA supplements the action prediction objective with offline localization targets for the robot's visible body in the front view. The grounding loss uses normalized KL over normalized image-token centers with a weight of 1.0. Table 7 summarizes the fine-tuning settings.

Table 7: Fine-tuning settings for SimplerEnv.

Setting	Value
Base checkpoint	GR00T-N1.6-bridge / GR00T-N1.6-fractal
Training steps	500
Learning rate	1e-4
Warmup ratio	0.03
Weight decay	1e-5
Fine-tuned top LLM layers	4

Meta-World. We evaluate $\pi_{0.5}$ (Black et al., 2025a) on Meta-World MT50 (Yu et al., 2020), which contains 50 tasks across four difficulty levels: Easy, Medium, Hard, and Very Hard. The training set contains 2,500 episodes, with 50 per task. We evaluate each task with 10 trials for each of three environment seeds (42, 43, and 44), giving 30 trials per task. A trial succeeds only when the task is fully completed. Appendix C.1 defines the task-weighted aggregation.

We fine-tune the RLinf/RLinf-Pi05-MetaWorld-SFT checkpoint for 1,000 steps on an NVIDIA A40 GPU using the Hydra-based supervised fine-tuning (SFT) pipeline in RLinf (Yu et al., 2026). The policy takes one image as input and predicts five-step action chunks in a four-dimensional action space. SGA supplements the action prediction objective with localization targets mined from the Meta-World training data. Table 8 summarizes the fine-tuning settings.

C.2.2 Simulation Camera Perturbations

We perturb the third-person camera for WidowX and the overhead camera for Google Robot while keeping camera intrinsics and the physical scene fixed. In Table 3, Yaw, Pitch, and Roll denote $+10^\circ$ rotations, while Forward, Right, and Up denote $+2\,\mathrm{cm}$ translations. Composite combines yaw $+5^\circ$, pitch -5°, rightward $1\,\mathrm{cm}$, and upward $1\,\mathrm{cm}$. Rotations are about the camera center, and positional shifts follow the nominal camera axes.

Table 9 reports the mean and sample standard deviation across fine-tuning seeds 45, 46, and 47. Each seed is evaluated with 200 rollouts per task and camera condition, following Appendix C.1.

Table 8: Fine-tuning settings for Meta-World.

Setting	Value
Task configuration	
Training paradigm	Supervised fine-tuning, SFT
Environment / dataset	metaworld_mt50, LeRobot-v2.1 format
Model	
Base policy	OpenPI $\pi_{0.5}$
OpenPI configuration	pi05_metaworld
Initialization checkpoint	RLinf/RLinf-Pi05-MetaWorld-SFT
Action chunk length	5
Action dimension	4
Value head / critic	Disabled
Training	
Gradient checkpointing	Disabled
Microbatch size	4
Global batch size	32
Training seed	42
Training steps	1000
Adam optimizer	
Learning rate	2.5e-5
Minimum learning rate	2.5e-6
Learning rate schedule	Cosine decay
Warmup steps	30
Betas	(0.9, 0.95)
Weight decay	1.0e-10
Gradient clipping	1.0

Table 9: S^3-VLA success (%, mean $\pm$ sample standard deviation) across three fine-tuning seeds. The Avg. row first averages the seven perturbations within each seed.

Camera perturbation	WidowX	Google Robot
Yaw $+10^\circ$	12.2 ± 8.7	73.3 ± 3.3
Pitch $+10^\circ$	4.3 ± 5.9	59.0 ± 4.1
Roll $+10^\circ$	31.3 ± 17.9	93.3 ± 1.5
Forward $+2\,\mathrm{cm}$	64.5 ± 9.5	93.8 ± 1.0
Right $+2\,\mathrm{cm}$	44.2 ± 22.9	97.2 ± 1.0
Up $+2\,\mathrm{cm}$	22.8 ± 18.4	94.7 ± 1.9
Composite	35.7 ± 17.5	65.3 ± 3.4
Avg.	30.7 ± 9.2	82.4 ± 0.3

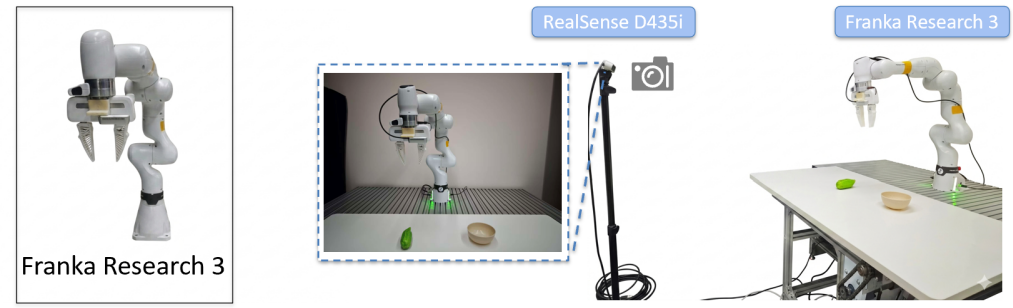
C.2.3 Real World Robot Setups

We use a Franka Research 3 (FR3) arm with a parallel gripper and Intel RealSense D435i RGB-D cameras for both wrist and frontal views. Figure 9 shows the robot and workspace, and Figure 10 shows the four task setups.

Training protocol. Both the GR00T-N1.6 baseline and S^3-VLA are initialized from the same original GR00T-N1.6 checkpoint and fine-tuned for one epoch on the same real-world dataset, with 200 trajectories per task. The baseline uses the standard action prediction objective, while S^3-VLA adds SGA.

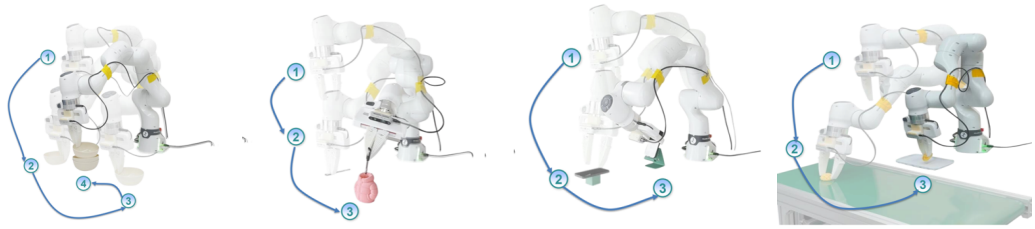
Camera perturbations. Figure 1 shows the six setups. **Wrist** shifts the wrist camera left or right by $3\,\mathrm{cm}$; **Front-H** changes the frontal camera angle left or right by about 15°; **Front-V** shifts it up or

down by 10 cm. The task and other camera remain fixed. Each reported success rate averages the two directions, with 15 trials per direction. The clean condition uses 30 trials per task and method.



(a) Franka Research 3. (b) Workspace setup with RGB view.

Figure 9: **Real-world experimental setup.** (a) The Franka Research 3 robot arm used in our experiments. (b) The physical workspace equipped with the robot arm and two Intel RealSense D435i RGB-D cameras for wrist and frontal views; the inset shows the frontal RGB view.



(a) Stack Bowls (b) Insert Pen (c) Place Phone (d) Conveyor Belt

Figure 10: Real-world task setups for the four manipulation tasks.

C.3 Simulation Task Design

To comprehensively evaluate S^3-VLA, we select tasks across two simulation platforms: SimplerEnv and Meta-World. We adopt the following tasks in the WidowX + Bridge setting: *Spoon on towel*, *Carrot on plate*, *Eggplant in basket*, *Stack cube*, *Eggplant in sink*, *Close drawer*, *Open drawer*. For the Google Robot + Fractal setting, we test the following tasks: *Pick coke can*, *Pick object*, *Move near*, *Open drawer*, *Close drawer*, *Place in closed drawer*. We evaluate SimplerEnv task success rates over 200 rollouts per task. In Meta-World, MT50 evaluation uses all 50 Meta-World environments. We present the tasks in Figure 11, Figure 12, and Figure 13.

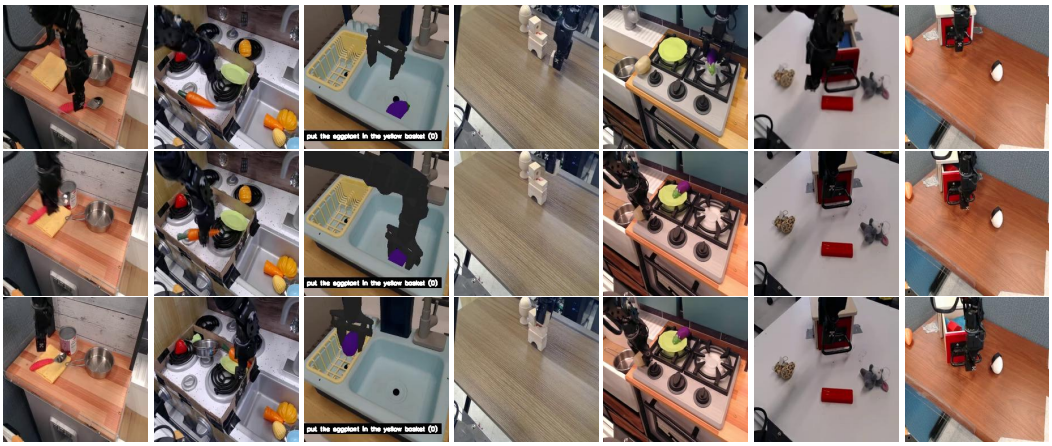


Figure 11: **WidowX + Bridge Benchmark.** 7 tasks in the Bridge benchmark.

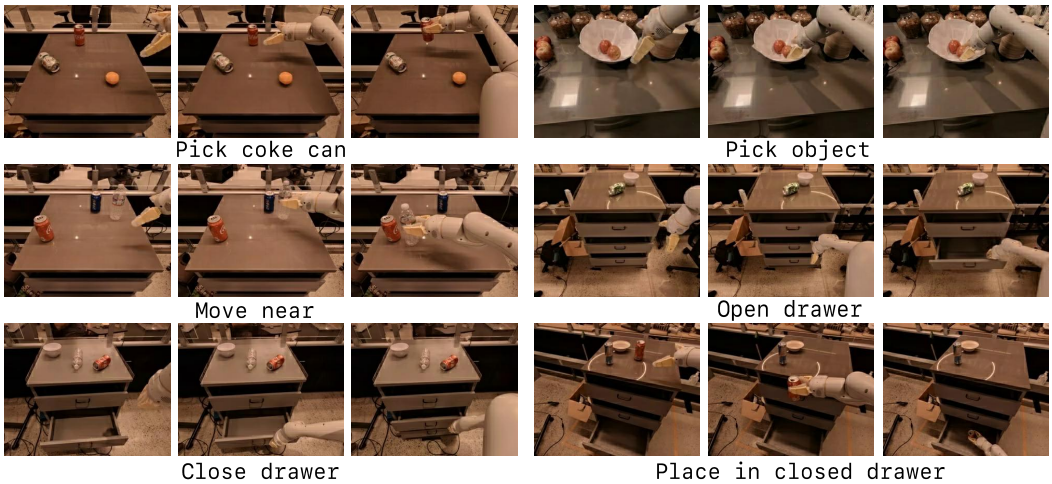


Figure 12: **Google Robot + Fractal Benchmark.** 6 tasks in Fractal benchmark.

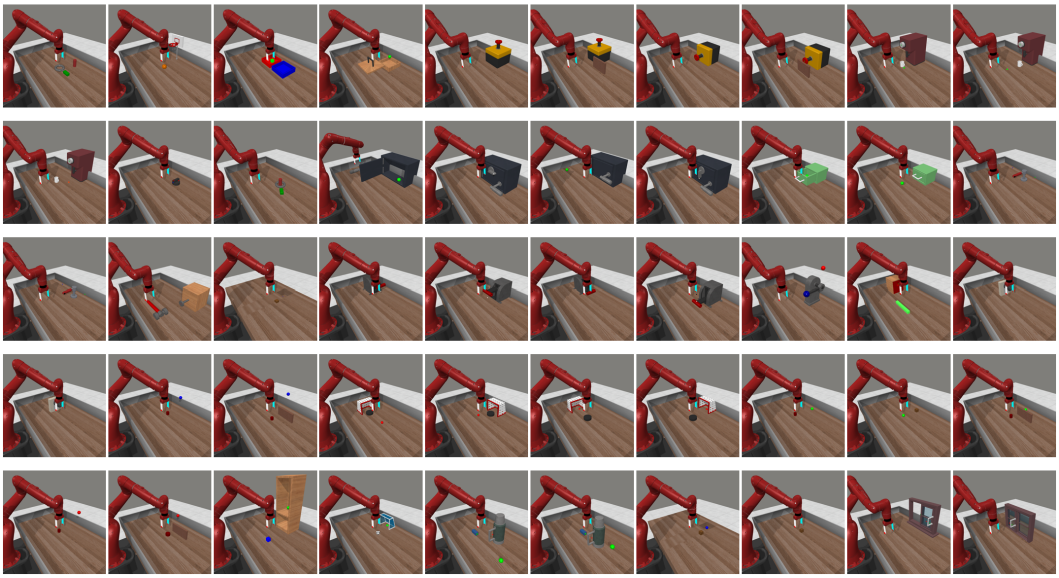


Figure 13: **Meta-World Benchmark.** MT50 evaluation uses all 50 Meta-World environments.

D Additional Results and Ablations

D.1 Expanded Real-World Evaluation

Table 10 reports the trial counts underlying Figure 5. S^3-VLA achieves higher success in 11 of 12 camera-perturbed conditions and three of four clean-view conditions. The two conditions where GR00T-N1.6 is ahead, *Conveyor Belt* under the clean view and *Insert Pen* under Front-H, each differ by one successful trial, so the pooled gain is not carried by a single task or view. Front-V, where the baseline is weakest, shows the largest gain: pooled over tasks, success rises from 11/120 to 31/120 trials.

Table 10: Real-world success under clean and perturbed camera views. Both methods fine-tune GR00T-N1.6 on the same task data. Entries show successful/total trials, success rates (%), and 95% Wilson intervals. Each perturbation includes 15 trials in each direction. Pooled rows combine results across tasks within the clean or perturbed setting.

		GR00T-N1.6		S^3-VLA	
Condition	**Task**	**Count**	**Rate [95% CI]**	**Count**	**Rate [95% CI]**
Clean	Stack Bowls	14/30	46.7 [30.2, 63.9]	17/30	56.7 [39.2, 72.6]
	Insert Pen	9/30	30.0 [16.7, 47.9]	13/30	43.3 [27.4, 60.8]
	Place Phone	8/30	26.7 [14.2, 44.4]	13/30	43.3 [27.4, 60.8]
	Conveyor Belt	22/30	73.3 [55.6, 85.8]	21/30	70.0 [52.1, 83.3]
Wrist	Stack Bowls	5/30	16.7 [7.3, 33.6]	8/30	26.7 [14.2, 44.4]
	Insert Pen	4/30	13.3 [5.3, 29.7]	8/30	26.7 [14.2, 44.4]
	Place Phone	1/30	3.3 [0.6, 16.7]	4/30	13.3 [5.3, 29.7]
	Conveyor Belt	9/30	30.0 [16.7, 47.9]	12/30	40.0 [24.6, 57.7]
Front-H	Stack Bowls	8/30	26.7 [14.2, 44.4]	12/30	40.0 [24.6, 57.7]
	Insert Pen	6/30	20.0 [9.5, 37.3]	5/30	16.7 [7.3, 33.6]
	Place Phone	2/30	6.7 [1.8, 21.3]	7/30	23.3 [11.8, 40.9]
	Conveyor Belt	13/30	43.3 [27.4, 60.8]	19/30	63.3 [45.5, 78.1]
Front-V	Stack Bowls	5/30	16.7 [7.3, 33.6]	12/30	40.0 [24.6, 57.7]
	Insert Pen	0/30	0.0 [0.0, 11.4]	1/30	3.3 [0.6, 16.7]
	Place Phone	2/30	6.7 [1.8, 21.3]	5/30	16.7 [7.3, 33.6]
	Conveyor Belt	4/30	13.3 [5.3, 29.7]	13/30	43.3 [27.4, 60.8]
Clean pooled		53/120	44.2 [35.6, 53.1]	64/120	53.3 [44.4, 62.0]
Perturbed pooled		59/360	16.4 [12.9, 20.6]	106/360	29.4 [25.0, 34.3]

D.2 Detailed Component Ablations

To understand which components drive the gains in Table 4, Table 11 reports task-level results for four ablations. We remove localization supervision or either motion cue. The "w/o normalized KL" variant replaces the localization objective with Gaussian cross-entropy (Gaussian CE).

Removing localization supervision reverses most of the gains on the two eggplant tasks, which benefit most from S^3-VLA in Table 1: success falls from 89.0 to 59.5 on *eggplant in basket* and from 69.0 to 49.5 on *eggplant in sink*. The two motion cues fail on different tasks. Keeping only image differences is better on *carrot on plate* (73.0 vs. 60.0), whereas keeping only optical flow is better on *eggplant in sink* (65.5 vs. 44.5). Both single-cue variants, however, lose most of the *eggplant in basket* gain (55.5 and 66.0) and fall below the variant without grounding on Fractal *open drawer* (50.5 vs. 65.0). Noisy targets can thus be worse than no grounding, and the tasks that benefit most from self-grounding need both cues. Single-cue variants exceed the full model on only three tasks: *stack cube*, where every variant stays below 17%; *carrot on plate* without optical flow (73.0 vs. 66.5); and *move near* without optical flow, by one rollout (96.5 vs. 96.0).

The "w/o normalized KL" variant keeps most of the *eggplant in basket* gain (87.5) but recovers less than half of the *eggplant in sink* gain (58.5, vs. 49.5 without grounding and 69.0 for the full model), and it falls below action-only training on *spoon on towel* (66.5 vs. 75.0). This task dependence matches the scale sensitivity of the unnormalized loss analyzed in Appendix E.4.

Table 11: Task success rates (%) for component ablations in SimplerEnv. Each task is evaluated over 200 rollouts.

	Bridge (WidowX)							**Fractal (Google Robot)**					
Variant	Spoon on towel	Carrot on plate	Stack cube	Eggplant in basket	Eggplant in sink	Open drawer	Close drawer	Pick coke can	Move near	Open drawer	Close drawer	Place in closed drawer	Pick object
w/o $\mathcal{L}_{\mathrm{g}}$	75.0	61.5	6.5	59.5	49.5	91.5	100.0	100.0	89.5	65.0	65.5	3.5	95.5
w/o normalized KL	66.5	54.5	8.0	87.5	58.5	95.5	95.0	100.0	86.0	68.0	71.0	4.0	96.5
w/o image diff	74.0	60.0	16.0	55.5	65.5	87.5	96.0	100.0	96.0	50.5	71.5	13.0	93.0
w/o optical flow	79.0	73.0	15.5	66.0	44.5	85.5	82.0	99.0	96.5	50.5	69.0	10.0	84.5

D.3 State-Prediction Controls

We compare SGA with two state-based auxiliary objectives under matched training settings. Using the same visual features, a lightweight predictor with learned queries estimates either the normalized end-effector position and gripper state s_t or its standardized one-step change $\Delta s_t = (s_{t+1} - s_t)/\sigma$, where σ denotes the per-coordinate training-set standard deviation. The queries do not receive the target, and the prediction is trained with an MSE loss of weight 0.2. The auxiliary loss is used only during training; predictions are not fed back to the policy, and neither variant uses robot-location supervision. All other training settings match the w/o $\mathcal{L}_{\mathrm{g}}$ variant.

We evaluate each method over three seeds on WidowX *Eggplant in sink* and Google Robot *Pick object*, with 200 rollouts per seed and camera setting. Table 5 reports mean success across seeds for the original view and the average over seven camera shifts. Predicting Δs_t performs better than predicting s_t, but S^3-VLA outperforms both variants and w/o $\mathcal{L}_{\mathrm{g}}$ in both settings.

D.4 Spatial Forcing Comparison

We adapt Spatial Forcing (Li et al., 2026b) to GR00T-N1.6 using the same Bridge training data and schedule. It transfers representations from a frozen VGGT teacher, whereas S^3-VLA derives targets

from demonstration motion without an external teacher. Table 12 evaluates the Bridge/WidowX *eggplant in sink* task under each camera perturbation. Spatial Forcing obtains 19.7% on average, 1.6 points above the 18.1% of the GR00T-N1.6 checkpoint, whereas S^3-VLA reaches $30.7 \pm 9.2\%$. The two methods also help under different shifts. Spatial Forcing leaves roll unchanged (7.0) and lowers success under upward shifts (2.5 vs. 4.5), where S^3-VLA gains 24.3 and 18.3 points; it is strongest on the composite shift (39.0 vs. 35.7 for S^3-VLA). Aligning features with a 3D teacher enriches scene geometry without singling out the robot's own body, the cue that SGA supervises.

In the original-camera Bridge comparison, success rates are 60.1%, 65.6%, and 74.1% for GR00T-N1.6, Spatial Forcing, and S^3-VLA, respectively.

Table 12: Bridge camera-perturbation success (%) for the Spatial Forcing adaptation on GR00T-N1.6. Each condition uses 200 rollouts per checkpoint. For S^3-VLA, values are mean $\pm$ sample standard deviation across three fine-tuning seeds.

Method	Yaw	Pitch	Roll	Forward	Right	Up	Composite	**Avg.**
GR00T-N1.6	**14.5**	2.0	7.0	48.0	24.0	4.5	27.0	18.1
Spatial Forcing	7.0	1.0	7.0	53.5	28.0	2.5	**39.0**	19.7
S^3-VLA	12.2 $\pm$8.7	**4.3** $\pm$5.9	**31.3** $\pm$17.9	**64.5** $\pm$9.5	**44.2** $\pm$22.9	**22.8** $\pm$18.4	35.7 $\pm$17.5	**30.7** $\pm$9.2

D.5 Comparison of Localization Target Sources

Motion-mined targets are noisier than ground-truth robot masks. To measure what this noise costs, we replace the RVM targets with simulator-projected robot masks under matched training and evaluation settings (Table 13).

RVM trails the projected masks by 2.3 points on Bridge and exceeds them by 0.4 points on Fractal. Ground-truth masks therefore add little over motion-mined targets, which need neither simulator access nor camera calibration and thus also apply to real-world demonstrations.

Table 13: Task success (%) with simulator-projected robot masks or RVM targets in the target-source comparison.

Supervision	**Bridge**	**Fractal**
Simulator-projected robot mask	76.4	75.7
RVM	74.1	76.1

E Additional Grounding Analyses

E.1 Relevance Map Visualization

Figure 7 visualizes relevance maps derived from self-attention in the 16th layer of the GR00T-N1.6 VLM backbone. Both models receive the same front-view image and prompt: "Where is the robot itself". We average attention to image tokens over all heads and non-image query tokens. The maps are resized to the image resolution and independently min–max normalized for display.

E.2 Linear-Probe Evaluation Protocol

Figure 6 uses 135 clean-view fitting frames from Carrot on Plate, Spoon on Towel, Stack Cube, and Open Drawer. Evaluation adds Close Drawer and both eggplant tasks, which are excluded from probe fitting but not from policy training. After excluding overlapping states and episodes whose initial image matches a fitting image, evaluation uses 177 states rendered from five views (885 images): clean, yaw $\pm 10^\circ$, and right translation $\pm 2\,\mathrm{cm}$.

Each frozen policy has a separately fitted linear head of identical capacity on its 81 image tokens at the action-head input (2,048 features per token, with layer normalization). The head predicts robot occupancy with binary cross-entropy and gripper location with spatial cross-entropy, using simulator masks downsampled to the 9×9 token grid. Fitting uses frames with visible robot and gripper masks,

AdamW for 1,500 steps, batch size 16, learning rate 10^{-3}, and weight decay 10^{-4}. Predictions from three probe fits are averaged for each fixed policy checkpoint.

Gripper error is the distance from the predicted distribution's center to the visible gripper-mask center, divided by the image diagonal. PCK is the fraction within a given error threshold. Robot-mask AP compares the upsampled occupancy map with the simulator mask. Metrics average frames within episodes, episodes within initial scenes, and scenes within tasks; task-group summaries weight tasks equally. Perturbed-view results average the four camera shifts. Intervals use 5,000 bootstrap resamples of initial scenes within each task, conditional on the selected tasks and checkpoints. Panel (d) uses paired resampling of 78 matched states from 14 initial scenes on new trajectories of the four fitting tasks.

E.3 Robot-Target Attribution Protocol

Actor-mask audit on action-only trajectories. Figure 8 uses a separate audit of 40 action-only trajectories across seven WidowX tasks, including successful and failed rollouts. We use all 3,668 recorded frames from the trajectory collection used for the main-text localization analysis. Of these frames, 3,465 have valid motion targets, and 3,275 pass the original confidence threshold $c_t \geq 0.2$. We assign each retained peak to a mutually exclusive category using simulator actor masks: Robot combines the gripper and other robot links; Object denotes the task object being manipulated; Fixture denotes a destination object or task apparatus; and Other contains all remaining scene pixels and outside-crop points. The resulting counts are 2,687, 10, 50, and 528, respectively. Other includes 527 scene points and one outside-crop point. The displayed proportions pool retained points across frames.

For the grasp-state analysis, we restrict the audit to the five tasks with a manipulated object, excluding the drawer tasks. A grasp is defined by the simulator's current `check_grasp(source_entity)` value recorded during trajectory collection. This yields 438 grasp-state frames, of which 399 have valid motion targets and 358 pass the same confidence threshold. The retained subset spans 20 trajectories and 17 initial-scene clusters. The five tasks shown in Figure 8b contain 24, 102, 96, 75, and 61 retained points, respectively; pooling them gives 269 Robot, 7 Object, 3 Fixture, and 79 Other points. Object here is the object currently grasped. Both plotted fractions use all retained points in the corresponding group as their denominator. The "All grasped" summary pools retained points across the five tasks.

E.4 Additional Analysis of Gaussian CE and Normalized KL Objectives

We compare the normalized KL grounding loss in Eq. (4) with the Gaussian CE objective used by the "w/o normalized KL" variant, where CE denotes cross-entropy. Given a point pseudo-label $z_t \in [0,1]^2$, Gaussian CE first forms a token-level target

$$T_t^\sigma(p) = \frac{\exp\big(-\|u_p - z_t\|_2^2/(2\sigma^2)\big)}{\sum_{q\in\mathcal{P}_t}\exp(-\|u_q - z_t\|_2^2/(2\sigma^2))}. \tag{14}$$

It then minimizes the per-frame cross-entropy

$$\ell_t^{\mathrm{CE}} = -\sum_{p\in\mathcal{P}_t} T_t^\sigma(p)\log(A_t(p)+\varepsilon). \tag{15}$$

In training, this term uses a confidence-weighted average over pseudo-labels that pass the confidence threshold and is balanced against the action objective by an auxiliary-loss cap γ. Unlike normalized KL, this objective does not divide the prediction error by the information content of the target distribution, so its effective supervision can change substantially with target sharpness, token support, and the chosen cap.

Figure 14 compares grounding maps on high-confidence *Stack Bowls* samples. Gaussian CE gives diffuse responses at $\gamma = 1$. Raising the cap to $\gamma = 100$ increases activation around the robot. In these examples, normalized KL focuses more closely on the robot's visible body, with less background activation. Gaussian CE thus depends on the cap, as the scale analysis above anticipates: even a 100-fold larger cap only partly focuses it on the robot, whereas normalized KL fixes the supervision scale through its denominator.

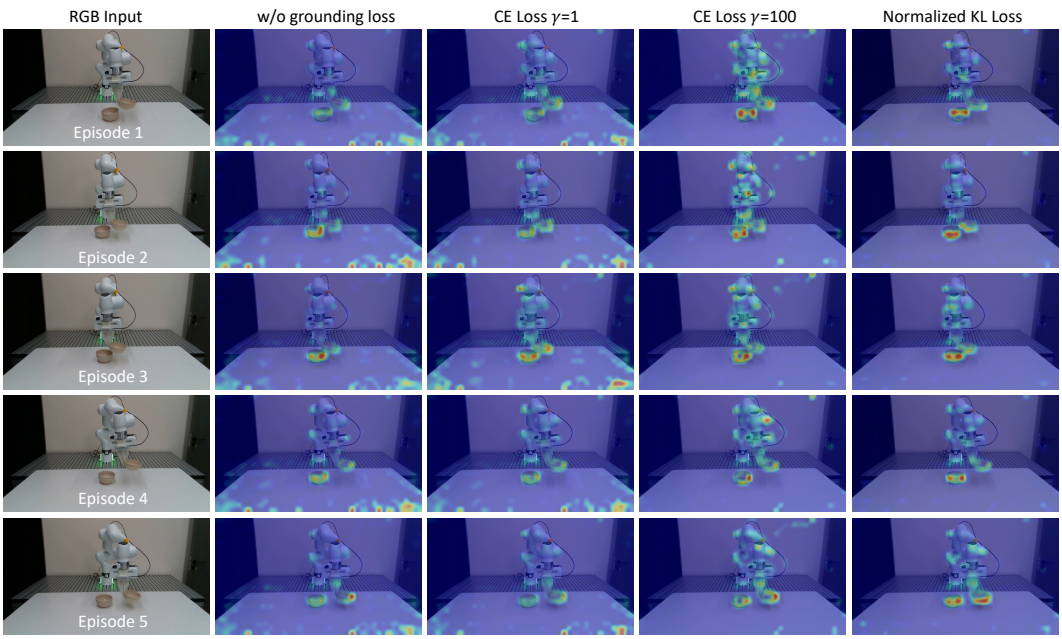


Figure 14: Qualitative comparison of grounding objectives on high-confidence *Stack Bowls* samples. Columns show the RGB input, w/o grounding loss, Gaussian CE with auxiliary-loss cap $\gamma = 1$, Gaussian CE with cap $\gamma = 100$, and the normalized KL objective. All rows use the same prompt. Within each row, the methods are compared on the same front-view image. The overlays do not show pseudo-label boxes.

E.5 Additional Control Analyses

Relationship between grounding quality and task success. To examine the role of visible robot cues in task execution, we occlude half or all of the gripper and compare with equal-area background occlusions. Matching the occlusion area and texture controls for the extent and appearance of the occluded region. We evaluate a fixed S^3-VLA checkpoint with 60 rollouts per condition and measure its trained grounding head alongside task success. For each episode, grounding quality Q is the fraction of valid policy queries whose grounding peak lies inside the gripper mask.

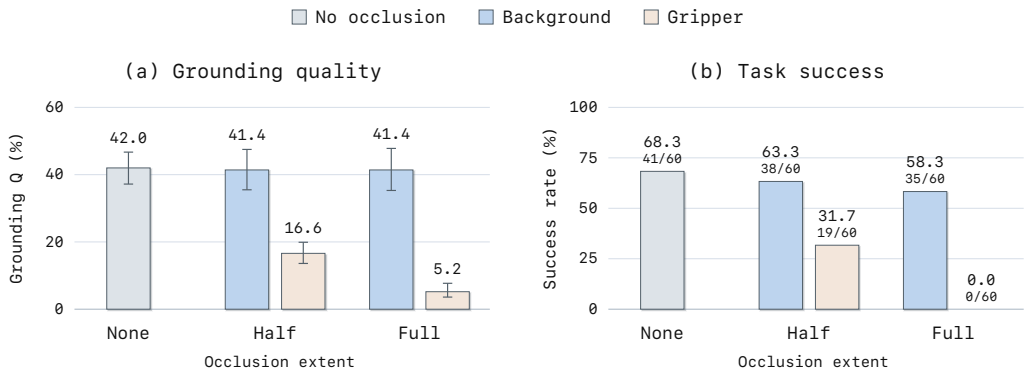


Figure 15: Grounding quality and task success under gripper and matched background occlusions. Each condition uses 60 rollouts. (a) Grounding quality Q with 95% confidence intervals. (b) Task success rates with successful rollout counts. Both panels use the same condition order, with separate vertical scales.

Background occlusions preserve grounding quality at 41.4%, close to the unoccluded value of 42.0%, while retaining substantially higher success than the matched gripper occlusions (Figure 15). Occluding the gripper reduces localization quality and success together, with a larger effect under full occlusion. Since equal-area background occlusions barely change grounding quality, the drop in

success is specific to hiding the robot: the policy relies on seeing its own gripper, the cue that SGA trains it to localize.

Action consistency under camera shifts. To test whether action predictions remain consistent when only the view changes, we evaluate 192 simulator states while holding the scene and robot state fixed. Moving the camera from $-2\,\mathrm{cm}$ to $+2\,\mathrm{cm}$ changes the image while preserving the physical task. Under this controlled comparison, S^3-VLA reduces the endpoint shift of the predicted eight-step trajectory from $5.54\,\mathrm{cm}$ to $4.78\,\mathrm{cm}$, a reduction of $0.76\,\mathrm{cm}$ relative to the GR00T-N1.6 baseline. Because the physical scene is fixed, this endpoint shift is caused by the camera alone and directly measures the *image-space shortcut* of Section 3; SGA reduces it by 14%.

F LIMITATIONS

Front-view target mining. S^3-VLA currently mines RVM pseudo-labels from external front-view observations, so it depends on motion of the robot's visible body under a mostly static camera. The targets can be weak when the robot is out of view or heavily occluded, and noisy when task objects, deformable materials, backgrounds, or other agents move in sync with the robot.

Multi-view extension. We have not yet studied pseudo-label extraction from wrist-view observations. A wrist camera moves with the robot, so parts of the robot's visible body can remain relatively stationary in visual observations while the surrounding scene moves. This complements front-view videos, where the camera is static and the robot moves. Future work can exploit this contrast and jointly model front and wrist views.