SPAVLA: FROM EXPLICIT 3D SPATIAL UNDERSTANDING TO ROBOTIC MANIPULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision–language–action (VLA) models inherit rich semantic knowledge from pretrained vision–language models (VLMs), yet developing 3D spatial understanding and using it to guide action generation remain challenging. We propose **SpaVLA**, a spatially enhanced VLA model that connects geometric perception to action generation through explicit, manipulation-relevant 3D spatial understanding. To enhance the 3D spatial capabilities, we first construct **ManiSpatial-565K**, which pairs robot manipulation data with 3D spatial question-answering (QA) annotations to provide fine-grained, manipulation-relevant supervision. To support reasoning about these spatial relationships, we introduce a 3D-aware module that incorporates geometric representations into SpaVLA's VLM backbone. We then introduce a progressive training strategy to align this enhanced spatial understanding with robotic control. By conditioning the action expert on fine-grained spatial reasoning generated by the VLM, SpaVLA connects geometric perception to action generation, enabling the VLA model to act on manipulation-relevant 3D spatial relationships. To evaluate how well this spatial understanding supports manipulation, we further introduce **SimplerEnv-Spatial**, a 20-task benchmark for spatial referring and reasoning. SpaVLA achieves 48.6% on MMSI-Bench and 95.6% on MindCube, with manipulation success rates of 84.4% on Google Robot under SimplerEnv's Visual Matching protocol and 65.6% on SimplerEnv-Spatial. Real-world experiments further demonstrate spatial instruction following under object rearrangement, destination relocation, and illumination changes. The dataset and code will be publicly available.

1 INTRODUCTION

Recent advances in large-scale vision–language pretraining and multimodal instruction tuning have enabled vision–language models (VLMs) to achieve strong performance in visual recognition, language understanding, and multimodal reasoning (Radford et al., 2021; Liu et al., 2023a). Building on these advances, vision–language–action (VLA) models such as RT-2 (Zitkovich et al., 2023), OpenVLA (Kim et al., 2025), and π_0 (Black et al., 2025) extend pretrained VLMs to robot action generation, aiming to transfer visual and semantic knowledge acquired from internet-scale data to language-conditioned robotic manipulation. Robots must not only identify what to manipulate, but also determine where objects are in 3D, how they relate spatially, and how these relationships should guide their actions. However, the semantic knowledge inherited from VLMs does not by itself ensure this fine-grained 3D spatial understanding (Chen et al., 2024a; Cheng et al., 2024).

Recent efforts have strengthened spatial capabilities in VLAs by incorporating 3D geometric priors and improving image-plane grounding. SpatialVLA (Qu et al., 2025) injects 3D positional information into visual representations, while Spatial Forcing (Li et al., 2025c) aligns policy features with representations from a pretrained 3D foundation model. These mechanisms enrich policy features with geometric cues, but do not directly supervise the VLM to express manipulation-relevant 3D properties and spatial relations. Reconstructing task-relevant image regions (Song et al., 2025b) and predicting points, bounding boxes, and trajectories (Chen et al., 2025) also improve 2D grounding.

However, these spatial supervision targets primarily describe image-plane regions and coordinates, rather than explicitly characterizing object depth, physical size, and 3D spatial relationships. This motivates us to combine 3D geometric representations with manipulation-relevant spatial supervi-

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.

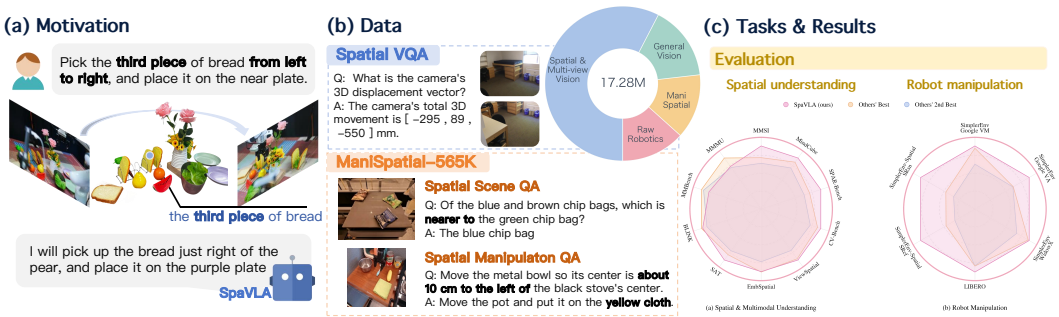


Figure 1: **SpaVLA: spatial understanding for robotic manipulation.** **(a)** Spatial reasoning resolves spatially constrained instructions into explicit manipulation targets and destinations. **(b)** ManiSpatial-565K pairs robot manipulation data with spatial scene QA and spatial manipulation QA to enhance manipulation-relevant 3D understanding. **(c)** SpaVLA demonstrates strong spatial understanding and competitive robotic manipulation performance.

sion, so that the VLM learns to interpret and express task-relevant 3D properties and relationships, and the VLA uses this understanding to guide action generation.

To this end, we introduce **SpaVLA**, a spatially enhanced VLA framework for robotic manipulation (see figure 1). We first strengthen the VLA's 3D spatial understanding by enhancing its VLM backbone with manipulation-relevant supervision and geometric representations. To provide this supervision, we construct **ManiSpatial-565K**, a fine-grained 3D spatial question-answering (QA) dataset aligned with robot manipulation trajectories. Scene-level QA teaches the VLM to describe 3D properties and spatial relationships, while trajectory-aligned instruction–answer pairs teach it to use spatial information to identify the intended manipulation target or operation. Complementing this supervision, a 3D-aware module integrates VGGT's geometric representations (Wang et al., 2025a) with the VLM's visual–semantic features, providing cues for interpreting these properties and relationships. We then align this enhanced spatial understanding with robotic control through a progressive training strategy that incorporates joint learning of spatial QA and action prediction. Beyond joint optimization, task-relevant spatial answers generated by the VLM explicitly condition the action expert, enabling SpaVLA to use the learned 3D spatial information to guide action generation. To evaluate how well this spatial understanding supports manipulation, we further introduce **SimplerEnv-Spatial**, a 20-task benchmark for spatial referring and reasoning. By assessing spatial instruction following through manipulation success, this benchmark complements public spatial-understanding benchmarks and standard SimplerEnv evaluations.

SpaVLA achieves 48.6% accuracy on MMSI-Bench and 95.6% on MindCube, with the same checkpoint reaching 65.6% task success on SimplerEnv-Spatial. In real-world experiments on a Franka Research 3 robot, SpaVLA achieves 83.3% success on pick-and-place tasks with ordinal references in rearranged layouts under partial occlusion. Answer interventions in simulation show that spatial answers influence the robot's target selection.

Our main contributions are as follows:

- We propose **SpaVLA**, a spatially enhanced VLA framework that integrates geometric representations with visual–semantic features to strengthen manipulation-relevant 3D spatial understanding and uses VLM-generated spatial answers to condition action generation.
- We construct **ManiSpatial-565K**, a fine-grained 3D spatial QA dataset aligned with robot manipulation trajectories. It provides supervision for understanding 3D properties and spatial relationships, as well as resolving spatial instructions into explicit manipulation targets or operations.
- We introduce a **progressive training strategy** that strengthens the VLM's spatial capabilities, then aligns this understanding with robotic control by jointly learning spatial QA and action prediction.

2 Related Work

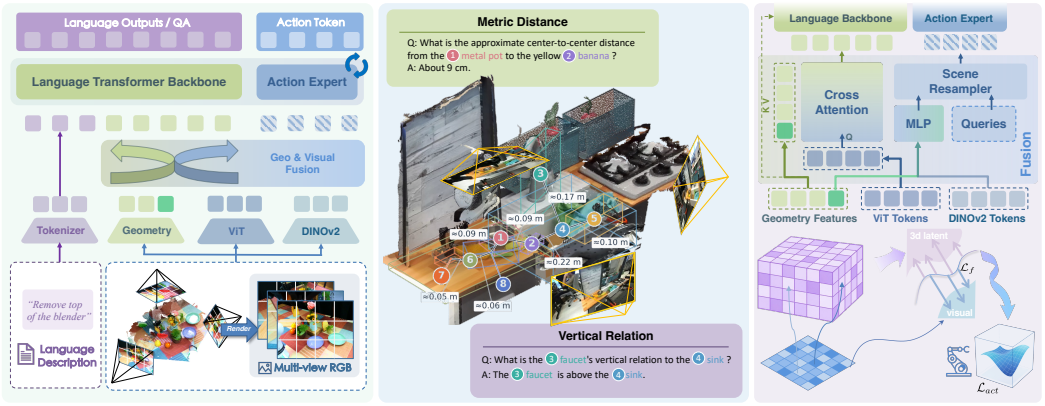


Figure 2: **SpaVLA architecture.** Visual–geometric fusion enhances manipulation-relevant 3D understanding. The action expert generates continuous action chunks from visual–geometric scene tokens and layer-wise VLM representations, including available task-resolving spatial answers.

Spatial reasoning in multimodal language models. Recent work improves spatial reasoning in VLMs through spatial supervision, geometry-aware representations, and explicit reasoning. SpatialVLM (Chen et al., 2024a) scales spatial question answering through automatically generated 3D annotations, while RoboSpatial (Song et al., 2025a) provides robotics-oriented spatial supervision across multiple reference frames. SpatialRGPT (Cheng et al., 2024) combines region-level supervision with depth-aware visual features to support grounded spatial reasoning. To incorporate geometric priors from RGB observations, VG-LLM (Zheng et al., 2025a) and Spatial-MLLM (Wu et al., 2025) integrate visual–semantic features with representations from pretrained geometry models. SpatialStack (Zhang et al., 2026) further explores hierarchical geometry–language fusion across multiple network layers. For explicit spatial reasoning, SpatialReasoner (Ma et al., 2025) uses shared 3D representations to connect perception, geometric computation, and reasoning.

Spatially enhanced VLA policies. Spatially enhanced VLA policies use enriched representations, spatially guided training, and intermediate reasoning. PointVLA (Li et al., 2025a) injects point-cloud features into the action expert, whereas BridgeVLA (Li et al., 2025e) aligns multi-view projections of 3D observations with 2D heatmap-based action prediction. SpatialVLA (Qu et al., 2025) introduces egocentric 3D positional encodings and adaptive spatial action discretization. FALCON (Zhang et al., 2025b) conditions a spatially enhanced action head on geometric tokens, while 3D-Mix (Yu et al., 2026) studies semantic-conditioned gated fusion of visual and geometric features. Through auxiliary supervision, ReconVLA (Song et al., 2025b) reconstructs task-relevant image regions to improve visual grounding, while Spatial Forcing (Li et al., 2025c) aligns intermediate policy representations with geometric features from pretrained 3D models. RT-2 (Zitkovich et al., 2023) and $\pi_{0.5}$ (Physical Intelligence et al., 2025) combine robotic and non-robotic supervision to transfer multimodal knowledge to control. InternVLA-M1 (Chen et al., 2025) combines 2D spatial grounding pretraining with spatially guided joint training to preserve and leverage spatial understanding for action generation. For intermediate reasoning, ECoT (Zawalski et al., 2024) generates embodied reasoning steps before actions, while GraphCoT-VLA (Huang et al., 2025) combines structured reasoning with a dynamic 3D pose–object graph. In contrast, SpaVLA first strengthens 3D spatial understanding through geometric representations and trajectory-aligned 3D QA supervision, then progressively transfers this capability to the action expert. Manipulation-relevant spatial responses provide an explicit intermediate interface for action conditioning.

3 METHOD

Given current RGB observations $o_t = \{I_t^{(v)}\}_{v=1}^{N_v}$ from N_v camera views and a manipulation instruction or spatial query u, SpaVLA predicts an action chunk $A_t \in \mathbb{R}^{H \times d_a}$ with horizon H and action dimension d_a. As shown in Figure 2, geometric features augment the VLM and directly condition

the action expert. For answer-conditioned control, the VLM additionally produces a spatial answer r_t resolving the intended manipulation target or operation.

3.1 Geometry-Enhanced Visual-Language Representations

Semantic–geometric feature fusion. Qwen3-VL (Bai et al., 2025) and frozen VGGT (Wang et al., 2025a) encode the same views on aligned patch grids. For each view, let V denote semantic patch features, $G^{(L_g)}$ the final-layer geometry features, and c the inferred camera latent. The *geometry fusion connector* adds a camera-conditioned residual:

$$V^+ = V + \mathcal{F}_{\text{geo}}\Big(V, G^{(L_g)}, c\Big). \tag{1}$$

Here $\mathcal{F}_{\text{geo}}$ uses c to modulate the geometry memory and semantic queries to attend to it through cross-attention. The *visual token projector* maps V^+ to language-space tokens.

Visual and geometry DeepStack. We retain the backbone's visual DeepStack (Meng et al., 2024; Bai et al., 2025) and complement it with intermediate VGGT features (Zhang et al., 2026). Separate visual and geometric projectors map corresponding patches to the same token layout and hidden dimension. Their outputs are summed and injected at visual-token positions in selected early language decoder layers. Module configurations and layer selections appear in Appendix B.1.

3.2 Spatially Conditioned Action Generation

Scene and language conditioning. A scene tokenizer uses learned queries to resample frozen DINOv2 appearance features (Oquab et al., 2023) and multi-level VGGT features into fixed-length scene tokens S_t. These tokens enter the action stream alongside noisy-action embeddings. A dual-stream mixture-of-transformers (MoT), with separate language and action parameters, couples the action stream to the VLM encoding of (o_t, u) and, when present, r_t. At each layer ℓ, the action attention output is

$$O_a^{(\ell)} = \text{Attn}\Big(Q_a^{(\ell)}, [K_{\text{vlm}}^{(\ell)}; K_a^{(\ell)}], [V_{\text{vlm}}^{(\ell)}; V_a^{(\ell)}]\Big), \tag{2}$$

where Q, K, V are queries, keys, and values, and brackets denote concatenation. Subscripts distinguish the VLM and action streams. The VLM remains causal and does not attend to actions; action-stream attention is bidirectional. Thus, the expert accesses spatial-answer representations at every layer alongside direct scene features.

Flow-based action prediction. The action expert generates chunks through a conditional flow from Gaussian noise (Black et al., 2025). For answer-conditioned inference, the VLM first generates r_t from (o_t, u); the completed context is re-encoded to cache its layer-wise keys and values for action sampling via Euler integration. Direct instruction control omits answer generation. Neither annotated answers nor future observations are required (Appendix B.4).

3.3 Progressive Spatial-to-Action Training

Supervision and objectives. Using the annotations in Section 4, scene-level QA supervises its annotated frame, while accepted task-resolving instruction–answer pairs are associated with current-observation action windows from the same demonstration, preserving the recorded action targets. Episodes without accepted pairs retain their original instructions. Although annotation may consult trajectory endpoints, this action route receives only current observations. Training minimizes

$$\mathcal{L} = \lambda_{\text{text}}\mathcal{L}_{\text{text}} + \lambda_{\text{act}}\mathcal{L}_{\text{act}}, \tag{3}$$

where $\mathcal{L}_{\text{text}}$ is cross-entropy over supervised assistant tokens and $\mathcal{L}_{\text{act}}$ is the mean absolute error between predicted and target flow velocities, averaged over valid action dimensions and timesteps. Each example activates only its available supervision; no geometric reconstruction loss is used. Flow targets, loss weights, and sampling details appear in Appendix B.3.

Stage 1: Spatial understanding enhancement. Spatial QA and general VQA train the language backbone and output head, visual token projectors, and geometry modules using $\mathcal{L}_{\text{text}}$, with the semantic vision encoder frozen.

Stage 2: Action adaptation. Starting from the pretrained action branch, we train language-attention LoRA adapters (Hu et al., 2022), the scene tokenizer, and the action expert, keeping other VLM parameters frozen. *Action-only adaptation* uses $\mathcal{L}_{\text{act}}$; *LoRA-based co-training* then introduces VQA and robotic supervision under the joint objective. Auxiliary trajectory QA provides text supervision, but its answer tokens remain excluded from the action context.

Stage 3: Joint spatial–action fine-tuning. After merging LoRA, we jointly fine-tune the full VLM, scene tokenizer, and action expert on VQA, direct-action examples, and task-linked instruction–answer–action examples. For task-linked examples, teacher-forced spatial answers become visible to the action expert, and action gradients propagate through layer-wise VLM keys and values to update its representations. VGGT and DINOv2 remain frozen throughout.

4 Manipulation-Oriented Spatial Data and Benchmark

Robot manipulation datasets typically provide visual observations, language instructions, and action trajectories, with fine-grained spatial information largely implicit in observations and actions. Object categories alone cannot resolve manipulation-relevant distinctions in relative direction, distance, or ordinal position among same-category objects. We therefore augment existing demonstrations with manipulation-oriented spatial annotations that make geometric relations explicit through spatial question-answering and instruction supervision. Grounding these annotations in demonstrated tasks connects spatial understanding to manipulation target identification and interpretation of task-relevant spatial configurations. We describe the annotation pipeline and data categories below, followed by our SimplerEnv-Spatial benchmark.

4.1 Data Construction Pipeline

As illustrated in Figure 3,Given a robot manipulation trajectory, we first identify scene objects and combine visual observations, depth, and geometric information to recover object-level spatial structure. We organize object attributes and their spatial relations into a temporal scene graph, providing structured evidence for subsequent annotation. Using the actual manipulation target in the demonstration, we construct candidate sets of same-category or task-related objects and generate spatial questions and manipulation instructions from the corresponding spatial relations. We prioritize relations that help disambiguate the demonstrated target or describe task-relevant spatial configurations. Automatic screening and quality checks assess object identity, temporal consistency, and spatial relations to filter out ambiguous or geometrically inconsistent samples. The scene graphs serve as offline annotation evidence rather than additional policy inputs. Appendix A provides implementation details and screening criteria.

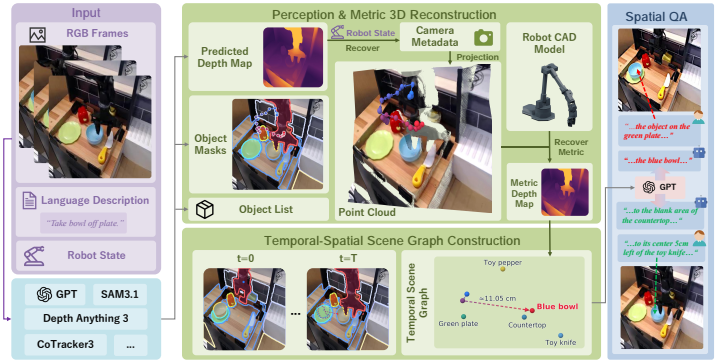


Figure 3: Our spatial data construction pipeline. Robot trajectories undergo metric 3D reconstruction and temporal-spatial scene graph construction to generate spatial QA annotations.

4.2 Dataset Composition

The dataset, **ManiSpatial-565K**, contains two complementary types of QA pairs (Figure 4): **spatial manipulation QA**, resolving spatial constraints in manipulation instructions, and **spatial scene QA**, supporting understanding of scene geometry and target-directed gripper motion.

Spatial manipulation QA. Each question reformulates a demonstrated instruction using spatial constraints, and the answer resolves these constraints into an explicit instruction naming the intended object or destination. The constraints must disambiguate the manipulated object, the placement destination, or both among multiple candidates, while preserving the demonstration's target and goal. They cover *relative relations* (position, distance comparison, and ordinal position), *metric relations* (distance alone or combined with direction), and *spatial reasoning* (imagined viewpoints, composed relations, and hypothetical camera turns).

Spatial scene QA. Each question refers to a specific frame and may concern any clearly visible object, not just the manipulation target. Questions cover the same relative, metric, and reasoning categories, with an additional *action planning* category specifying the direction and distance the gripper must move to reach the target. Most answers are short phrases or sentences, including yes/no judgments. Distance-only questions use a multiple-choice format, while direction–distance and ordinal-position questions use numeric answers.

Appendix A details per-source counts and distributions over skills and answer formats.

4.3 SIMPLERENV-SPATIAL

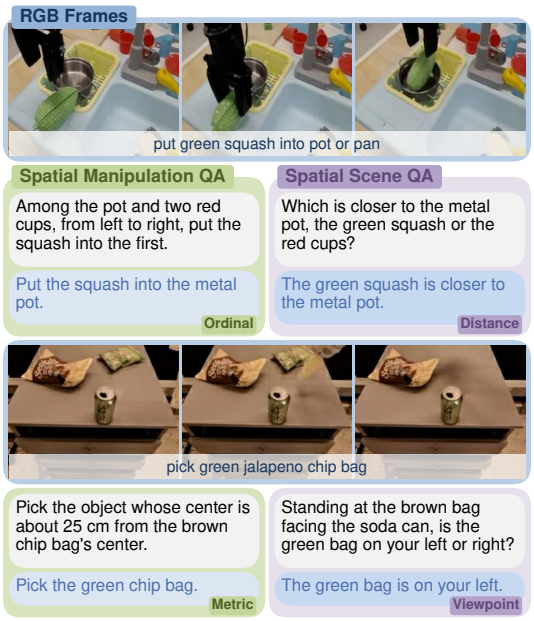


Figure 4: ManiSpatial-565K examples. Each illustrated Bridge or Fractal demonstration includes one spatial manipulation QA pair and one spatial scene QA pair. Tags indicate corresponding skills.

We introduce **SimplerEnv-Spatial**, a Google Robot benchmark built on SimplerEnv (Li et al., 2024b) for manipulation requiring spatial target or destination selection. Each instruction uniquely identifies a target through spatial constraints; removing these constraints leaves multiple valid candidates.

The benchmark comprises 20 tasks following our spatial manipulation annotation taxonomy (Figure 5). **Spatial Referring (SRef; 10 tasks)** covers directional relations, proximity comparisons, and ordinal selection. **Spatial Reasoning (SRsn; 10 tasks)** covers metric-distance constraints, alone or combined with direction, and reference-frame reasoning involving imagined camera turns, alternative viewpoints, or object-defined compass frames.

Instructions are generated from settled simulator states and validated using geometric margins. We evaluate 1,440 trials per model across three seeds, with each task assigned to Visual Matching (VM) or Variant Aggregation (VA). We report task-averaged success per suite and their equally weighted mean. Appendix C provides task definitions, construction details, and evaluation protocols.

5 EXPERIMENTS

Our experiments address three questions: **(Q1)** How well does SpaVLA perform on spatial understanding and manipulation, particularly when spatial relations determine what to manipulate or where to act? **(Q2)** Can it follow spatial instructions under changes in real-world object arrangements, placement locations, and illumination? **(Q3)** How do geometric representations, spatial supervision, and training choices affect these capabilities, and how do spatial answers influence action generation? We examine these questions through benchmark evaluation, real-world experiments, and ablations with answer interventions.

5.1 SPATIAL UNDERSTANDING AND MANIPULATION

Table 1: **Spatial and general multimodal understanding.** In both tables, † denotes baselines adapted using our training data and recipe.

	Spatial understanding										General understanding	
Model	CV-Bench	BLINK	MMSI	MindCube	ViewSpatial	EmbSpatial	SAT	SPAR-Bench	SPBench-SI	SPBench-MV	MMMU	MMBench
Vision–language models												
Qwen3-VL-4B (Bai et al., 2025)	85.8	**63.7**	27.8	28.1	38.7	**79.3**	69.7	35.8	51.0	63.0	49.9	83.5
SenseNova-SI-1.1-8B (Cai et al., 2025)	81.4	61.0	43.6	88.5	54.1	70.4	69.7	46.0	62.6	73.5	50.4	**84.4**
VG-LLM-4B (Zheng et al., 2025a)	80.7	49.2	28.9	44.0	43.5	67.8	58.7	69.7	49.4	59.3	43.4	75.3
SpatialStack-5B (Zhang et al., 2026)	85.8	58.7	28.4	41.4	51.0	76.7	72.3	68.7	59.9	75.6	**55.0**	83.4
RynnBrain1.1-2B (Li et al., 2026b)	86.7	56.5	41.2	61.1	58.4	75.4	70.7	49.7	58.4	**78.6**	44.8	80.7
SpatialEvo-3B (Li et al., 2026a)	76.6	47.7	25.0	49.0	42.1	61.8	65.0	36.6	44.0	54.2	46.2	78.2
Vision–language–action models												
Magma-8B (Yang et al., 2025a)	58.4	44.5	25.9	39.8	35.1	65.4	56.0	24.1	36.9	36.5	35.8	70.1
$\pi^{\dagger}_{0.5}$ (Physical Intelligence et al., 2025)	46.6	37.7	26.5	41.0	32.5	34.8	51.3	38.7	34.2	25.8	25.0	33.1
InstructVLA† (Yang et al., 2025b)	56.8	41.5	31.7	47.2	40.4	54.3	56.0	53.6	44.5	45.5	39.1	64.2
InternVLA-M1† (Chen et al., 2025)	83.1	40.4	31.3	78.6	40.7	67.0	72.0	74.8	54.2	54.5	35.9	64.5
SpaVLA (Ours)	**87.1**	63.1	**48.6**	**95.6**	**59.0**	79.2	**72.7**	**87.2**	**65.2**	65.3	50.9	81.2

Table 2: **Robotic manipulation.** For SimplerEnv-Spatial, ‡ on **Avg.** denotes results from our fine-tuned checkpoints; unmarked results are zero-shot.

	Google Robot								WidowX Robot					SimplerEnv-Spatial		
Model	Pick Coke Can		Move Near		Open/Close Drawer		Avg.		Put Spoon	Put Carrot	Stack Block	Put Eggplant	Avg.	Spatial Referring	Spatial Reasoning	Avg.
	VM	VA	VM	VA	VM	VA	VM	VA	VM							
OpenVLA-7B (Kim et al., 2025)	18.0	60.8	56.3	67.7	**63.0**	28.8	45.8	52.4	4.2	0.0	0.0	12.5	4.2	20.0	18.8	19.4
SpatialVLA-4B (Qu et al., 2025)	81.0	**89.5**	69.6	71.7	59.3	36.2	70.0	65.8	20.8	20.8	25.0	70.8	34.3	21.2	19.0	20.1
Magma-8B (Yang et al., 2025a)	83.7	68.8	65.4	65.7	56.0	53.4	68.4	62.6	37.5	31.0	12.7	60.5	35.4	18.5	19.6	19.1
$\pi_{0.5}$ (Physical Intelligence et al., 2025)	67.7	64.4	80.8	67.0	56.5	24.9	68.3	52.1	49.3	**64.7**	44.7	69.7	57.1	21.2	26.8	24.0‡
InstructVLA† (Yang et al., 2025b)	67.7	68.2	56.7	60.3	23.6	36.5	49.3	55.0	40.3	44.4	13.9	76.4	43.8	27.6	28.2	27.9‡
InternVLA-M1† (Chen et al., 2025)	**97.0**	79.4	93.3	59.3	57.4	**59.3**	82.6	66.0	72.9	60.4	43.8	**91.7**	67.2	38.1	36.4	37.3‡
SpaVLA (Ours)	94.3	85.0	**98.8**	**87.0**	60.2	51.6	**84.4**	**74.5**	**75.0**	54.2	**60.4**	85.4	**68.8**	**68.3**	**62.9**	**65.6**‡

We evaluate spatial and general understanding on the benchmarks in Table 1, and manipulation on SimplerEnv and SimplerEnv-Spatial in Table 2. The comparisons include the original Qwen3-VL-4B backbone, spatially enhanced VLMs, and VLA baselines. VLMs are evaluated using shared question sets and scoring functions, with model-specific image preprocessing and chat templates. Locally adapted baselines are marked in the tables; fine-tuned and zero-shot results on SimplerEnv-Spatial are distinguished separately.

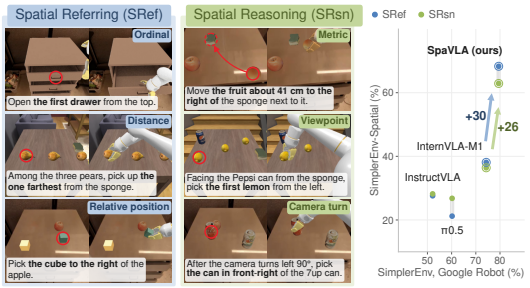


Figure 5: SimplerEnv-Spatial comprises Spatial Referring (SRef) and Spatial Reasoning (SRsn) tasks for spatially constrained manipulation.

Spatial and general understanding. SpaVLA achieves 48.6% on MMSI and 95.6% on MindCube, compared with 27.8% and 28.1% for the original Qwen3-VL backbone. It also obtains 87.1% on CV-Bench and 59.0% on ViewSpatial. The improvements are task-dependent: performance remains close to the backbone on BLINK and EmbSpatial. Beyond spatial tasks, SpaVLA scores 50.9% on MMMU and 81.2% on MMBench, compared with 49.9% and 83.5% for the backbone. Thus, the action-trained model exhibits strong spatial understanding, with a modest trade-off on MMBench rather than uniform gains across all multimodal tasks.

Robotic manipulation. On standard SimplerEnv tasks, SpaVLA achieves average success rates of 84.4% under Visual Matching and 74.5% under Variant Aggregation on Google Robot, together with 68.8% on WidowX. These results show competitive manipulation performance alongside the understanding capabilities evaluated above. However, spatial question answering and standard task completion do not by themselves establish whether a policy follows spatial constraints during execution. We therefore evaluate SimplerEnv-Spatial, where success requires acting on the target or destination specified by a spatial relation. SpaVLA achieves 68.3% on Spatial Referring and 62.9% on Spatial Reasoning, yielding 65.6% overall. These results complement the understanding benchmarks by evaluating whether spatial requirements are satisfied through physical actions.



Figure 6: **Real-world evaluation.** *Left:* a Franka Research 3 with a Robotiq gripper and three Intel RealSense D435i cameras, two external (CAM1, CAM2) and one on the wrist (CAM3); flowers partially occlude the external views. *Middle:* the in-domain task (a) and three out-of-distribution variants: rearranged bread slices (b), relocated plates (c) and changed lighting (d). *Right:* success rates of SpaVLA, $\pi_{0.5}$ and InstructVLA over 18 trials in (a) and (b) and 16 trials in (c) and (d).

5.2 Real-World Generalization

We compare SpaVLA with $\pi_{0.5}$ and InstructVLA on the three-camera Franka Research 3 setup in Figure 6. All models are trained on 1.2 hours of real-robot data (Appendix E). SpaVLA achieves 94.4% in-domain success, compared with 55.6% and 66.7% for the two baselines, respectively. We further evaluate three changes to the scene.

For spatial reference generalization, bread slices are rearranged while flowers partially occlude external views. The requested left-to-right ordinal is defined over the physical arrangement in the robot's reference frame, rather than within each camera image. SpaVLA achieves 83.3% success, versus 44.4% for $\pi_{0.5}$ and 55.6% for InstructVLA. This supports its ability to resolve ordinal references in new layouts from partial observations, rather than merely grasping an arbitrary slice.

For location generalization, plates are displaced by more than 20 cm from their training positions. The instruction still specifies the plate nearer to the robot, requiring both relative destination selection and placement at a new physical location. SpaVLA achieves 68.8% success, whereas both baselines score 0.0%. This contrast highlights generalization beyond familiar placement endpoints: the spatial criterion remains unchanged, but its execution must adapt to the current scene.

Under changed illumination, SpaVLA achieves 56.3% success, exceeding $\pi_{0.5}$ at 31.3% and InstructVLA at 25.0%. Although SpaVLA leads in all evaluated settings, the decline under lighting changes indicates that visual robustness remains a limitation alongside its stronger performance on spatial layout generalization.

5.3 Ablation Studies

We examine how geometry, spatial supervision, and the training schedule affect understanding and manipulation, and whether spatial answers guide action selection.

Geometric representations. Under matched Stage 1 training budgets, adding geometry raises mean accuracy over 11 spatial benchmarks from 64.4% to 67.4% (Table 3(a)), including gains on MMSI (37.8% to 43.8%) and MindCube (91.6% to 94.8%). The improvement is not uniform: CV-Bench remains comparable, whereas SPAR-Bench decreases from 77.7% to 74.4%. These results support geometry as a complement to spatial QA supervision, with benefits that vary across benchmarks. Full results and checkpoint comparisons appear in Appendix Table 10.

Spatial supervision and training data. Table 3(b) compares Full with a branch without our scene-level spatial QA and a Robot-only reference. Full achieves 65.6% on SimplerEnv-Spatial versus 43.6% without scene-level QA, despite similar MMSI accuracy (48.6% versus 47.9%). Robot-only reaches 80.6% on Google Robot VM but only 30.7% on SimplerEnv-Spatial. Strong spatial QA or standard manipulation performance alone thus does not establish spatial instruction following. Since

Table 3: **Ablations (%).** Comparisons are within blocks: (b) compares training recipes, while (c,d) use 4k updates. *Direct-action inference on SimplerEnv-Spatial.

	Manipulation			Understanding	
	VM	VA	Spatial	MMSI	MindCube
(a) Geometry, Stage 1					
w/o geometry	–	–	–	37.8	91.6
w/ geometry	–	–	–	**43.8**	**94.8**
(b) Spatial supervision					
Robot-only	80.6	72.9	30.7*	26.0	36.8
w/o scene QA	78.5	72.4	43.6*	47.9	95.3
Full	**84.4**	**74.5**	**65.6**	**48.6**	**95.6**
(c) Training schedule					
Direct full	63.4	63.3	**46.0**	44.4	94.3
LoRA 400 → full	**69.1**	**66.6**	43.5	**46.0**	**95.0**
(d) Answer access					
w/o answer	58.5	57.8	37.6	**47.7**	92.5
w/ answer	**62.0**	**62.7**	**42.5**	47.5	**94.9**

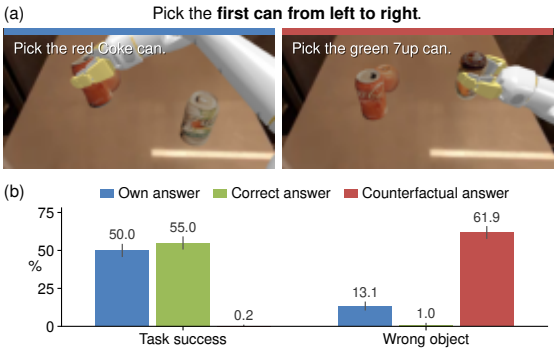


Figure 7: **Spatial answers guide target selection.** First-frame answer interventions on a fixed 4k checkpoint (504 paired episodes per condition). Error bars indicate 95% confidence intervals.

the branches also differ in training history, robot-data mixture, and inference mode, this comparison concerns training recipes rather than the isolated effect of scene-level QA.

Training schedule. From the same action-adapted checkpoint and with identical data, we compare direct full fine-tuning with 400 LoRA co-training updates followed by 3,600 full-VLM updates, using one run per schedule (Table 3(c)). The short LoRA phase improves Google Robot VM from 63.4% to 69.1% and VA from 63.3% to 66.6%, with comparable spatial understanding. However, direct full-VLM performs better on SimplerEnv-Spatial (46.0% versus 43.5%). Thus, delaying full unfreezing benefits standard manipulation at this 4k budget, but not all task settings.

Spatial-answer conditioning. To separate answer access from QA supervision, we retrain a variant whose action expert cannot read answer representations while retaining the same QA supervision (Table 3(d)). At 4k updates, answer access improves all four manipulation benchmarks, including SimplerEnv-Spatial (42.5% versus 37.6%) and WidowX (34.4% versus 26.0%). The gain is larger on SRef than SRsn, while MMSI remains similar (47.5% versus 47.7%). This supports providing spatial answers to the action expert, rather than using them only as language supervision, within the evaluated training budget.

We further replace only the same checkpoint's first-frame answer, holding the initial scene, instruction, and seed fixed across seven tasks and three seeds (504 paired episodes per condition; Figure 7). Candidate objects are visually distinct. Replacement answers are derived from environment records, automatically validated, and specify either the correct target or another candidate. Correct answers increase task success from 50.0% to 55.0% (McNemar's test, $p = 0.001$); among the 48 episodes whose original answer misidentified the target, success rises from 0.0% to 33.3%. Counterfactual answers reduce true-target success to 0.2% and lead the robot to manipulate the named alternative in 61.9% of episodes. Under the model's own answers, the overall non-target manipulation rate is 13.1%. Together, error correction and target-specific redirection show that answer content participates in action selection rather than merely accompanying execution. Correct-answer results are diagnostic, not end-to-end scores; additional breakdowns appear in Appendix Table 11.

6 Conclusion

We presented **SpaVLA**, which integrates 3D geometric features with visual–semantic representations and conditions action generation on explicit spatial answers. Progressive training connects spatial understanding with robotic control, supported by the **ManiSpatial-565K** dataset. We also introduced **SimplerEnv-Spatial** to evaluate spatially constrained manipulation. Experiments demonstrate strong spatial understanding, competitive manipulation performance, and generalization to new real-world layouts, while answer interventions show that spatial answers actively influence action selection.

AI USE STATEMENT

We used generative AI tools mainly to improve the presentation of the manuscript, including language polishing and checks for possible inconsistencies in wording, equations, and reported numerical comparisons. All AI-assisted outputs were carefully examined and, when necessary, modified by the authors before being included in the paper. The authors remain fully responsible for the accuracy, validity, and final presentation of all content, analyses, claims, and experimental results.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-VL Technical Report, 2025. URL https://arxiv.org/abs/2511.21631.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. In *Proceedings of Robotics: Science and Systems*, 2025. URL https://arxiv.org/abs/2410.24164.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael S Ryoo, Grecia Salazar, Pannag R Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan H Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-1: Robotics transformer for real-world control at scale. In *Proceedings of Robotics: Science and Systems*, 2023. doi: 10.15607/RSS.2023.XIX.025. URL https://www.roboticsproceedings.org/rss19/p025.html.

Zhongang Cai, Ruisi Wang, Chenyang Gu, Fanyi Pu, Junxiang Xu, Yubo Wang, Wanqi Yin, Zhitao Yang, Chen Wei, Qingping Sun, Tongxi Zhou, Jiaqi Li, Hui En Pang, Oscar Qian, Yukun Wei, Zhiqian Lin, Xuanke Shi, Kewang Deng, Xiaoyang Han, Zukai Chen, Xiangyu Fan, Hanming Deng, Lewei Lu, Liang Pan, Bo Li, Ziwei Liu, Quan Wang, Dahua Lin, and Lei Yang. Scaling Spatial Intelligence with Multimodal Foundation Models, 2025. URL https://arxiv.org/abs/2511.13719.

Boyuan Chen, Zhuo Xu, Sean Kirmani, Brian Ichter, Danny Driess, Pete Florence, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. SpatialVLM: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024a. URL https://arxiv.org/abs/2401.12168.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, and Feng Zhao. Are We on the Right Way for Evaluating Large Vision-Language Models?, 2024b. URL https://arxiv.org/abs/2403.20330v2.

Xinyi Chen, Yilun Chen, Yanwei Fu, Ning Gao, Jiaya Jia, Weiyang Jin, Hao Li, Yao Mu, Jiangmiao Pang, Yu Qiao, Yang Tian, Bin Wang, Bolun Wang, Fangjing Wang, Hanqing Wang, Tai Wang, Ziqin Wang, Xueyuan Wei, Chao Wu, Shuai Yang, Jinhui Ye, Junqiu Yu, Jia Zeng, Jingjing Zhang, Jinyu Zhang, Shi Zhang, Feng Zheng, Bowen Zhou, and Yangkun Zhu. InternVLA-M1: A Spatially Guided Vision-Language-Action Framework for Generalist Robot Policy, 2025. URL https://arxiv.org/abs/2510.13778.

An-Chieh Cheng, Hongxu Yin, Yang Fu, Qiushan Guo, Ruihan Yang, Jan Kautz, Xiaolong Wang, and Sifei Liu. SpatialRGPT: Grounded spatial reasoning in vision language models, 2024. URL https://arxiv.org/abs/2406.01584.

Mengfei Du, Binhao Wu, Zejun Li, Xuanjing Huang, and Zhongyu Wei. EmbSpatial-Bench: Benchmarking Spatial Understanding for Embodied Tasks with Large Vision-Language Models, 2024. URL https://arxiv.org/abs/2406.05756v1.

Haodong Duan, Xinyu Fang, Junming Yang, Xiangyu Zhao, Zerun Ma, Yuxuan Qiao, Mo Li, Tianhao Liang, Lin Zhu, Amit Agarwal, Xiaozhe Li, Shengyuan Ding, Jiazi Bu, Ziyu Liu, Zhangyang Qi, Yifei Li, Yuhang Zang, Zhe Chen, Lin Chen, Yuan Liu, Yubo Ma, Hailong Sun, Yifan Zhang, Shiyin Lu, Tack Hwa Wong, Weiyun Wang, Peiheng Zhou, Chaoyou Fu, Junbo Cui, Jixuan Chen, Enxin Song, Song Mao, Junming Lin, Xilin Wei, Jinsong Li, Zeyi Sun, Zhaowei Wang, Zicheng Zhang, Xiaoyi Dong, Junjun He, Pan Zhang, Jiaqi Wang, Dahua Lin, and Kai Chen. VLMEvalKit: An open-source toolkit for evaluating large multi-modality models, 2024. URL https://arxiv.org/abs/2407.11691v5.

Zhiwen Fan, Jian Zhang, Renjie Li, Junge Zhang, Runjin Chen, Hezhen Hu, Kevin Wang, Huaizhi Qu, Shijie Zhou, Dilin Wang, Zhicheng Yan, Hongyu Xu, Justin Theiss, Tianlong Chen, Jiachen Li, Zhengzhong Tu, Zhangyang Wang, and Rakesh Ranjan. VLM-3R: Vision-Language Models Augmented with Instruction-Aligned 3D Reconstruction, 2025. URL https://arxiv.org/abs/2505.20279v5.

Xingyu Fu, Yushi Hu, Bangzheng Li, Yu Feng, Haoyu Wang, Xudong Lin, Dan Roth, Noah A. Smith, Wei-Chiu Ma, and Ranjay Krishna. BLINK: Multimodal Large Language Models Can See but Not Perceive, 2024. URL https://arxiv.org/abs/2404.12390v4.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Helong Huang, Min Cen, Kai Tan, Xingyue Quan, Guowei Huang, and Hong Zhang. GraphCoT-VLA: A 3D spatial-aware reasoning vision-language-action model for robotic manipulation with ambiguous instructions, 2025. URL https://arxiv.org/abs/2508.07650.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, Peter David Fagan, Joey Hejna, Masha Itkina, Marion Lepert, Yecheng Jason Ma, Patrick Tree Miller, Jimmy Wu, Suneel Belkhale, Shivin Dass, Huy Ha, Arhan Jain, Abraham Lee, Youngwoon Lee, Marius Memmel, Sungjae Park, Ilija Radosavovic, Kaiyuan Wang, Albert Zhan, Kevin Black, Cheng Chi, Kyle Beltran Hatch, Shan Lin, Jingpei Lu, Jean Mercat, Abdul Rehman, Pannag R Sanketi, Archit Sharma, Cody Simpson, Quan Vuong, Homer Rich Walke, Blake Wulfe, Ted Xiao, Jonathan Heewon Yang, Arefeh Yavary, Tony Z. Zhao, Christopher Agia, Rohan Baijal, Mateo Guaman Castro, Daphne Chen, Qiuyu Chen, Trinity Chung, Jaimyn Drake, Ethan Paul Foster, Jensen Gao, David Antonio Herrera, Minho Heo, Kyle Hsu, Jiaheng Hu, Donovon Jackson, Charlotte Le, Yunshuang Li, Roy Lin, Zehan Ma, Abhiram Maddukuri, Suvir Mirchandani, Daniel Morton, Tony Nguyen, Abigail O'Neill, Rosario Scalise, Derick Seale, Victor Son, Stephen Tian, Emi Tran, Andrew E. Wang, Yilin Wu, Annie Xie, Jingyun Yang, Patrick Yin, Yunchu Zhang, Osbert Bastani, Glen Berseth, Jeannette Bohg, Ken Goldberg, Abhinav Gupta, Abhishek Gupta, Dinesh Jayaraman, Joseph J Lim, Jitendra Malik, Roberto Martín-Martín, Subramanian Ramamoorthy, Dorsa Sadigh, Shuran Song, Jiajun Wu, Michael C. Yip, Yuke Zhu, Thomas Kollar, Sergey Levine, and Chelsea Finn. DROID: A large-scale in-the-wild robot manipulation dataset. In *Proceedings of Robotics: Science and Systems*, 2024. doi: 10.15607/RSS.2024.XX.120. URL https://www.roboticsproceedings.org/rss20/p120.html.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025. URL https://proceedings.mlr.press/v270/kim25c.html.

Chengmeng Li, Junjie Wen, Yan Peng, Yaxin Peng, Feifei Feng, and Yichen Zhu. PointVLA: Injecting the 3D world into vision-language-action models, 2025a. URL https://arxiv.org/abs/2503.07511.

Dinging Li, Yingxiu Zhao, Xinrui Cheng, Kangheng Lin, Hongbo Peng, Hongxing Li, Zixuan Wang, Yuhong Dai, Haodong Li, Jia Wang, Yukang Shi, Liang Zhao, Jianjian Sun, Zheng Ge,

Xiangyu Zhang, Weiming Lu, Jun Xiao, Yueting Zhuang, and Yongliang Shen. SpatialEvo: Self-Evolving Spatial Intelligence via Deterministic Geometric Environments, 2026a. URL https://arxiv.org/abs/2604.14144.

Dingming Li, Hongxing Li, Zixuan Wang, Yuchen Yan, Hang Zhang, Siqi Chen, Guiyang Hou, Shengpei Jiang, Wenqi Zhang, Yongliang Shen, Weiming Lu, and Yueting Zhuang. ViewSpatial-Bench: Evaluating Multi-perspective Spatial Localization in Vision-Language Models, 2025b. URL https://arxiv.org/abs/2505.21500v2.

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. *arXiv preprint arXiv:2510.12276*, 2025c. doi: 10.48550/arXiv.2510.12276. URL https://arxiv.org/abs/2510.12276.

Hongxing Li, Dingming Li, Zixuan Wang, Yuchen Yan, Hang Wu, Wenqi Zhang, Yongliang Shen, Weiming Lu, Jun Xiao, and Yueting Zhuang. SpatialLadder: Progressive Training for Spatial Reasoning in Vision-Language Models, 2025d. URL https://arxiv.org/abs/2510.08531v1.

Kehan Li, Bohan Hou, Minghao Zhu, Tianyi Zhang, Zesen Cheng, Zhikai Wang, Sicong Leng, Xin Li, Xiao Lin, Biying Yao, Minghua Zeng, Jiangpin Liu, Ronghao Dang, Jiayan Guo, Siteng Huang, Haoyu Zhao, Heng Ping, Yaxi Zhao, Tong Zhao, Kexiang Wang, Tong Lu, Shengke Xue, Jiahao Tang, Yulei Wang, Zejing Wang, Jianwei Gao, Shijian Lu, Chengju Liu, Jianfei Yang, Mingxiu Chen, and Deli Zhao. RynnBrain 1.1: Towards More Capable and Generalizable Embodied Foundation Model, 2026b. URL https://arxiv.org/abs/2607.17977.

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, 2025e. URL https://arxiv.org/abs/2506.07961.

Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, Xiaofan Wang, Bei Liu, Jianlong Fu, Jianmin Bao, Dong Chen, Yuanchun Shi, Jiaolong Yang, and Baining Guo. CogACT: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation, 2024a. URL https://arxiv.org/abs/2411.19650.

Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, et al. Evaluating real-world robot manipulation policies in simulation. *arXiv preprint arXiv:2405.05941*, 2024b.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 34892–34916, 2023a. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/6dcf277ea32ce3288914faf369fe6de0-Abstract-Conference.html.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, Kai Chen, and Dahua Lin. MMBench: Is Your Multi-modal Model an All-around Player?, 2023b. URL https://arxiv.org/abs/2307.06281v5.

Wufei Ma, Haoyu Chen, Guofeng Zhang, Yu-Cheng Chou, Jieneng Chen, Celso M de Melo, and Alan Yuille. 3DSRBench: A Comprehensive 3D Spatial Reasoning Benchmark, 2024. URL https://arxiv.org/abs/2412.07825v4.

Wufei Ma, Yu-Cheng Chou, Qihao Liu, Xingrui Wang, Celso de Melo, Jianwen Xie, and Alan Yuille. SpatialReasoner: Towards explicit and generalizable 3D spatial reasoning, 2025. URL https://arxiv.org/abs/2504.20024.

Lingchen Meng, Jianwei Yang, Rui Tian, Xiyang Dai, Zuxuan Wu, Jianfeng Gao, and Yu-Gang Jiang. Deepstack: Deeply stacking visual tokens is surprisingly simple and effective for lmms. *Advances in Neural Information Processing Systems*, 37:23464–23487, 2024.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025. doi: 10.48550/arXiv.2504.16054. URL https://arxiv.org/abs/2504.16054.

Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Jiayuan Gu, Zhigang Wang, Yan Ding, Bin Zhao, Dong Wang, and Xuelong Li. SpatialVLA: Exploring spatial representations for visual-language-action models. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.011. URL https://www.roboticsproceedings.org/rss21/p011.html.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Arijit Ray, Jiafei Duan, Ellis Brown, Reuben Tan, Dina Bashkirova, Rose Hendrix, Kiana Ehsani, Aniruddha Kembhavi, Bryan A. Plummer, Ranjay Krishna, Kuo-Hao Zeng, and Kate Saenko. SAT: Dynamic Spatial Aptitude Training for Multimodal Language Models, 2024. URL https://arxiv.org/abs/2412.07755v3.

Chan Hee Song, Valts Blukis, Jonathan Tremblay, Stephen Tyree, Yu Su, and Stan Birchfield. RoboSpatial: Teaching spatial understanding to 2D and 3D vision-language models for robotics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025a. URL https://arxiv.org/abs/2411.16537.

Wenxuan Song, Ziyang Zhou, Han Zhao, Jiayi Chen, Pengxiang Ding, Haodong Yan, Yuxin Huang, Feilong Tang, Donglin Wang, and Haoang Li. ReconVLA: Reconstructive vision-language-action model as effective robot perceiver. *arXiv preprint arXiv:2508.10333*, 2025b. doi: 10.48550/arXiv.2508.10333. URL https://arxiv.org/abs/2508.10333.

Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai Charitha Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, Ziteng Wang, Rob Fergus, Yann LeCun, and Saining Xie. Cambrian-1: A Fully Open, Vision-Centric Exploration of Multimodal LLMs, 2024. URL https://arxiv.org/abs/2406.16860v2.

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, Abraham Lee, Kuan Fang, Chelsea Finn, and Sergey Levine. BridgeData V2: A dataset for robot learning at scale. In *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 1723–1736. PMLR, 2023. URL https://proceedings.mlr.press/v229/walke23a.html.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. VGGT: Visual geometry grounded transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025a. URL https://arxiv.org/abs/2503.11651.

Qineng Wang, Baiqiao Yin, Pingyue Zhang, Jianshu Zhang, Kangrui Wang, Zihan Wang, Jieyu Zhang, Keshigeyan Chandrasegaran, Han Liu, Ranjay Krishna, Saining Xie, Jiajun Wu, Li Fei-Fei, and Manling Li. MindCube: Spatial Mental Modeling from Limited Views, 2025b. URL https://arxiv.org/abs/2506.21458v2.

Diankun Wu, Fangfu Liu, Yi-Hsin Hung, and Yueqi Duan. Spatial-MLLM: Boosting MLLM capabilities in visual-based spatial intelligence, 2025. URL https://arxiv.org/abs/2505.23747.

Jianwei Yang, Reuben Tan, Qianhui Wu, Ruijie Zheng, Baolin Peng, Yongyuan Liang, Yu Gu, Mu Cai, Seonghyeon Ye, Joel Jang, Yuquan Deng, Lars Liden, and Jianfeng Gao. Magma: A Foundation Model for Multimodal AI Agents, 2025a. URL https://arxiv.org/abs/2502.13130.

Shuai Yang, Hao Li, Bin Wang, Yilun Chen, Yang Tian, Tai Wang, Hanqing Wang, Feng Zhao, Yiyi Liao, and Jiangmiao Pang. InstructVLA: Vision-Language-Action Instruction Tuning from Understanding to Manipulation, 2025b. URL https://arxiv.org/abs/2507.17520.

Sihan Yang, Runsen Xu, Yiman Xie, Sizhe Yang, Mo Li, Jingli Lin, Chenming Zhu, Xiaochen Chen, Haodong Duan, Xiangyu Yue, Dahua Lin, Tai Wang, and Jiangmiao Pang. MMSI-Bench: A Benchmark for Multi-Image Spatial Intelligence, 2025c. URL https://arxiv.org/abs/2505.23764v3.

Bin Yu, Shijie Lian, Xiaopeng Lin, Zhaolong Shen, Yuliang Wei, Haishan Liu, Changti Wu, Hang Yuan, Bailing Wang, Cong Huang, and Kai Chen. 3D-Mix for VLA: A plug-and-play module for integrating VGGT-based 3D information into vision-language-action models, 2026. URL https://arxiv.org/abs/2603.24393.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, Cong Wei, Botao Yu, Ruibin Yuan, Renliang Sun, Ming Yin, Boyuan Zheng, Zhenzhu Yang, Yibo Liu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. MMMU: A Massive Multi-discipline Multimodal Understanding and Reasoning Benchmark for Expert AGI, 2023. URL https://arxiv.org/abs/2311.16502v4.

Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning, 2024. URL https://arxiv.org/abs/2407.08693.

Jiahui Zhang, Yurui Chen, Yanpeng Zhou, Yueming Xu, Ze Huang, Jilin Mei, Junhui Chen, Yu-Jie Yuan, Xinyue Cai, Guowei Huang, Xingyue Quan, Hang Xu, and Li Zhang. From Flatland to Space: Teaching Vision-Language Models to Perceive and Reason in 3D, 2025a. URL https://arxiv.org/abs/2503.22976v7.

Jian Zhang, Shijie Zhou, Bangya Liu, Achuta Kadambi, and Zhiwen Fan. SpatialStack: Layered geometry-language fusion for 3D VLM spatial reasoning, 2026. URL https://arxiv.org/abs/2603.27437.

Zhengshen Zhang, Hao Li, Yalun Dai, Zhengbang Zhu, Lei Zhou, Chenchen Liu, Dong Wang, Francis E. H. Tay, Sijin Chen, Ziwei Liu, Yuxiao Liu, Xinghang Li, and Pan Zhou. From spatial to actions: Grounding vision-language-action model in spatial foundation priors, 2025b. URL https://arxiv.org/abs/2510.17439.

Duo Zheng, Shijia Huang, Yanyang Li, and Liwei Wang. Learning from videos for 3D world: Enhancing MLLMs with 3D vision geometry priors, 2025a. URL https://arxiv.org/abs/2505.24625.

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, Ya-Qin Zhang, Jiangmiao Pang, Jingjing Liu, Tai Wang, and Xianyuan Zhan. X-VLA: Soft-prompted transformer as scalable cross-embodiment vision-language-action model, 2025b. URL https://arxiv.org/abs/2510.10274.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas,

and Kehang Han. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183. PMLR, 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.

SpaVLA: From Explicit 3D Spatial Understanding to Robotic Manipulation

Supplementary Material

ABSTRACT

This supplementary material complements our main paper by providing additional details on data construction, implementation, and evaluation, together with extended experimental results and analyses, to support the reproducibility and assessment of SpaVLA.

▷ **Sec. A**: We describe the spatial data construction pipeline, including data sources, geometry recovery, spatial question–answer generation, and quality control, and present dataset statistics and representative examples.

▷ **Sec. B**: We provide additional implementation and training details, including the network architecture, stage-wise training configurations, data mixing, loss functions, inference procedures, action execution, and evaluation benchmark descriptions with their original references.

▷ **Sec. C**: We detail the proposed SimplerEnv-Spatial benchmark, covering task definitions, scene construction, training and test splits, success criteria, and per-task evaluation results.

▷ **Sec. D**: We specify the evaluation protocols for public understanding benchmarks and standard SimplerEnv, including baseline adaptations, model checkpoints, and scoring procedures.

▷ **Sec. E**: We describe the real-world experimental setup, including the robotic platform, data collection, and generalization settings, and provide detailed results and trajectory visualizations.

▷ **Sec. F**: We present additional results and analyses, including complete understanding benchmark results, extended ablation studies, training dynamics, answer-intervention experiments, and supplementary reference experiments.

▷ **Sec. G**: We discuss the limitations of SpaVLA with respect to the data, model capabilities, and evaluation coverage.

A DATA CONSTRUCTION AND STATISTICS

This appendix provides the geometry, temporal modeling, and curation details of the offline pipeline summarized in Section 4.

We construct spatial annotation evidence from robot manipulation trajectories by organizing object geometry and interaction events into temporal metric scene graphs. The inputs comprise RGB videos, synchronized robot states, task instructions, and upstream object lists, together with available depth and camera calibration. The pipeline first establishes geometry in the robot-base frame, then estimates object states over time, and finally organizes spatial relations and manipulation events for downstream language annotation. An episode with T sampled frames, indexed by $t \in \{0, \dots, T-1\}$, is represented as

$$\mathcal{G} = \left(\mathcal{V}, \{\mathcal{E}_t\}_{t=0}^{T-1}, \mathcal{O}, \mathcal{D}, \mathcal{P}\right), \tag{4}$$

where $\mathcal{V}$ contains retained physical instances, $\mathcal{E}_t$ contains their metric relations at time t, $\mathcal{O}$ records occlusion evidence, $\mathcal{D}$ records deformation events, and $\mathcal{P}$ stores provenance, calibration status, and quality information. Instance identity, visibility, motion, and geometric reliability are represented separately, so that temporary observation loss does not automatically erase an object's state or imply physical motion.

A.1 ROBOT-GROUNDED METRIC GEOMETRY

Coordinate alignment. Let $T_{BC} = [R_{BC}, \mathbf{t}_{BC}]$ transform coordinates from camera frame C into robot-base frame B, with rotation R_{BC} and translation $\mathbf{t}_{BC}$. Let K be the camera intrinsic matrix

for the actual image grid. For pixel coordinates (u, v), the homogeneous pixel $\mathbf{u} = (u, v, 1)^\top$ with optical-axis depth z is back-projected to the base-frame point $\mathbf{p}_B$ as

$$\mathbf{p}_B = R_{BC}\big(zK^{-1}\mathbf{u}\big) + \mathbf{t}_{BC}. \tag{5}$$

Coordinates are expressed in meters, with $+X$, $+Y$, and $+Z$ denoting forward, left, and up in the pipeline's robot-base convention. These base-frame directions are distinct from image-relative directions. We use different calibration and depth-processing branches for single-view predicted depth and multi-view stereo depth.

Robot-constrained calibration and depth scale. For Bridge and Fractal, we estimate fixed-camera geometry from image anchors on stable gripper components and synchronized end-effector poses. Multiple perspective-n-point initializations are followed by robust reprojection refinement, accounting for the local offset between an image anchor and the recorded end-effector origin. Intrinsics and temporal offsets are also refined where enabled. Bridge uses an approximate intrinsic profile, whereas Fractal uses multiple focal-length initializations and refines unfixed intrinsics; exact per-episode calibration is not assumed.

After fitting camera geometry, we separately verify the scale of Depth Anything 3 predictions $\widehat{D}_t$ using robot-anchor depths, sufficiently long robot-motion baselines, and observable gripper dimensions from robot CAD assets. The correction has the form

$$D_t = a\widehat{D}_t + b. \tag{6}$$

Here D_t is the corrected depth at frame t, a is the scale factor, and b is an additive depth bias. We prioritize scale-only correction with $b = 0$. The primary branch combines compatible pose- and CAD-based scale estimates, while a separately checked pose–trajectory branch handles insufficient CAD support. A nonzero bias is allowed only with sufficient depth variation and improved blocked-validation performance. Verification removes each held-out frame from all participating scale-evidence branches. Outputs distinguish corrected depth, verified native scale, unverified native scale, and geometry failure; the checks do not establish a scene-wide reconstruction-error bound.

Stereo-depth screening. For DROID, we use stereo depth recovered offline from the original SVO recordings. Camera-serial-specific intrinsics are paired with separately provided camera-to-base calibration. We first validate episode, camera, and source-frame correspondence, then project the recorded end-effector reference point into each fixed view and compare its camera-axis depth with the observed depth at the projected pixel. Each of the two fixed cameras must provide at least three in-bounds test frames, with strictly more than one third having valid depth and a residual below $5\,\mathrm{cm}$. Invalid depth does not count as an inlier. This eligibility test does not involve the wrist camera or re-estimate a global depth scale.

Instance geometry and cross-view consistency. SAM 3.1 supplies the main instance observations using prompts derived from task instructions and object lists. DROID additionally uses RoboInter contact, box, and human-reviewed mask annotations after identity and temporal alignment checks. Within each instance mask, we filter depth contamination and estimate object centers and bounding geometry in the base frame. Predicted-depth data use percentile trimming, connectivity, and statistical outlier filtering; DROID uses robust depth and radial filtering followed by quantile-based oriented bounding boxes. These estimates describe observed geometry rather than complete object meshes or centers of mass.

DROID instances are associated one-to-one across fixed views, but geometric fusion additionally requires center agreement and bidirectional RGB-D reprojection support. Low coverage remains unknown, and conflicting views are not averaged. Static support surfaces use plane and overlap checks to accommodate different visible portions. Wrist-camera poses follow the end-effector trajectory through a trajectory-specific hand–eye transform; wrist geometry fills missing fixed-view positions only after co-visible observations establish identity and geometric consistency. Each camera retains its own visible masks.

A.2 Visibility-Aware Temporal Scene Graphs

Figure 8 shows the resulting scene graphs on Bridge and Fractal demonstrations.

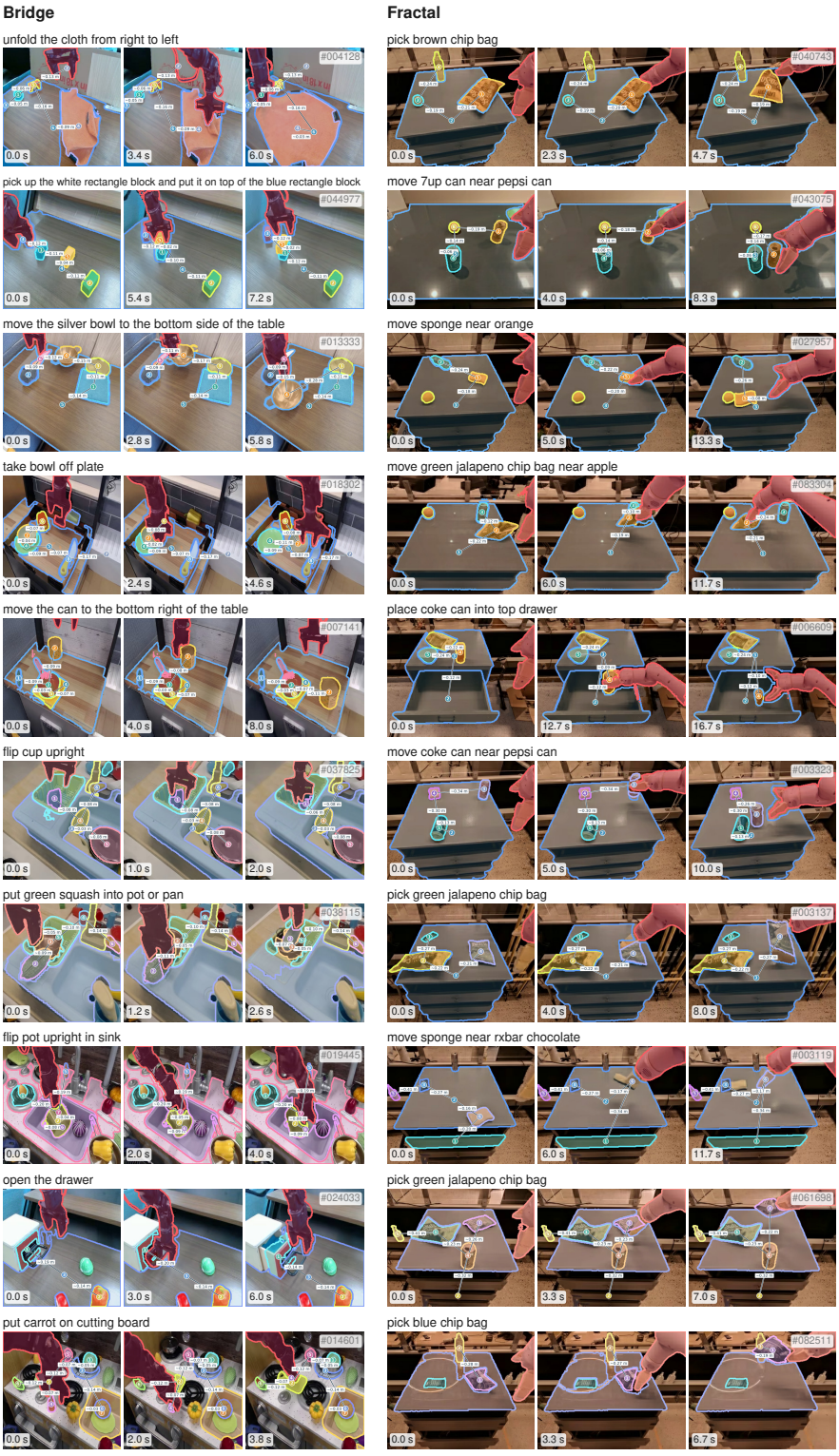


Figure 8: Scene graphs across time on 10 Bridge (left) and 10 Fractal (right) demonstrations, three frames each. Colors and IDs identify objects within an episode; edges follow the first-frame graph and are labeled with estimated center-to-center distances. Distances are approximate, and the masks include previously curated keyframes.

Motion and interaction evidence. The pipeline uses a configurable sparse-motion prior to distinguish static context from the robot and a small set of manipulated objects. Interaction selection combines instruction compatibility, end-effector proximity, gripper-state changes, mask variation, and available motion evidence. When point tracks are available, background-based affine alignment compensates for image motion, and robot-affected points are excluded from object-motion estimation. Activity intervals use separate entry and exit thresholds. Object-list targets and action-specific rules further constrain selection, including interactions such as pressing a button that need not produce substantial object-center displacement.

Static consensus and visibility. For objects confirmed as static, we cluster relatively unoccluded masks around a mask medoid. For instance i in a fixed view, the temporal support $P_i(\mathbf{u})$ is the fraction of consistent observations whose masks contain pixel $\mathbf{u}$:

$$P_i(\mathbf{u}) = \frac{1}{|\mathcal{C}_i|} \sum_{t \in \mathcal{C}_i} \mathbf{1}[\mathbf{u} \in M_{i,t}], \tag{7}$$

Here $M_{i,t}$ is the observed mask of instance i at frame t, $\mathcal{C}_i$ is its set of consistent frame indices, and $\mathbf{1}[\cdot]$ is the indicator function. A majority-supported core is extended only by connected lower-support regions to form canonical support $\overline{M}_i$. Robust aggregation of reliable observations provides a static center, extent, and selected orientation. These geometric states are retained during occlusion rather than recomputed from every partial mask. Current visible support $M_{i,t}^{\text{vis}}$ is estimated by

$$M_{i,t}^{\text{vis}} = \overline{M}_i \setminus \text{dilate}(M_t^{\text{robot}}), \tag{8}$$

Here M_t^{robot} is the robot mask and dilate denotes morphological dilation. The estimate is subject to compatibility with the direct observation. The pipeline distinguishes partial and full occlusion from out-of-view and tracking-loss states. Canonical support is an aggregation of observed regions, not an amodal ground-truth mask; consensus-derived masks retain their own provenance. Object–object occlusion edges are stored as hypotheses with unverified depth ordering.

Dynamic states and manipulation events. Dynamic objects retain frame-specific geometry. Missing observations are completed only through validated short-gap interpolation, trajectory models, or an end-effector-relative carry model supported by contact and grasp intervals. Direct observations and completed geometry remain distinguishable.

For potentially deformable objects, we combine material/action priors, interaction evidence, and temporal changes in eligible point tracks, mask shape, and bounding extents. Strongly robot-occluded frames are excluded from shape statistics. Within supported intervals, source and target masks are aligned by two-dimensional translation and rotation without scaling. Contact-local support, with contour differences as a fallback, identifies candidate active regions; their registered common support provides a relatively stable region. Regions with sufficient pixel and depth support receive base-frame geometric summaries. The output distinguishes localized deformation, lower-confidence whole-object deformation, and insufficient evidence. These are region-level shape-change hypotheses, not dense material correspondences or physical strain measurements.

Metric spatial relations. For retained instances i and j with base-frame centers $\mathbf{c}_{i,t}$ and $\mathbf{c}_{j,t}$ at frame t, we compute their relative displacement $\Delta_{ij,t}$ and Euclidean center-to-center distance $d_{ij,t}$:

$$\Delta_{ij,t} = \mathbf{c}_{i,t} - \mathbf{c}_{j,t}, \qquad d_{ij,t} = \|\Delta_{ij,t}\|_2. \tag{9}$$

For coordinate axis $k \in \{X, Y, Z\}$, a directional label is assigned only when the corresponding signed displacement exceeds the threshold $\delta_{ij,k}$:

$$\delta_{ij,k} = \max\left(0.03\,\text{m}, 2\sqrt{\sigma_{i,k}^2 + \sigma_{j,k}^2}\right), \tag{10}$$

where $\sigma_{i,k}$ describes the geometric spread of instance i along axis k, or its heuristic proxy for dynamic states, rather than a calibrated posterior uncertainty. Static relations are stored once, while relations involving moving instances retain time-indexed samples and compressed temporal segments. Necessary metric relations must be available for retained instances, including manipulation targets and instruction-required reference objects. This completeness criterion does not imply exhaustive detection of every object in the video.

A.3 Spatial Annotation Interface

Structured evidence for language annotation. The resulting graphs provide an interface for generating spatial instructions and questions concerning object relations, temporal changes, and manipulated regions. The interface exposes instance identities and roles, qualified relation segments, source/target states, and supported interaction events, together with their provenance and limitations. Selected RGB and mask keyframes can supplement ambiguous identities or local-region descriptions. The intended language-generation contract retains unknown quantities as unknown rather than inventing metric distances, and preserves distinctions between direct observations, temporal consensus, and predicted geometry. Structured records are stored in Scene Graph V3 JSON, with pixel-level mask evidence maintained separately in NPZ files.

A.4 Spatial QA Generation

Generator input. For each demonstration, GPT-5.6 receives one storyboard image with the first, middle and last frames, each as raw RGB beside its SAM instance map (three views per row for DROID), and a text summary of the scene graph: object tags and names, the demonstrated manipulated object, and metric candidates, i.e. center-to-center distances between retained instances with uncertainty intervals and a computed direction prior. References to the manipulated object are grounded in the first frame and references to its destination in the last frame; the middle frame only helps track identities. One request per demonstration produces both QA types.

Task planning. A rule-based planner assigns each demonstration's task slots from its scene graph, aiming at about six spatial manipulation QA (preferably two relative, three metric and one reasoning) and six spatial scene QA (three relative, two metric and one reasoning), with reserve slots in case the model refuses one. Distance candidates are offered only for instances that pass segmentation-confidence, visibility, robot-occlusion and geometry-confidence gates, and a distance disambiguates two candidates only when their uncertainty intervals, $\pm \max(0.15\,d,\ u_{95,i} + u_{95,j})$, do not overlap. When the evidence is insufficient, fewer QA are scheduled rather than the quota being filled.

Annotation rules. The prompt enforces the following rules.

- A spatial manipulation question is an executable instruction whose spatial constraint leaves at least two candidates when removed and exactly one when kept, without revealing the demonstration; the answer names the resolved object or destination.
- Ordinal positions count complete objects in the raw RGB, including missed segmentations, in the stated spatial order rather than in label order.
- Directions follow the camera view (front is farther from the viewer, behind nearer), with support and containment preferred over planar directions; a camera turn rotates every view-dependent direction.
- Distances are approximate center-to-center values in centimeters taken from the 3D centers, never from pixels, and each direction prior is confirmed or corrected against the RGB.
- Scene questions name their frame; multiple-choice answers give only the option letter, with balanced answer positions.

The model may refuse a slot when the evidence does not support a unique answer.

Validation and delivery. Replies are parsed against a fixed JSON schema, and the slots are filled from the accepted QA according to the quota; refused and unused reserve QA remain only in audit records, and delivered text is not rewritten. Action-planning QA come from a separate pass on Fractal demonstrations: their direction and distance are checked against the gripper-to-target displacement recomputed in the robot-base frame, and a QA is kept only when its target is the demonstrated manipulated object. The prompts were developed on sampled demonstrations reviewed by the authors. Figure 9 shows further examples.

A.5 Data Curation and Screening Statistics

Curation and quality status. We validate episode and camera identities, source-frame mappings, required objects, relation endpoints, and temporal numerical consistency. Unrecoverable



Figure 9: Further examples from ManiSpatial-565K on two Bridge and two Fractal demonstrations, two spatial manipulation QA and two spatial scene QA each, none of them shown in Figure 4; the tag gives the skill.

task-irrelevant background candidates may be excluded, but required targets or reference objects are not deleted to obtain acceptance. Geometry-screening eligibility, scene-graph acceptance, depth-scale verification, and semantic accuracy are tracked separately. In particular, an accepted graph can retain unverified native depth, so acceptance alone does not define a scale-verified subset.

For DROID, early screening retains 10,379 of 18,274 eligible input candidates. The selected 8,007 episodes with object lists are reconstructed at 5 Hz from the original 15 Hz recordings and revalidated using explicit source-frame mappings. Five no-action instructions are excluded before depth testing; 7,864 of the remaining 8,002 episodes pass the two-fixed-view check. These are candidates for subsequent scene-graph annotation, not a count of completed graphs or final QA pairs.

A.6 Language-Supervision Alignment and Validation

Spatial supervision is constructed from manipulation trajectories, original task instructions, and offline object-centric scene annotations. Task-resolving pairs use relative object references, distance-based descriptions, or composed spatial constraints to specify an intended manipulation target; their answers express the resolved operation. Scene-level QA covers object relations, distances, and multi-step scene reasoning. View-relative descriptions follow the reference view used by the annotation, while approximate metric labels retain their units and uncertainty metadata. These scene annotations are construction-time information, not additional runtime policy inputs.

For task-resolving pairs, validation checks episode identity, required text fields, annotation acceptance status, and complete action-window bounds. Recorded continuous action targets are not changed. Episodes without a usable accepted pair fall back to their original instructions. Initial-frame scene QA is retained only as language supervision for that frame; it is not propagated to later action windows.

Offline annotation may consult pre-action and final observations to identify the manipulated object and intended outcome. The aligned action route supplies only the current observation to the policy. Episode-to-window matching establishes task alignment, not independent geometric reannotation or temporal verification at every frame. Annotation provenance and uncertainty are retained; structural acceptance does not imply exhaustive human verification.

A.7 Dataset Statistics

ManiSpatial-565K annotates 54,000 demonstrations from Bridge V2 (Walke et al., 2023), Fractal (Brohan et al., 2023) and DROID (Khazatsky et al., 2024) with 565,305 QA pairs, about ten per demonstration: 251,057 spatial manipulation QA and 314,248 spatial scene QA (Table 4). Table 5 and Figure 10 break the QA down by skill and answer format. Spatial manipulation QA

always answers with a rewritten instruction; spatial scene QA answers with a short phrase or sentence (including yes/no judgments), a choice among options (distance-only questions), or a number (direction–distance and ordinal-position questions).

Table 4: Demonstrations and QA pairs per source dataset.

Source	Demonstrations	Manipulation QA	Scene QA	Total
Bridge V2	21,025	93,326	118,363	211,689
Fractal	29,047	148,674	174,871	323,545
DROID	3,928	9,057	21,014	30,071
Total	54,000	251,057	314,248	565,305

Table 5: QA pairs per skill. SimplerEnv-Spatial evaluates the first six skills and adds an object-defined compass frame, which the data does not contain.

Skill	Manipulation QA	Scene QA	Total	Share
Relative position	65,735	83,730	149,465	26.4%
Distance comparison	40,841	54,698	95,539	16.9%
Ordinal position	65,194	45,947	111,141	19.7%
Metric distance	41,795	77,929	119,724	21.2%
Viewpoint	1,703	13,859	15,562	2.8%
Camera turn	35,696	29,611	65,307	11.6%
Relation composition	93	2,787	2,880	0.5%
Action planning	–	5,687	5,687	1.0%
Total	251,057	314,248	565,305	100%

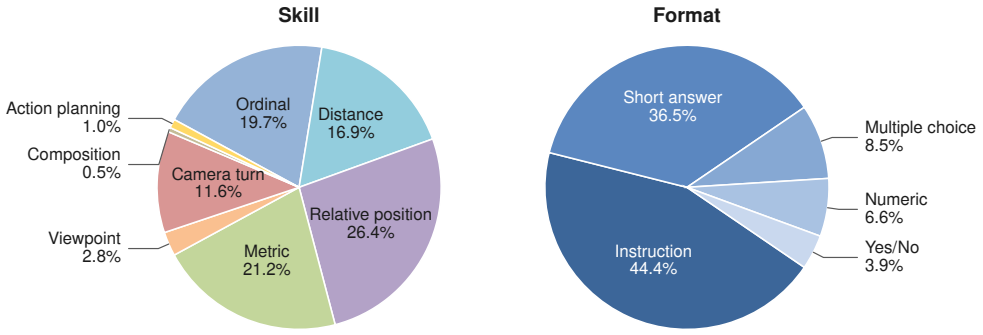


Figure 10: Distribution of ManiSpatial-565K by skill (left) and answer format (right). Instruction: the rewritten instruction of a spatial manipulation QA.

B Implementation and Training Details

B.1 Model Architecture and Input Processing

Model implementation. We instantiate SpaVLA with Qwen3-VL-4B, VGGT-1B, and a DINOv2-L-reg scene backbone. The language decoder has 36 layers with hidden dimension 2,560, while the semantic vision encoder has 24 blocks with hidden dimension 1,024. VGGT supplies 2,048-dimensional patch features and camera latents.

Input alignment and preprocessing. The semantic and geometry image streams share an even patch grid but use patch sizes of 16 and 14, respectively. For example, a square image becomes 576×576 in the semantic stream and 504×504 in the VGGT stream; non-square images use aspect-ratio-dependent grid heights. The existing per-device bottom-padding convention is retained

before geometry encoding, with valid-grid features extracted afterward. Views belonging to one example are encoded as a single scene by VGGT, without cross-example geometric attention.

Camera-conditioned geometry fusion. The geometry fusion connector uses 16 attention heads with an internal width of 2,048. For one view, let $V \in \mathbb{R}^{P \times d_v}$ denote semantic tokens before the visual token projector, $G^{(g)} \in \mathbb{R}^{P \times d_g}$ the geometry patch features, and $c \in \mathbb{R}^{d_g}$ the camera latent. Here P is the number of patches in the aligned grid, and each ϕ below is a learned token-wise affine projection. The camera-conditioned modulation is broadcast over patches of the corresponding view. The camera latent is not a calibrated pose; its projected token participates only in the fusion memory and is not appended to the language sequence. The cross-attention within $\mathcal{F}_{\text{geo}}$ in Equation 1 is

$$\begin{aligned} Q &= \phi_q(\mathrm{LN}(V)), & C &= \phi_c(c), \\ K &= \phi_k(\widetilde{G}), & U &= \phi_v(\widetilde{G}), \\ Z &= \mathrm{LN}(\phi_o(\mathrm{MHA}(Q, [C; K], [C; U]))) , \end{aligned} \tag{11}$$

Here LN denotes layer normalization, MHA is multi-head attention, $[\cdot;\cdot]$ denotes token concatenation, and $\widetilde{G}$ is the camera-modulated geometry. Q contains semantic queries, K and U are geometric keys and values, C is the projected camera token, and Z is the normalized, projected attention output. The SwiGLU residual branch uses the sigmoid linear unit SiLU and element-wise multiplication $\odot$:

$$\mathcal{R}(Z) = \phi_r(\mathrm{SiLU}(\phi_g(Z)) \odot \phi_u(Z)) . \tag{12}$$

The visual token projector then compresses $V^+ = V + \mathcal{R}(Z)$ into language-width tokens. Zero-initializing ϕ_r preserves the original visual stream when the connector is introduced.

Multi-level geometry injection. Three geometric token projectors map features from VGGT layers 11, 17, and 23 into the language space, augmenting native DeepStack features from vision blocks 5, 11, and 17. Their outputs are injected after language blocks 0, 1, and 2, respectively (all listed layer indices are **zero-based**). For the selected geometry layers, the independent projectors $\mathcal{M}_j$, with $j \in \{1, 2, 3\}$ in the order listed above, use

$$\mathcal{M}_j(x) = \phi_2^{(j)}\Big(\mathrm{GELU}(\phi_1^{(j)}(x))\Big) . \tag{13}$$

Here x concatenates the features in a 2×2 patch neighborhood after layer normalization, and GELU denotes the Gaussian error linear unit. Their outputs supplement the native semantic DeepStack features and are injected only at visual-token positions. The final projection $\phi_2^{(j)}$ of each geometric token projector is zero-initialized. The policy uses geometry-encoder features without an additional depth, point-cloud reconstruction, or camera-pose supervision objective.

Scene tokenizer and action expert. The LingBot-style action expert has $L_d = 36$ layers, hidden dimension 768, and feed-forward dimension 2,304. Its scene tokenizer uses 64 learned queries and four resampler blocks. The scene queries resample DINO appearance features and a learned softmax-weighted combination of projected multi-level VGGT features. Each resampler block applies attention to appearance memory, gated attention to geometry memory, and a feed-forward update. All scene tokens are computed from views belonging to the same example. We predict $H = 16$ actions per chunk, with $d_a = 7$ valid action dimensions padded to $D = 32$, without proprioceptive observations. The zero state vector is projected into a learned constant state token. Flow time τ conditions both the noisy-action embeddings of $X_{t,\tau}$ and the action blocks.

Action context and attention. The action stream attends to its own tokens and the VLM's layer-wise keys and values, as defined in Equation 2. Attention is bidirectional within the action stream, while the VLM retains causal attention and does not attend to action tokens. In the answer-conditioned route, spatial answers influence action prediction through their contextualized layer-wise representations. Direct-action examples omit assistant answers while retaining the same expert and scene-token pathway.

B.2 Training Data and Sampling

The offline annotation pipeline and validation procedures are described in Appendix A; here we specify training-time data usage.

Trajectory-aligned annotations and sampling. The annotation-bearing group contains 29,047 Fractal episodes; another 58,165 episodes supply original-instruction supervision. Together they yield 2,405,168 distinct complete $H = 16$ action windows. We assign sampling weights of three and one to windows from the annotated and remaining groups, respectively. Within the annotated group, each exposure uses the original instruction with probability 0.2 or a task-resolving spatial pair with probability 0.8, falling back to the original instruction when the pair is unavailable. Thus, the 80% rate applies only within that group, not to all robot samples.

Language-only supervision and scheduled exposures. Separately, 169,184 unique initial-frame scene QA entries provide language-only supervision. After mixture expansion and batch alignment, the full-VLM schedule contains 12,811,968 VQA exposures and 4,270,656 robot exposures; these are one-pass target counts rather than the number already consumed by an intermediate checkpoint.

B.3 Optimization and Training Schedule

Trainable modules and answer visibility. Stage 1 updates the language backbone and head, visual token projectors, and geometry fusion modules while keeping the semantic vision backbone and VGGT frozen. The principal action-only Fractal and LoRA co-training runs update the language attention adapters, scene tokenizer, and action expert, with other VLM parameters and the external VGGT/DINO backbones frozen. In LoRA co-training, auxiliary two-frame QA answers receive language supervision but are masked from the action context. Full-VLM joint fine-tuning merges the adapters and unfreezes the language components, semantic vision encoder, visual token projectors, and geometry fusion modules alongside the action branch; VGGT and DINO remain frozen. Task-resolving answers are teacher-forced and visible to the action expert in this phase. Language and action losses share the same VLM computation, and its layer-wise keys and values remain attached to the computation graph so that action gradients update trainable VLM parameters. Answers are not autoregressively sampled before training-time action prediction.

Optimization. Table 6 summarizes the principal training runs. LoRA is applied to the query, key, value, and output projections of all language attention layers, with rank 128, scaling parameter 256, and dropout 0.05; no vision or language-MLP adapters are used. Co-training uses AdamW with moment-decay coefficients $(\beta_1, \beta_2) = (0.9, 0.95)$, numerical-stability constant $\epsilon_{\text{opt}} = 10^{-8}$, gradient clipping at 1.0, action-branch weight decay 0.01, and VLM weight decay zero. We use cosine learning-rate decay, a 1% warmup ratio, BF16, and ZeRO-1 in these joint runs. The documented spatial-pretraining run instead uses a 3% warmup ratio, $\beta_2 = 0.999$, and weight decay 0.01.

Flow-velocity loss and noise sampling. The action objective supervises flow velocity, not the final integrated action chunk. For horizon H and padded action dimension D, let m_d equal 1 for valid control dimensions and 0 for padding. The action objective defined in Section 3.3 expands to

$$\mathcal{L}_{\text{act}} = \mathbb{E}\left[\frac{\sum_{h=1}^{H}\sum_{d=1}^{D} m_d \left|v_\theta(X_{t,\tau}, \tau)_{h,d} - (\epsilon - \bar{A}_t)_{h,d}\right|}{H\sum_{d=1}^{D} m_d}\right], \tag{14}$$

Here $\bar{A}_t$ is a normalized, padded action chunk; h indexes actions within the chunk and d indexes padded action dimensions. As in Section 3.2, dependence on the scene tokens and visible VLM context is implicit. The expectation is over action examples, Gaussian noise, and flow time. Padded dimensions do not contribute to the loss. Text loss is normalized over supervised assistant tokens, whereas action loss is averaged over examples with valid action targets. VQA-only examples incur no action loss, and direct-action examples without assistant targets incur no language loss. For action training, we draw $s \sim \text{Beta}(1.5, 1.0)$ from a beta distribution and set $\tau = 0.999s + 0.001$. Each action example uses one independent noise/time draw in the final action-only run and four in co-training, sharing its visual encoding across draws. We set $\lambda_{\text{act}} = 1$ for action training and $\lambda_{\text{text}} = 0.05$ and 0.1 for LoRA and full-VLM co-training, respectively.

Table 6: Principal training settings. Batch denotes the global batch size. Steps refer to the selected local checkpoint in each run, not the total historical budget. "Visual/fusion LR" applies only to visual and geometric token projectors and geometry fusion connectors in Stage 1, and also to the semantic vision backbone in full-VLM fine-tuning. Language learning rates in LoRA runs apply to adapters. A dash denotes a frozen or absent parameter group.

Run	Steps	Global batch	Language LR	Visual/fusion LR	Action LR	λ_{text}
Stage 1: spatial VLM	6,000	384	2×10^{-7}	2×10^{-6}	—	1
Stage 2.1: Fractal action	43,140	1,600	1×10^{-5}	—	5×10^{-5}	0
Stage 2.2: LoRA, first run	10,000	320	5×10^{-5}	—	5×10^{-5}	0.05
Stage 2.2: LoRA, continuation	4,000	384	5×10^{-5}	—	5×10^{-5}	0.05
Stage 3: answer-conditioned full	10,000	768	1×10^{-5}	5×10^{-6}	5×10^{-5}	0.1

B.4 Reference Inference

Answer generation and context caching. Direct control conditions the action expert on the current observation and original instruction. For answer-conditioned control, the VLM first generates the spatial answer r_t from the current observations o_t and language input u. The reference helper then re-encodes the completed question–answer context rather than directly transferring the autoregressive generation cache. The re-encoded context's layer-wise keys and values are cached during action sampling. Neither annotated answers nor future observations are required.

Action sampling and execution. The reference sampler starts from $X_{t,1} \sim \mathcal{N}(0, I)$ and uses ten Euler steps from flow time $\tau = 1$ to $\tau = 0$. Action normalization is dataset-specific; sampled chunks are converted into robot commands using the corresponding embodiment's action statistics and execution conventions. The answer-generation frequency and number of actions executed per predicted chunk are determined by the rollout protocol.

B.5 Evaluation Benchmarks

We use complementary benchmarks to assess visual perception, spatial reasoning, general multimodal understanding, and robotic manipulation. This subsection identifies their original sources and evaluation scope; the data splits, scoring rules, and protocol adaptations used in our experiments are specified in Appendix D and Table 8.

Visual perception and embodied spatial understanding. CV-Bench, introduced with Cambrian-1 (Tong et al., 2024), evaluates vision-centric understanding with 2D and 3D questions. BLINK (Fu et al., 2024) covers 14 visual perception tasks, including relative depth, visual correspondence, and multi-view reasoning; its overall score thus measures a broader set of perceptual abilities than spatial reasoning alone. EmbSpatial-Bench (Du et al., 2024) evaluates egocentric spatial relations in embodied scenes, while 3DSRBench (Ma et al., 2024) probes 3D spatial reasoning in natural images, including object height, orientation, location, and relations among multiple objects.

Multi-view and perspective-dependent reasoning. MMSI-Bench (Yang et al., 2025c) tests spatial intelligence from multiple images. MindCube (Wang et al., 2025b) assesses the ability to form a spatial mental model from limited views and reason about layout, perspective, and motion. ViewSpatial-Bench (Li et al., 2025b) evaluates spatial localization across camera-centered and other viewpoints. SPAR-Bench, introduced in Zhang et al. (2025a), covers spatial perception and reasoning with single-view and multi-view inputs. SPBench-SI and SPBench-MV are the single-image and multi-view subsets of the SPBench benchmark released with SpatialLadder (Li et al., 2025d); they share one source reference and are reported separately in our results.

Motion and temporal spatial reasoning. SAT (Ray et al., 2024) studies static and dynamic spatial reasoning, including how object or camera motion changes spatial relations; our evaluation uses its real-image test set. VSTI-Bench, introduced with VLM-3R (Fan et al., 2025), assesses evolving spatial relationships in video. The release and frame-sampling settings used here are given in Appendix D.1.

General multimodal understanding. MMMU (Yue et al., 2023) evaluates multidisciplinary multimodal understanding and reasoning requiring college-level subject knowledge. MMBench (Liu et al., 2023b) evaluates a broad range of multimodal abilities using multiple-choice questions. MM-Star (Chen et al., 2024b) emphasizes questions that require visual evidence and assesses six core multimodal capabilities. These benchmarks complement the spatial evaluations by measuring more general visual and language understanding.

Robotic manipulation. SimplerEnv (Li et al., 2024b) evaluates real-world robot policies in simulation; we report Google Robot and WidowX results using the protocols in Appendix D.2. Our SimplerEnv-Spatial extension evaluates spatially constrained manipulation through Spatial Referring and Spatial Reasoning tasks, defined in Appendix C.

C SimplerEnv-Spatial Details

Task suites. SimplerEnv-Spatial evaluates spatially constrained manipulation on Google Robot in SimplerEnv (Li et al., 2024b). The 20 tasks are divided equally into Spatial Referring (SRef) and Spatial Reasoning (SRsn), following our spatial manipulation annotation taxonomy. SRef contains seven ordinal-reference tasks, two proximity-comparison tasks, and one directional-reference task. These relations specify the object to manipulate, the destination, or the drawer to open. SRsn contains five metric-distance tasks, including combinations with direction, one imagined-camera-turn task, two imagined-viewpoint tasks, and two object-defined compass-frame tasks. For example, an imagined-viewpoint instruction asks the policy to stand conceptually at the sponge, face the Pepsi can, and select the first lemon from left to right in that view. A compass-frame instruction defines north as the direction from the sponge to the Pepsi can and identifies a destination relative to that frame. Table 7 lists all tasks and their visual settings; Figures 11 and 12 show one rollout per task.

Instruction construction and validation. Instructions are generated from simulator states after the scene has settled. Each instruction must resolve exactly one valid candidate; removing its spatial constraint must leave multiple valid candidates. The applicable checks enforce at least 18 px separation between candidates in the image, a 2.5 cm distance gap for nearest/farthest comparisons, and a 3 cm deadband for directional relations. For metric references, the stated distance must be within 1 cm of the target's true distance, while each competitor's distance must differ from the stated value by at least 4 cm. Layouts violating these margins are rejected at reset, and every evaluation episode is reset and checked before policy evaluation. Front and behind are defined in the robot base frame, with front directed away from the robot. An imagined camera turn is an in-place 90° yaw of the observer.

Objects and visual settings. In addition to the original SimplerEnv objects, the benchmark uses pears and lemons, and cubes, eggplants, and carrots transferred from the Bridge scenes. The 3 cm Bridge cubes are scaled to 5 cm for the Google gripper. In the inspected set of 148,674 Fractal spatial-annotation questions, pears, lemons, cubes, eggplants, and carrots occur 1, 0, 0, 0, and 4 times, respectively; twelve tasks involve these objects. These frequencies describe this annotation set only, not absence from all pretraining or robot-training data. Each task uses the VM or VA setting specified in Table 7. VM uses a real-image overlay, while VA uses eight rendered variants involving backgrounds, lighting, table textures, and distractors. Tasks are not each evaluated under both settings.

Evaluation and success criteria. Each task is evaluated on 24 episodes under each of three seeds (20260810, 20260922, and 20261022), yielding 72 trials per task and $20 \times 3 \times 24 = 1{,}440$ trials per model. All models are evaluated on the same seeds and therefore on the same initial layouts and instructions. Each episode runs for at most 120 steps, compared with the original Google-task limits of 80 steps and 113 steps for drawers, and is counted as successful at the first step satisfying its task predicate. For *pick* tasks, the designated target must be grasped and lifted by more than 1 cm without other contact; grasping a non-target object fails the episode. For *move* tasks, the source must move by more than 5 cm, reach within 10 cm of the destination measured surface to surface, and be at least 1 cm closer to that destination than to any other candidate; every other object must move by less than 5 cm. For *drawer* tasks, the designated drawer must open by at least 15 cm while no other drawer moves by more than 3 cm. We first compute each task's success rate over its 72 trials, then

take the unweighted mean over the ten tasks in each suite. The overall score is the equally weighted mean of the two suite scores.

Policy evaluation. Released OpenVLA, SpatialVLA, and Magma checkpoints are evaluated zero-shot with their model-specific SimplerEnv implementations: action scale 1, gripper threshold 0.5, one action executed per control step (SpatialVLA with its released action-chunk ensembling), together with the benchmark's sticky-gripper handling. Locally adapted baselines use SpaVLA's deployment settings: action scale 2.75, gripper threshold 0.8, a 15-step sticky gripper, and execution of the first four actions from each predicted 16-action chunk. SpaVLA generates a spatial answer once, from the first observation, and conditions all subsequent action chunks of the episode on it; InstructVLA also uses answer-before-action inference in its native format. Released and locally adapted results are distinguished in Table 2; the deployment settings above are not assumed to be identical across these two groups.

Table 7: SimplerEnv-Spatial tasks and SpaVLA success rates (%, 72 trials per task: 24 episodes $\times$ 3 seeds). OOD marks objects absent or nearly absent from our spatial annotations.

ID	Task	Objects	Setting	SpaVLA
SRef-1	Move a can near the apple, by order	Coke / 7up cans, apple	VM	98.6
SRef-2	Move the Pepsi can near the n-th lemon	lemons (OOD), Pepsi can	VM	95.8
SRef-3	Pick a can by order	Coke / 7up cans, apple	VM	88.9
SRef-4	Pick the pear closest / farthest to the sponge	pears (OOD), sponge	VA	73.6
SRef-5	Pick the n-th closest pear to the sponge	pears (OOD), sponge	VA	63.9
SRef-6	Move the sponge near the n-th pear	pears (OOD), sponge	VA	68.1
SRef-7	Pick the n-th lemon	lemons (OOD), Pepsi can	VA	69.4
SRef-8	Move a vegetable near the sponge, by order	carrot / eggplant (OOD), sponge	VM	48.6
SRef-9	Pick the cube left / right of the apple	cubes (OOD), apple	VM	47.2
SRef-10	Open the n-th drawer	cabinet	VM	29.2
SRsn-1	Pick a can after an imagined camera turn	Coke / 7up cans, apple	VM	86.1
SRsn-2	Move a can by direction and distance	Coke / 7up cans, apple	VM	75.0
SRsn-3	Pick a can by direction and distance	Coke / 7up cans, apple	VM	72.2
SRsn-4	Move a fruit by direction and distance	orange / apple, sponge	VM	70.8
SRsn-5	Pick a pear from an imagined viewpoint	pears (OOD), sponge, Pepsi can	VA	62.5
SRsn-6	Pick a lemon from an imagined viewpoint	lemons (OOD), sponge, Pepsi can	VA	54.2
SRsn-7	Pick the pear at a given distance	pears (OOD), sponge	VA	54.2
SRsn-8	Move the sponge near a lemon in a compass frame	lemons (OOD), sponge, Pepsi can	VA	55.6
SRsn-9	Pick a pear by direction and distance	pears (OOD), sponge	VA	56.9
SRsn-10	Move the sponge near an apple in a compass frame	apples, sponge, Pepsi can	VA	41.7
SRef / SRsn / Avg.				68.3 / 62.9 / 65.6

D Evaluation Protocols

This section specifies the protocols behind Tables 1 and 2. Benchmark descriptions and original references are collected in Appendix B.5. For each benchmark, all models that we evaluate share the question set or task configuration, prompts, decoding settings, scoring code, and seeds. The only intended difference is the inference mode each model was trained for: whether it generates a spatial answer before acting.

D.1 Understanding Benchmarks

Shared protocol. Each question is posed with a fixed prompt that includes its answer-format instruction, and responses are decoded greedily with at most 512 new tokens. Answers are scored with deterministic rules only; no LLM judge is used. Multiple-choice answers are matched with the option matchers of VLMEvalKit (Duan et al., 2024), numerical answers are scored by mean relative accuracy (MRA), and MMSI-Bench and SAT use the answer extractors released by their authors. SpaVLA and the locally adapted VLAs receive all images before the text, as in training. VLM baselines are evaluated from their released weights with their native chat templates and image preprocessing. On MMMU, only the images referenced by a question are inserted, in order of reference, as in VLMEvalKit. Table 8 lists the split, aggregation, and deviations from the official protocol of each benchmark.

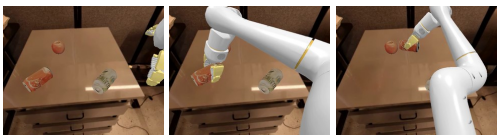
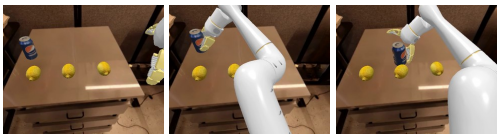
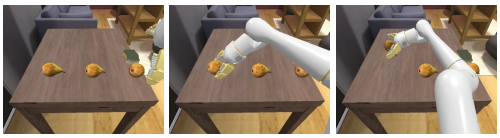
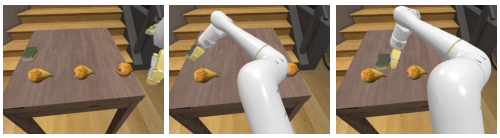
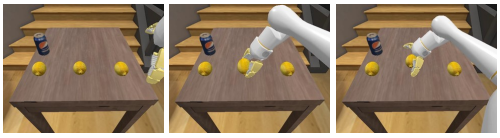
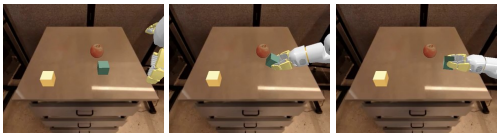
Rollout	Instruction	Response
	Move the first can from left to right among the red Coke can and green 7up can near the apple.	Move the red Coke can near the apple.
	Move the Pepsi can near the second of the three lemons, counting from left to right.	Move the Pepsi can near the middle lemon.
	Pick the second can from left to right among the red Coke can and green 7up can.	Pick the green 7up can.
	Among the three pears, pick up the one farthest from the sponge.	Pick up the left pear.
	Among the three pears, pick up the first closest to the sponge.	Pick up the leftmost pear.
	Move the sponge near the first of the three pears, counting from left to right.	Move the sponge near the front-left pear.
	Among the three lemons, pick up the second one from left to right.	Pick up the middle lemon.
	Move the second vegetable from left to right among the carrot and eggplant near the sponge.	Move the eggplant near the sponge.
	Pick the cube to the right of the apple.	Pick the green cube.
	open the first drawer from the top	Open the top drawer.

Figure 11: **Spatial Referring rollouts.** One successful SpaVLA episode per SRef task: the first episode, in seed and episode order, that succeeds and whose answer names the target. Frames show the first observation, the first successful step (for pick tasks, the target lifted), and the final step, with the instruction and the answer generated from the first observation.

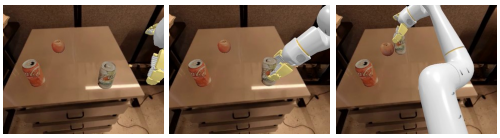
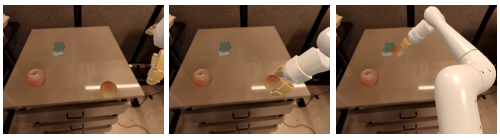
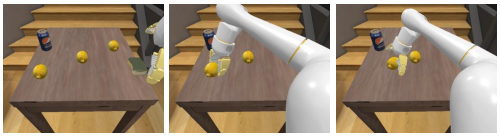
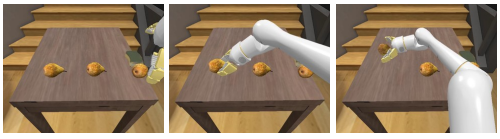
Rollout	Instruction	Response
	After the camera turns left by 90 degrees, pick the can in front-right of the green 7up can.	Pick the red coke can.
	Move the can whose center is to the right of and about 37 cm from the apple's center near the apple.	Move the silver 7up can near the apple.
	Pick the can whose center is to the left of and about 11 cm from the apple's center.	Pick the red soda can.
	Move the fruit whose center is to the right of and about 41 cm from the sponge's center near the sponge.	Move the orange near the sponge.
	Imagine standing at the sponge and facing the Pepsi can. From that viewpoint, pick up the first pear from left to right among the three pears.	Pick up the front-left pear.
	Imagine standing at the sponge and facing the Pepsi can. From that viewpoint, pick up the first lemon from left to right among the three lemons.	Pick up the front-left lemon.
	Pick up the pear about 46 centimeters from the sponge.	Pick up the left pear.
	Treat the direction from the sponge to the Pepsi can as north. Move the sponge near the lemon farthest west among the three lemons in that compass frame.	Move the sponge near the left lemon.
	Pick up the pear to the left of the sponge and about 45 cm away from it.	Pick up the left pear.
	Treat the direction from the sponge to the Pepsi can as north. Move the sponge near the apple farthest east among the three apples in that compass frame.	Move the sponge near the apple farthest east.

Figure 12: **Spatial Reasoning rollouts.** One successful SpaVLA episode per SRsn task, chosen and shown as in Figure 11; failure cases are shown in Figure 15.

Model-specific rules. Two rules differ across models. On MMSI-Bench, SpaVLA answers the original questions only, whereas each baseline receives the better of two prompts: the original question and the question followed by "Answer with the option letter only." Without this instruction, some baselines rarely produce a parsable option, so the rule can only favor the baselines. On MMMU, 10.8% of Qwen3-VL-4B's validation responses exceeded the 512-token limit, because the model often reasoned at length despite the direct-answer instruction. We therefore re-ran Qwen3-VL-4B on MMMU with an 8,192-token limit and report the resulting 49.9%; all responses that had ended within the original limit were unchanged. Every other model reaches the limit on at most two MMMU questions.

Agreement with official protocols. To check that the unified protocol does not favor our model, we re-evaluated SpaVLA under the official protocol of each benchmark: official data files, prompts, generation lengths, answer extraction, and aggregation, without the optional LLM-judge fallback. Scores change by at most 1.3 points on every re-evaluated benchmark except MMStar, where the official data file raises SpaVLA's score by 2.8 points (Table 8). VSTI-Bench is evaluated on its corrected release, so its scores are not directly comparable with results published on the original release.

Table 8: **Understanding benchmarks.** Split and aggregation used for each benchmark, deviations from its official protocol, and SpaVLA's score under our protocol (Ours) and under the official protocol (Official). All benchmarks use greedy decoding and deterministic answer matching. MRA: mean relative accuracy.

Benchmark	Split and aggregation	Deviation from official protocol	Ours	Official
CV-Bench	2,638 questions; 0.25 ADE$+0.25$ COCO$+0.5$ Omni3D	Answer-format wording	87.1	88.4
BLINK	val, 1,901 questions; mean over 14 tasks	Current Hugging Face labels; no disclaimer prompts; no LLM fallback	63.1	63.5
MMSI-Bench	1,000 questions	No answer-format suffix for SpaVLA	48.6	47.5
MindCube	Tiny, 1,050 questions	Answer-format wording	95.6	95.1
ViewSpatial	5,712 questions	Answer-format wording; 512 instead of 128 tokens	59.0	58.4
EmbSpatial	3,640 questions	Prompt wording (no official inference code)	79.2	79.1
SAT	Real, 150 questions $\times$ 2 option orders	None	72.7	72.7
SPAR-Bench	7,211 questions; mean over 20 tasks; MRA for numerical answers	Aggregation of the view-change questions	87.2	87.1
3DSRBench	5,157 circular groups; a group counts only if all variants are correct	Prompt wording	52.1	51.8
SPBench-SI / MV	1,009 / 319 questions; mean over 4 / 5 types	No "Options:" header	65.2 / 65.3	65.1 / 65.1
VSTI-Bench	Corrected release, 5,736 questions, 32 frames; mean over 9 types	Data release, prompt, and averaging	58.5	–
MMBench	EN v1.1 dev, 1,292 circular groups	Answer-letter instruction; no LLM fallback	81.2	80.5
MMStar	1,500 questions	Hugging Face release with inline options; no LLM fallback	58.7	61.5
MMMU	val, 900 questions	Answer-format instruction; no LLM fallback	50.9	49.6

D.2 SIMPLERENV

Google Robot. We evaluate Pick Coke Can, Move Near, and Open/Close Drawer under Visual Matching (VM) and Variant Aggregation (VA) with SIMPLER's episode configurations (Li et al., 2024b): 756 VM episodes (300, 240, 108, and 108 for the four tasks) and 1,803 VA episodes (825, 600, 189, and 189). The reported average is the mean of Pick, Move, and Drawer success, where Drawer averages Open and Close. Published results in Table 2 are recomputed from their per-task numbers in the same way. All models that we evaluate receive the original task instruction and act without generating a spatial answer. As in the official X-VLA evaluation (Zheng et al., 2025b), an episode ends and counts as successful at the first step at which the success condition holds.

WidowX. We evaluate the four Bridge tasks (Put Spoon on Towel, Put Carrot on Plate, Stack Block, and Put Eggplant in Basket) and report task success rather than grasp success. Each task uses SIMPLER's 24 episode layouts under two seeds (20260810 and 20261022). Episodes run for 120 control steps, and success is evaluated at the final step.

Control settings. Table 9 lists the per-platform settings. SpaVLA and all locally adapted baselines share these settings, with two exceptions described in Appendix D.3. Baselines keep their native action samplers: 10 flow steps for $\pi_{0.5}$ and InstructVLA, and 5 DDIM steps with a guidance scale of 1.5 for InternVLA-M1. Appendix C gives the SimplerEnv-Spatial protocol, which uses the Google Robot settings.

Table 9: **SimplerEnv control settings** for SpaVLA and the locally adapted baselines.

	Google Robot	WidowX
Action chunk	16 predicted, first 4 executed	16 predicted, first 4 executed
Action scale (translation, rotation)	2.75 for all six dimensions	(1, 1, 1) and (1.25, 0.9, 1.25)
Gripper	Binarized at 0.8; 15-step sticky gripper	Binarized at 0.65; no sticky gripper
Proprioceptive state	None	None
Success	First step satisfying the condition	Final step
Episodes	756 (VM) and 1,803 (VA)	4 tasks $\times$ 24 layouts $\times$ 2 seeds

D.3 Evaluated Checkpoints and Baselines

SpaVLA. All understanding benchmarks, Google Robot, and SimplerEnv-Spatial use a single checkpoint: the Stage-3 model after 20k updates (Appendix B.3).

Vision–language baselines. We evaluate the released Qwen3-VL-4B-Instruct, SenseNova-SI-1.1 (InternVL3-8B), VG-LLM-4B, SpatialStack-5B, RynnBrain1.1-2B, SpatialEvo-3B, and Magma-8B checkpoints. Inference follows each model's official repository.

Released VLAs. Standard SimplerEnv results for OpenVLA, SpatialVLA, and Magma are taken from published evaluations. OpenVLA's results are those reported by CogACT (Li et al., 2024a), since the OpenVLA paper does not evaluate on SimplerEnv. SpatialVLA's are its zero-shot results (Qu et al., 2025). Magma reports VM and VA jointly (Yang et al., 2025a); we use the VM/VA split reported by InternVLA-M1 (Chen et al., 2025), whose average matches Magma's combined results, and Magma's own WidowX results. Their SimplerEnv-Spatial results are our zero-shot evaluations of the released checkpoints (Appendix C).

Locally adapted VLAs. InstructVLA, InternVLA-M1, and $\pi_{0.5}$ (marked †) are trained with our co-training data and recipe. For InstructVLA and InternVLA-M1, training continues from 10k updates with the spatial manipulation QA supervised in each model's native format: InstructVLA generates the answer before acting, whereas InternVLA-M1 co-trains on the QA without exposing answers to its action module. On SimplerEnv-Spatial, InstructVLA answers before acting as in training, and the other baselines act directly from the instruction. Two manipulation results come from other sources. Table 2 reports a $\pi_{0.5}$ model trained on Fractal and Bridge only; we evaluate its Google Robot and SimplerEnv-Spatial performance, and its WidowX results are taken from reported evaluations. For WidowX, InstructVLA is evaluated from its released generalist checkpoint with its own SimplerEnv evaluation code (7-step action ensembling, action scale 1, no sticky gripper), averaging three runs of 24 episodes per task.

E Real-World Experiments

This section details the setup, data, training, and evaluation behind Section 5.2.

Setup and task. We use a Franka Research 3 arm with a Robotiq gripper and three Intel RealSense D435i cameras: two external views (CAM1, CAM2) and a wrist view (CAM3), all recorded as 640×480 RGB (Figure 6). From left to right, the table holds a cutting board, two bread racks, a vase of flowers, and a third rack; each rack has three slots, which hold a bread slice, a lettuce leaf, or nothing. Two plates and several fruits and vegetables complete the scene, and the flowers hide parts of it from each external camera. Instructions take the form "Pick up the k-th slice of bread from the left and place it on the near/far plate." The ordinal counts all bread slices from left to right in the robot's reference frame, including a slice that may lie on the cutting board and is never a target; near and far refer to the distance from the robot base.

Demonstrations. We collect 1.2 hours of real-robot data: 200 successful demonstrations by SpaceMouse teleoperation, recorded at 10 Hz. They cover 12 arrangements with three to seven slices, and target ordinals from first to seventh (14 distinct instructions); in each arrangement, every slice on a rack is moved twice to each plate. The scene is reset before each demonstration, and the two plates stay at fixed positions. Actions are 7-dimensional: the translation and rotation-vector change of the end effector between consecutive measured poses, in the robot base frame, and a binary gripper command, each normalized to $[-1, 1]$ by its 1st and 99th percentiles. Policies receive the three views and the instruction, without proprioceptive state. We annotate the demonstrations with the pipeline of Appendix A using GPT-5.6 on the RGB views, with directions defined from the robot base facing the workspace; this yields 1,213 spatial manipulation QA and 1,226 spatial scene QA.

Training. SpaVLA starts from the final Stage-3 checkpoint (Appendix **??**) with a new optimizer, and trains the same modules with the same objectives ($\lambda_{\text{text}} = 0.1$) for 25,000 updates at a global batch of 256 on 32 GPUs. Learning rates peak at 1×10^{-5} (language), 5×10^{-6} (visual/fusion), and 5×10^{-5} (action), with 1% warmup and cosine decay. VQA and robot exposures are scheduled at three to one (4.8M and 1.6M). The robot stream covers the 53,398 complete $H = 16$ action windows about 30 times; each exposure uses the task-resolving pair with probability 0.7 and the original instruction otherwise. The VQA stream replays the Stage-3 mixture, with the 111,141 ordinal-position QA of ManiSpatial-565K repeated two to five times, and adds the real-robot scene QA, repeated 30 times. $\pi_{0.5}$ and InstructVLA start from their co-trained checkpoints (Appendix D.3) with new optimizers and train for 25,000 updates at a global batch of 256 on 16 GPUs, using the same images, action normalization, and 16-step chunks, with their native losses and preprocessing. $\pi_{0.5}$ is trained on the demonstrations alone, with their original instructions and actions (6.4M robot exposures). InstructVLA uses the same mixture as SpaVLA: the VQA stream and robot exposures at three to one, with the task-resolving pair used with probability 0.7 and its answer placed before the meta tokens that condition its action module.

Evaluation. Each model predicts 16-step action chunks and executes the first four before predicting again; SpaVLA acts directly from the instruction. We run 18 trials in the in-domain setting and under rearranged slices, and 16 trials with relocated plates and under changed lighting. In the rearranged setting, the slices are placed in arrangements not seen in training while the flowers still occlude the external views. With relocated plates, both plates are moved more than 20 cm from their training positions. A trial succeeds when the requested slice ends on the requested plate. Figure 13 shows example SpaVLA rollouts.

F Additional Results

F.1 Geometric Representations: All Benchmarks

Table 10 extends Table 3(a) to all spatial and general benchmarks. Both models are evaluated at the end of Stage 1, before action adaptation. Besides the final no-geometry checkpoint, we report the best score per benchmark over nine no-geometry checkpoints saved between 25k and 45k updates, so that the comparison does not depend on a single checkpoint of that run. Adding geometry improves eight of the eleven spatial benchmarks by 1.7–7.0 points, and none of the nine no-geometry checkpoints reaches the geometry-enabled model on MMSI, SAT, MindCube, ViewSpatial, SPBench, VSTI, or the spatial average. CV-Bench and EmbSpatial are within one point, SPAR-Bench decreases by 3.3 points, and the general-benchmark average does not decrease.

Table 10: **Geometry ablation: all spatial and general benchmarks at the end of Stage 1** (%). "Best of 9" is the best score per benchmark over nine no-geometry checkpoints (25k–45k updates); bold compares the two end-of-Stage-1 models. Sp. Avg. averages the 11 spatial benchmarks (SPBench as the mean of SI and MV); Gen. Avg. averages MMBench, MMStar, and MMMU.

	MMSI	SAT	BLINK	CV	MindCube	Emb	View	SPAR	3DSR	SPB	VSTI	Sp. Avg.	Gen. Avg.
w/o geometry (45k)	37.8	69.3	60.2	85.5	91.6	77.7	49.2	**77.7**	49.9	59.6	50.0	64.4	64.5
w/o geometry (best of 9)	38.2	70.7	60.9	86.3	92.2	78.8	51.7	77.9	51.3	61.1	52.0	65.2	65.5
w/ geometry	**43.8**	**76.3**	**63.0**	**85.8**	**94.8**	**78.0**	**54.0**	74.4	**51.6**	**63.6**	**56.3**	**67.4**	**66.0**

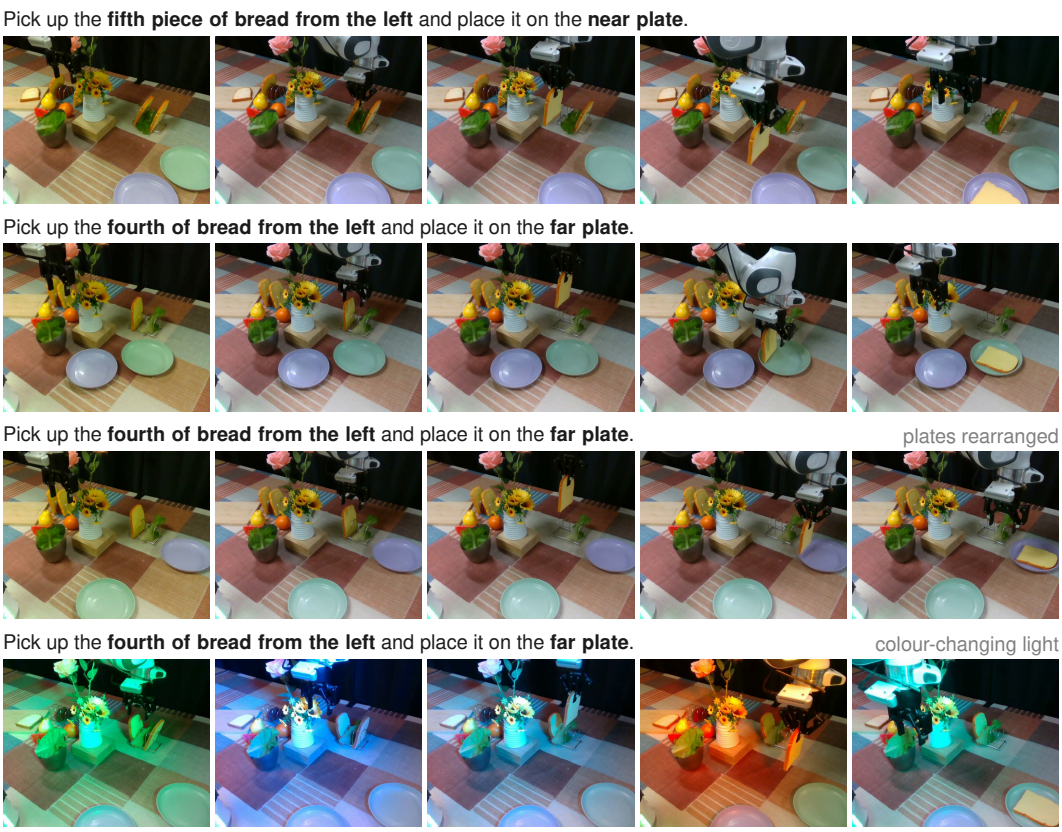


Figure 13: Example SpaVLA rollouts on the real robot, five frames each from an external camera. The instruction above each row is shown as issued, with the target phrases in bold; the last three rows repeat one instruction with the plates rearranged and under colour-changing light.

F.2 Answer Interventions: Breakdown

Table 11 breaks down the interventions of Figure 7 by task. All three conditions run the same 504 episodes (seven tasks, three seeds, 24 episodes each) from the same initial scene, instruction, and seed; only the first-frame answer differs. Replacement answers name the true target (correct answer) or the other candidate object recorded by the environment (counterfactual answer), are phrased in the model's own answer format, and are checked automatically to name exactly the intended object. For each episode, we record which of the two candidates the robot manipulated, using the peak lift for pick tasks (> 2 cm) and the planar displacement for move tasks (> 3 cm).

Counterfactual answers. In every task, true-target success drops to at most one episode, and the robot manipulates the object named by the counterfactual answer in 23.6–95.8% of episodes (61.9% overall, 95% CI 57.6–66.0), compared with 13.1% under the model's own answers. The effect is consistent across seeds: 60.1%, 67.3%, and 58.3% of episodes for the three seeds, with true-target success of at most 0.6%.

Correct answers. Correct answers raise overall success from 50.0% to 55.0%. On the paired episodes, 40 succeed only with the correct answer and 15 only with the model's own answer (exact McNemar test, $p = 0.001$); the gain is positive for each seed (+4.7, +4.7, and +5.4 points). The improvement comes entirely from episodes whose own answer was wrong or ambiguous. When the model's answer named the other object (48 episodes), success rises from 0/48 to 16/48 ($p < 0.001$); when it could not be mapped to a single candidate (66 episodes), from 36.4% to 53.0% ($p = 0.027$). When it already named the target (390 episodes), success is unchanged (58.5% versus 57.9%, $p = 0.81$), suggesting that the remaining failures in these episodes stem mainly from execution rather

than target identification. These correct-answer results are diagnostic and are not part of the end-to-end scores in Section 5.1.

Table 11: **Answer intervention per task** (4k answer-conditioned checkpoint; seeds 20260810, 20260922, and 20261022; 72 episodes per task). Task success is measured for the true target. "Wrong object" is the share of episodes in which the robot manipulated the other candidate object (peak lift >2 cm for pick, planar displacement >3 cm for move); under the counterfactual condition this is the object named by the answer. The last row restricts to episodes in which the model's own answer named the other object.

	Task success			Wrong object	
Task	Own	Correct	Counterf.	Counterf.	Own
Move, direction + metric	37/72	58/72	0/72	69/72	29/72
Move, ordinal (cans)	62/72	65/72	0/72	68/72	9/72
Move, ordinal (vegetables)	23/72	23/72	0/72	40/72	4/72
Pick, direction + metric	37/72	35/72	1/72	38/72	9/72
Pick, ordinal	40/72	40/72	0/72	45/72	0/72
Pick, relative (cubes)	17/72	19/72	0/72	17/72	1/72
Pick, camera rotation	36/72	37/72	0/72	35/72	14/72
All (%)	50.0	55.0	0.2	61.9	13.1
Own answer names the other object	0/48	16/48	0/48	38/48	38/48

F.3 Failure Analysis on SimplerEnv-Spatial

We classify the 495 failed episodes of SpaVLA's 1,440 SimplerEnv-Spatial trials by their final physical outcome (Figure 14). A *reasoning* error acts on a non-target: it grasps or pushes the wrong object, or moves the source to a non-target destination. An *execution* error selects the right target but does not complete the action: the target is not lifted, or it is not pushed or pulled far enough. Reasoning and execution errors account for 255 and 240 failures, respectively. Their balance differs by suite: 67% of the 228 SRef failures are execution errors, whereas 67% of the 267 SRsn failures are reasoning errors. Referring tasks are thus limited mainly by grasping and placement, and reasoning tasks by resolving the target: the two imagined-viewpoint and two compass-frame tasks alone account for 91 of the 180 SRsn reasoning errors. Figure 15 shows one example of each failure cause.

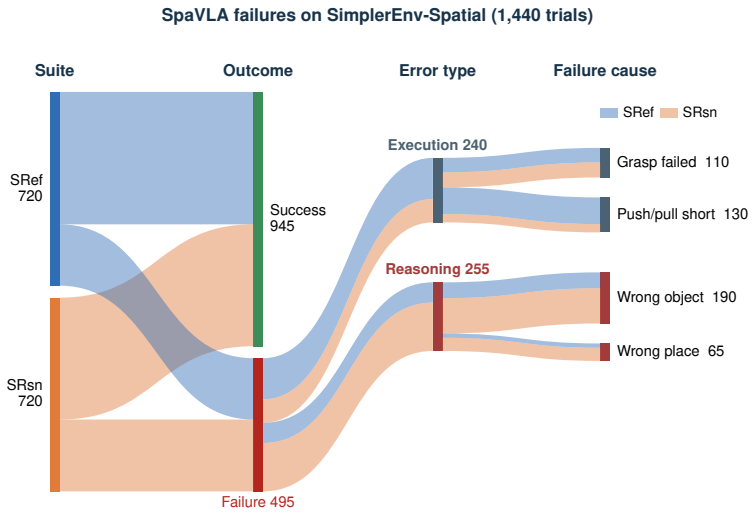


Figure 14: **SpaVLA failures on SimplerEnv-Spatial.** Episodes flow from suite to outcome, error type, and failure cause; band widths are proportional to episode counts.

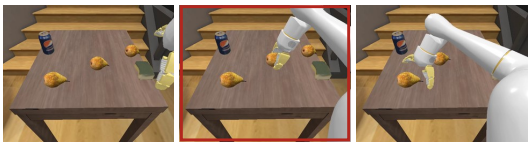
Rollout	Instruction	Response
	Imagine standing at the sponge and facing the Pepsi can. From that viewpoint, pick up the third pear from left to right among the three pears.	Pick up the rightmost pear. **Reasoning error: wrong object**
	Treat the direction from the sponge to the Pepsi can as north. Move the sponge near the lemon farthest west among the three lemons in that compass frame.	Move the sponge near the left lemon. **Reasoning error: wrong place**
	Pick the cube to the left of the apple.	Pick the yellow cube. **Execution error: grasp failed**
	open the third drawer from the top	Open the bottom drawer. **Execution error: push/pull short**

Figure 15: **Failure examples.** Two reasoning errors (wrong object, wrong place) and two execution errors (failed grasp, drawer not pulled far enough), each with the first, a middle (highlighted), and the final frame, the instruction, and the generated answer.

G LIMITATIONS

Embodiments and environments. Our simulated evaluations use the Google Robot and WidowX platforms, and our real-world study uses a single Franka setup. Extending SpaVLA to other embodiments, such as mobile or bimanual manipulators, and to longer-horizon tasks is left to future work. Under changed illumination, SpaVLA still outperforms both baselines, but its success drops to 56.3%, indicating that visual robustness can be further improved, for example with lighting augmentation during training.

Answer generation. In our simulated evaluations, SpaVLA generates a spatial answer once, from the first observation, and conditions the whole episode on it. Re-generating the answer as the scene changes could correct early mistakes and support tasks whose targets change during execution, at the cost of additional latency from answer generation.

Representations. The geometry and appearance encoders (VGGT and DINOv2) remain frozen, and the simulated policies use no proprioceptive state; adapting these encoders or adding proprioception may further improve action precision. Spatial training also slightly lowers MMBench accuracy relative to the Qwen3-VL backbone (81.2% versus 83.5%), which could be mitigated by a larger share of general multimodal data.