# POINTMAPVLA: SCALING ROBOT LEARNING WITH ROBOT-CENTRIC POINTMAPS

**Dongyoon Hwang**[1*] **Dongjin Kim**[1*] **Byungkun Lee**[1*] **Kyungmin Lee**[1] **Dohyun Lee**[1]
**Hojoon Lee**[2] **Minho Park**[1] **Jaegul Choo**[1†]
[1]KAIST AI [2]Holiday Robotics
{godnpeter, dj_kim, byungkun.lee}@kaist.ac.kr

## ABSTRACT

Vision-language-action models (VLAs) often struggle with precise manipulation because image observations do not directly provide the metric 3D geometry of the physical environment. We introduce **PointmapVLA**, the first VLA pretrained at scale with robot-centric pointmaps for joint 3D spatial understanding and robot control. Because pretrained VLM backbones lack a prior for interpreting pointmaps, we jointly pretrain on pointmap-augmented robot demonstrations and 3D spatial VQA, supervising 3D grounding, metric localization, spatial relation reasoning alongside action prediction. To support this pretraining, we construct a pointmap-paired corpus of 76.3M robot demonstration frames and 10.7M 3D spatial VQA examples. Across three simulation benchmarks and three real-world tasks, PointmapVLA delivers strong manipulation performance, establishing a new state of the art with a 92.4% success rate on LIBERO-Plus. In real-world manipulation under unseen target heights, lighting, and camera viewpoints, it achieves 1.8× the success rate of the strongest geometry-aware baseline. We will release the training code, model weights, and pointmap-paired training corpus.

## 1 INTRODUCTION

Vision-language-action models (VLAs) have shown promise in general-purpose robotic manipulation by leveraging the visual and semantic knowledge of pretrained vision-language models (VLMs) for robot control (Zitkovich et al., 2023; Kim et al., 2024; Black et al., 2025a; GEAR, 2025b). However, most VLAs rely solely on RGB observations, which do not directly provide the metric 3D structure of the robot's physical environment. Since robot manipulation takes place in 3D space, RGB-based policies must implicitly recover 3D spatial structure from 2D images before predicting actions. This indirect geometric inference can make policies sensitive to changes in camera viewpoint, object configuration, and visual appearance (Qu et al., 2025; Fan et al., 2026; Li et al., 2026b). These limitations motivate directly incorporating metric 3D geometry into VLA learning.

Recent 3D VLAs (Fan et al., 2026; Li et al., 2026a; Sun et al., 2025) explore various geometric inputs to better capture the physical 3D environment. Among geometric representations, robot-base frame pointmaps offer a natural representation for VLAs by preserving the image structure of RGB while expressing metric geometry directly in the robot frame. This aligns multi-view observations in a shared coordinate system in which robot actions are physically executed. Despite these advantages, pointmaps have only been explored in small-scale fine-tuning settings (Lee et al., 2026; Jia et al., 2025) and their use as an explicit input for large-scale VLA pretraining remains unexplored.

Scaling pointmaps to large-scale VLA pretraining, however, poses a fundamental challenge. VLAs inherit their backbones from VLMs pretrained extensively on RGB images, but these backbones have never been trained to interpret pointmaps (Team, 2026). Thus, simply adding pointmaps to robot demonstrations requires the model to learn an entirely new geometric modality through action prediction alone. Because robot demonstrations supervise which actions to take rather than how to interpret the observed geometry, they only indirectly guide pointmap understanding. Pointmap-conditioned 3D spatial VQA (Liang et al., 2026) instead provides direct supervision by explicitly

[*]Equal contribution.
[†]Corresponding author.

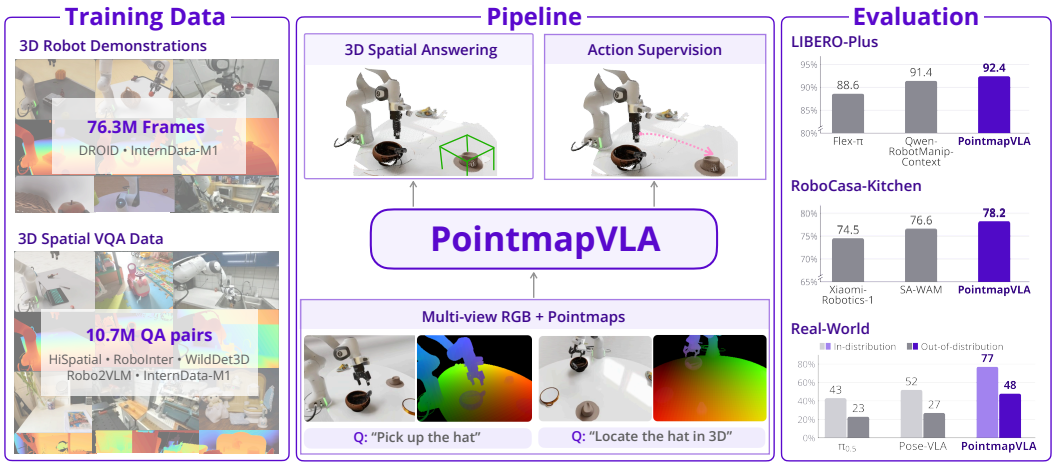



Figure 1: **Overview.** PointmapVLA jointly learns robot control and 3D spatial understanding from multi-view RGB and pointmaps, using 76.3M pointmap-enriched robot frames and 10.7M pointmap-paired spatial VQA examples. It achieves state-of-the-art performance on LIBERO-Plus, 78.2% success on RoboCasa, and outperforms baselines in real-world manipulation.

training the model for 3D grounding, metric localization, and spatial relation reasoning. Without such supervision, the model may struggle to fully exploit the metric 3D information in pointmaps. Effectively scaling pointmaps for VLA pretraining therefore requires not only large-scale robot data, but also pointmap-based spatial supervision that teaches the model to interpret this new modality.

In response, we introduce **PointmapVLA**, the first VLA to pretrain at scale on robot-base frame pointmaps with both robot action and explicit 3D spatial supervision. We train PointmapVLA on 76.3M pointmap-enriched robot frames and 10.7M pointmap-paired 3D spatial VQA examples. Spatial VQA teaches the model to interpret the 3D metric pointmap geometry, and robot demonstrations teach it to act on that geometry. Architecturally, PointmapVLA extends a pretrained VLM with a pointmap encoder initialized from its RGB encoder and fuses spatially aligned RGB and pointmap features, so the backbone jointly encodes visual semantics and metric 3D geometry. We first jointly train the VLM backbone on 3D spatial VQA and discretized actions under a shared next-token objective, placing spatial reasoning and action semantics in a shared latent space. We then add a DiT-based action expert for fine-grained continuous actions while retaining the 3D spatial VQA and discrete-action objectives, so that spatial supervision persists throughout training. This two-stage training preserves explicit supervision for interpreting pointmap geometry while extending the model to fine-grained continuous control.

To validate PointmapVLA, we conduct extensive experiments across three simulation benchmarks, three real-world manipulation tasks, and a metric 3D spatial reasoning benchmark. PointmapVLA achieves 99.0% on LIBERO (Liu et al., 2023), sets a new state of the art on LIBERO-Plus (Fei et al., 2025) with 92.4%, and reaches 78.2% on RoboCasa (Nasiriany et al., 2024). In real-world Stack blocks, Peg in hole, and Pick-and-place, PointmapVLA achieves 77% and 48% average success under seen and unseen conditions, compared with 52% and 27% for the strongest baseline. On SpatialRGPT-Bench, it scores 70.4%, demonstrating strong metric 3D spatial understanding. Ablations show that robot-base frame pointmaps outperform alternative geometric representations and that 3D spatial VQA pretraining improves manipulation performance. Together, these results show that large-scale pointmap pretraining on both robot demonstrations and 3D spatial VQA enables PointmapVLA to acquire strong 3D spatial understanding and translate it into precise manipulation.

Our contributions are threefold.

- We introduce PointmapVLA, the first VLA to scale robot-base-frame pointmaps to large-scale pretraining for joint 3D spatial understanding and robot control.
- We pretrain on 76.3M pointmap-enriched robot frames and 10.7M pointmap-paired 3D spatial VQA examples, explicitly teaching pointmap geometry beyond action supervision.
- PointmapVLA achieves state-of-the-art performance on LIBERO-Plus and strong RoboCasa and real-world results, while ablations validate the benefits of robot-base-frame pointmaps and 3D spatial VQA pretraining.

## 2 Related Work

**3D Spatial Understanding in Vision-Language Models.** Recent work has explored enhancing spatial understanding in VLMs through explicit spatial supervision, including metric estimation, object localization, and spatial relation reasoning (Chen et al., 2024; Song et al., 2025). Other approaches incorporate geometric information through 3D features (Hong et al., 2023), positional embeddings (Cheng et al., 2026), or depth (Cheng et al., 2024; Chen et al., 2025b; Zhou et al., 2025a). Most closely related, HiSpatial (Liang et al., 2026) combines pixel-aligned metric pointmaps with hierarchical 3D spatial VQA, demonstrating the value of pointmaps for explicit spatial understanding. However, these works focus on acquiring 3D spatial understanding rather than transferring the learned knowledge to robot control. We extend this supervision to robot control by jointly training PointmapVLA for spatial VQA and action prediction, with pointmaps and coordinate-valued VQA targets expressed in the robot-base frame for robot scenes.

**3D Geometry and Spatial Learning for Robot Control.** Robot policies incorporate geometry through depth-derived positional embeddings (Qu et al., 2025), point-cloud features (Li et al., 2026a; Sun et al., 2025; Fan et al., 2026; Yang et al., 2025), and structured pointmaps (Lee et al., 2026; Jia et al., 2025). Another line of work uses 3D VQA pretraining (Nikolov et al., 2026), camera-centric pose pretraining (Lin et al., 2026a), or joint spatial-QA and action learning (Cai et al., 2025; Shi et al., 2026). Other methods introduce geometric priors through feature alignment (Li et al., 2026b), predictive kinematic and geometric supervision (Qian et al., 2026), or geometry and affordance supervision (Zheng et al., 2026b). VLM-generated 3D traces and trajectories guide manipulation (Zhou et al., 2025b; Tan et al., 2026; Hwang et al., 2026), while SG-WAM predicts future semantic and geometric features to guide world-action models (He et al., 2026). Geometric- and world-action models jointly predict actions and future geometry (Han et al., 2026; Yan et al., 2026; Gonzalez et al., 2026; Guo et al., 2026), while action-conditioned 3D dynamics support model-based control (Huang et al., 2026b). Most closely related are PointMapPolicy (Jia et al., 2025) and SeeR-VLA (Lee et al., 2026). PointMapPolicy trains pointmap-based policies on manipulation benchmarks, while SeeR-VLA incorporates pointmaps when fine-tuning pretrained VLAs. However, neither explores pointmaps as a core modality for large-scale VLA pretraining. In contrast, we pretrain PointmapVLA at scale on both pointmap-paired robot demonstrations and pointmap-conditioned 3D spatial VQA. Robot demonstrations supervise action prediction from robot-base-frame pointmaps, while VQA explicitly supervises 3D grounding, metric localization, and spatial relation reasoning. Together, these objectives teach the model to understand pointmap geometry and act on it.

## 3 PointmapVLA

We present PointmapVLA, which augments a pretrained VLM with paired RGB and pointmap observations to explicitly represent metric 3D geometry. The model is jointly trained for 3D spatial reasoning and robot control. We describe the architecture in Section 3.1, the training recipe in Section 3.2, and the training data in Section 3.3. Further details are provided in Appendix A.

### 3.1 Model Architecture

As illustrated in Figure 2, PointmapVLA receives multi-view visual observations $\mathcal{O} = \{(I^i, P^i)\}_{i=1}^{N_{\text{view}}}$ and a language input $\ell$, where each pair consists of an RGB image $I^i$ and its pixel-aligned pointmap $P^i$. The backbone predicts text answers for spatial VQA and discrete action tokens, while the action expert generates an $H$-step continuous action chunk conditioned on the backbone features and robot state $\mathbf{s}_t$.

**Pointmap Construction.** To provide explicit metric 3D geometry, we construct a pointmap for each camera view by back-projecting depth into 3D coordinates and transforming the resulting points into a reference frame $r$. For notational simplicity, we omit the view superscript $i$ below. Let $(u, v)$ denote pixel coordinates, $K \in \mathbb{R}^{3\times3}$ the camera intrinsics, and $D(u, v)$ the optical-axis depth in meters. The 3D point corresponding to each valid pixel is given by

$$P(u, v) = R_{r\leftarrow c}\big(D(u, v)K^{-1}[u, v, 1]^\top\big) + \mathbf{t}_{r\leftarrow c}, \tag{1}$$

where $R_{r\leftarrow c}$ and $\mathbf{t}_{r\leftarrow c}$ denote the rotation and translation from the camera frame $c$ to the reference frame $r$. For robot observations, $r$ is the robot-base frame. For general-scene VQA data without a physical robot, $r$ is a randomized pseudo-base frame, as detailed in Appendix B.3. The resulting pointmap $P^i \in \mathbb{R}^{h\times w\times3}$ is pixel-aligned with the corresponding RGB image $I^i$.

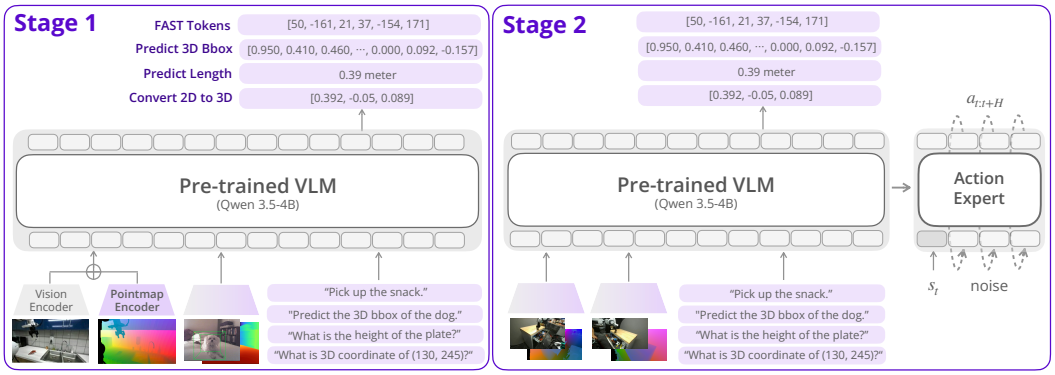



Figure 2: **PointmapVLA architecture and training recipe.** RGB images and aligned pointmaps are processed by separate visual encoders, whose features are fused patch-wise to combine visual semantics with metric 3D geometry. Stage 1 jointly trains spatial VQA and discretized action prediction, while Stage 2 adds a continuous action expert and retains both token-level objectives.

**Pointmap-Augmented Vision-Language Backbone.** To process pointmaps alongside RGB images, we introduce a dedicated pointmap encoder $E_{\text{pm}}$ with the same architecture as the VLM's RGB encoder $E_{\text{rgb}}$. Since the aligned RGB image and pointmap of each view $i$ share the same patch layout, we fuse their spatially corresponding features through simple elementwise addition:

$$Z^i = E_{\text{rgb}}(I^i) + E_{\text{pm}}(P^i). \tag{2}$$

To narrow the representation gap between $E_{\text{pm}}$ and the pretrained language backbone, we initialize $E_{\text{pm}}$ from the pretrained $E_{\text{rgb}}$, a choice validated in Table 5. The fused tokens from all views are concatenated with the language tokens of $\ell$ and processed by the language model. The resulting hidden states support spatial VQA and condition the action expert for continuous robot control.

**DiT-Based Action Expert.** For fine-grained physical control, we employ a Diffusion Transformer (DiT) as our continuous action expert (Black et al., 2025a; Ma et al., 2026; Ye et al., 2026a). The DiT matches the language model's depth, with learned linear projections mapping each VLM layer's hidden states into the corresponding DiT layer's embedding space. An MLP embeds the current robot state $\mathbf{s}_t$ as an additional input token alongside the action tokens. The expert interleaves self-attention over these input tokens with cross-attention to the projected VLM hidden states, iteratively transforming noise into the action chunk $\mathbf{a}_{t:t+H}$ through flow matching.

### 3.2 Training Recipe

As illustrated in Figure 2, we train PointmapVLA with a two-stage recipe that first develops 3D spatial understanding and then carries it into continuous robot action prediction. We jointly learn spatial grounding and discrete actions in the first stage, then introduce continuous control while retaining spatial and discrete-action supervision. This preserves spatial understanding during the transition to fine-grained manipulation. We next describe the two stages in detail.

**Stage 1. Spatial and Discrete Action Pretraining.** We jointly train the pointmap-augmented backbone on spatial VQA and robot demonstrations with a shared next-token prediction objective (Black et al., 2025a; Ma et al., 2026), so that spatial understanding and discrete action prediction are learned within the same latent space. The model predicts text answers for VQA samples and FAST-encoded action chunks (Pertsch et al., 2025) for robot samples, with the robot state additionally provided as input. For each task $k \in \{\text{VQA}, \text{FAST}\}$, let $x_n$ denote the input of sample $n$, $y_{n,j}$ its $j$-th target token, and $\mathcal{S}_k$ the corresponding set of target-token indices. We optimize

$$\mathcal{L}_k = -\frac{1}{|\mathcal{S}_k|} \sum_{(n,j)\in\mathcal{S}_k} \log p_\theta(y_{n,j} \mid x_n, y_{n,<j}), \qquad \mathcal{L}_{\text{S1}} = \mathcal{L}_{\text{VQA}} + \mathcal{L}_{\text{FAST}}. \tag{3}$$

Here, $\theta$ denotes the backbone parameters, and each loss is computed only over its target tokens. For Stage 1, we sample spatial VQA more heavily than robot demonstrations, encouraging the backbone to ground action semantics in metric 3D structure before continuous control is introduced in Stage 2.

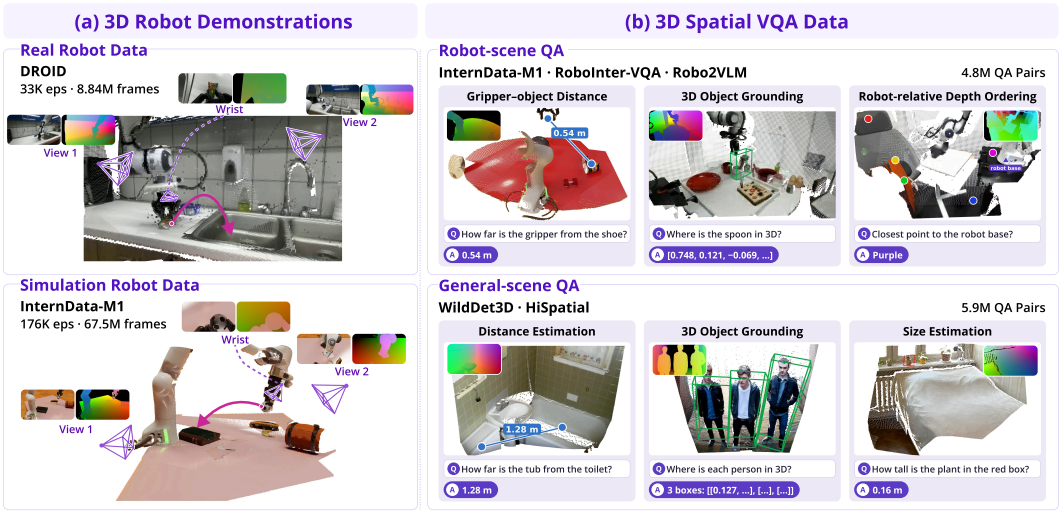



Figure 3: **Training data overview.** (a) 76.3M pointmap-paired robot demonstration frames for action learning. (b) 10.7M pointmap-paired VQA examples for 3D grounding and spatial reasoning.

**Stage 2. Continuous Action Co-training.** We introduce a randomly initialized action expert and train it jointly with the backbone for continuous control. Given an action chunk $A_t = \mathbf{a}_{t:t+H-1}$, we construct a noisy input $A_t^\tau = (1-\tau)\epsilon + \tau A_t$, where $\epsilon \sim \mathcal{N}(0, I)$ and $\tau \in [0, 1]$. Conditioned on the backbone hidden states and robot state $\mathbf{s}_t$, the expert predicts a velocity $\hat{v}_t$ from $A_t^\tau$ and $\tau$. To preserve spatial understanding acquired in Stage 1, we combine the flow-matching loss with $\mathcal{L}_{\text{S1}}$,

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{A_t,\tau,\epsilon}\left[\|\hat{v}_t - (A_t - \epsilon)\|_2^2\right], \qquad \mathcal{L}_{\text{S2}} = \mathcal{L}_{\text{FM}} + 0.1\,\mathcal{L}_{\text{S1}}. \tag{4}$$

We update all components end-to-end and shift the sampling ratio toward robot demonstrations, allowing continuous-control learning to refine the backbone while preserving the spatial supervision established in Stage 1. Overall, this two-stage training recipe carries 3D spatial understanding into continuous robot control while preserving explicit spatial supervision throughout training.

### 3.3 Training Data

We combine pointmap-augmented robot demonstrations for action supervision with 3D spatial VQA for spatial supervision, as shown in Figure 3. Detailed dataset composition, construction procedures, and reference-frame alignment are provided in Appendix B.

**Robot Demonstrations.** To provide action supervision with explicit 3D geometry, we construct a pointmap-augmented robot demonstration dataset of 210K episodes (76.3M frames) spanning real-world and simulated manipulation (Figure 3a). For DROID (Khazatsky et al., 2024), we use metric depth and refined exterior-camera extrinsics from PointWorld (Huang et al., 2026b), and recover wrist-camera poses from robot states and hand–eye calibration (Appendix B.1). For InternData-M1 (Chen et al., 2025c), since the original InternData-M1 dataset provides only RGB observation, we re-render the simulated scenes using the simulation metadata to extract precise metric depth maps. We express all pointmaps in the robot-base frame, providing a unified robot-centric 3D representation of the physical environment across camera views. We then pair these multi-view RGB–pointmap observations with language instructions and robot actions.

**3D Spatial VQA.** To explicitly supervise 3D spatial reasoning, we construct 10.7M pointmap-paired QA examples spanning robot and general scenes. For robot scenes, we build on InternData-M1 (Chen et al., 2025c), RoboInter-VQA (Li et al., 2026c), and Robo2VLM (Chen et al., 2025a) to construct manipulation-relevant 3D supervision, including object grounding, gripper-object distances, pick-and-place targets, and 3D extensions of existing 2D annotations. We also replace camera-relative queries with robot-base- and gripper-relative variants to encourage robot-centric spatial reasoning. For general scenes, we use WildDet3D (Huang et al., 2026a) and HiSpatial (Liang et al., 2026) to provide broader supervision for spatial relations, metric estimation, 3D localization, and 3D object detection. We additionally include 4.09M RGB-only VQA examples from

ST4VLA (Ye et al., 2026a) to preserve 2D grounding and general vision-language capabilities. For general-scene QA with 3D coordinate or vector answers, we express both the pointmaps and their targets in randomized pseudo-base frames, encouraging the model to interpret geometry under varying spatial references. Scalar, choice, and text examples from HiSpatial instead use upright camera-origin geometry and retain their corresponding answer conventions.

# 4 EXPERIMENTS

## 4.1 EXPERIMENTAL SETUP

**Benchmarks.** We evaluate PointmapVLA on LIBERO (Liu et al., 2023) for in-distribution performance and on LIBERO-Plus (Fei et al., 2025) for zero-shot robustness to visual, spatial, and linguistic perturbations without further fine-tuning. We also evaluate on RoboCasa (Nasiriany et al., 2024), which is composed of 24 atomic kitchen tasks requiring fine-grained spatial reasoning. In the real world, we design three tabletop tasks which require geometric reasoning: (i) Peg in hole, (ii) Stack blocks, and (iii) Pick-and-place with spatial instructions. We evaluate both in-distribution and out-of-distribution (OOD) conditions, including unseen visual disturbances, target heights, and novel viewpoints. Further details are provided in Appendix D.

**Baselines.** We group our baselines into four categories: WAMs, VLAs, geometry-aware WAMs, and geometry-aware VLAs. We define geometry-aware models as those that leverage 3D priors obtained through direct geometric inputs (Lin et al., 2026a; Yan et al., 2026), representations distilled from 3D foundation models (Li et al., 2026b), or auxiliary geometric objectives (Guo et al., 2026). Unless otherwise noted, we adopt baseline results reported in prior work. For real world experiments, we compare PointmapVLA against three baselines: (1) $\pi_{0.5}$ (Black et al., 2025a), an RGB-only generalist VLA, (2) FP3 (Yang et al., 2025), a 3D foundation policy that operates on point clouds, and (3) Pose-VLA (Lin et al., 2026a), a geometry-aware VLA that uses depth and ray inputs. Across all tables, **bold** and underlining indicate the best and second-best results, respectively.

**Implementation Details.** We use Qwen3.5-4B (Team, 2026) as the VLM backbone and pretrain for 100K updates (50K per stage) with a global batch size of 368. Downstream fine-tuning uses 300 demonstrations per task for RoboCasa and 60 per task for real-world tasks. Simulation pointmaps use ground-truth depth and camera extrinsics, while real-world pointmaps use stereo-estimated depth and calibrated extrinsics. We report mean success rates and standard deviations over three evaluation runs. Training and observation-processing details are provided in Appendix A, and simulation evaluation protocols in Appendix C.

## 4.2 MAIN RESULTS

**Simulation Benchmarks.** On LIBERO, where performance is largely saturated among recent methods, PointmapVLA achieves a 99.0% success rate, within 0.4 points of the best-performing baseline, ABot-M0.5 (Table 1). On the more challenging LIBERO-Plus, PointmapVLA sets a new state of the art with 92.4% overall success, surpassing the strongest VLA, Qwen-RobotManip-Context, by 1.0 point and the strongest geometry-aware WAM, Flex-$\pi$, by 3.8 points. On RoboCasa, PointmapVLA achieves 78.2%, outperforming the strongest baselines SA-WAM (76.6%) and Xiaomi-Robotics-1 (74.5%) (Table 2). The margin is larger over SeeR-VLA (61.5%), which introduces robot-centric pointmaps only during downstream fine-tuning, suggesting that learning pointmap representations during large-scale pretraining matters more than adding geometry at adaptation time. Full benchmark tables, including additional baselines and detailed per-suite, perturbation-wise, and per-task results, are provided in Appendix C. Overall, these results show that PointmapVLA maintains strong in-distribution performance while remaining robust to visual perturbations and performing well on spatially demanding manipulation tasks.

**Real-World Results.** As shown in Figure 4, PointmapVLA achieves the highest average success rates of 77% under seen and 48% under unseen conditions, while Pose-VLA, the strongest baseline, reaches 52% and 27%, respectively. Under seen conditions, PointmapVLA outperforms every baseline on every task, reaching 75% on Peg in hole, 75% on Stack blocks, and 80% on spatially instructed Pick-and-place. Its consistent gains over the RGB-only $\pi_{0.5}$, the point-cloud policy FP3, and the geometry-aware Pose-VLA suggest that robot-base pointmaps provide useful metric geometry for alignment, placement, and spatial grounding. Under unseen conditions, PointmapVLA retains

Table 1: **LIBERO and LIBERO-Plus.** Success rates (%). LIBERO-Plus is evaluated zero-shot.

| Method | LIBERO Avg. | LIBERO-Plus Total |
|---|---|---|
| *WAMs* | | |
| Cosmos-Policy (2026b) | 98.5 | 82.2 |
| ABot-M0.5 (2026) | **99.4** | 83.4 |
| *VLAs* | | |
| InternVLA-A1.5 (2026) | 98.9 | 84.8 |
| Qwen-RobotManip (2026a) | 99.1 | 89.0 |
| Qwen-RobotManip-Context (2026a) | 99.2 | 91.4 |
| *Geometry-aware WAMs* | | |
| GAM (2026) | 97.6 | 85.5 |
| GIFT-WAM-IDM (2026b) | 98.5 | 87.8 |
| Flex-$\pi$ (2026) | 98.5 | 88.6 |
| *Geometry-aware VLAs* | | |
| Spatial Forcing (2026b) | 98.5 | 29.1 |
| ABot-M0 (2026) | 98.6 | 80.5 |
| **PointmapVLA (Ours)** | 99.0±0.1 | **92.4**±0.3 |

Table 2: **RoboCasa.** Average success rate (%).

| Method | Avg. Success (%) |
|---|---|
| *WAMs* | |
| UWM (2025) | 60.8 |
| DreamZero (2026b) | 62.4 |
| Cosmos Policy (2026b) | 67.1 |
| *VLAs* | |
| RLDX-1 (2026a) | 70.6 |
| Xiaomi-Robotics-1 (2026) | 74.5 |
| *Geometry-aware WAMs* | |
| GAM (2026) | 69.4 |
| DeVA (2026a) | 72.0 |
| SA-WAM (2026) | 76.6 |
| *Geometry-aware VLAs* | |
| GeoPredict (2026) | 52.4 |
| SeeR-VLA (2026) | 61.5 |
| **PointmapVLA (Ours)** | **78.2**±0.9 |

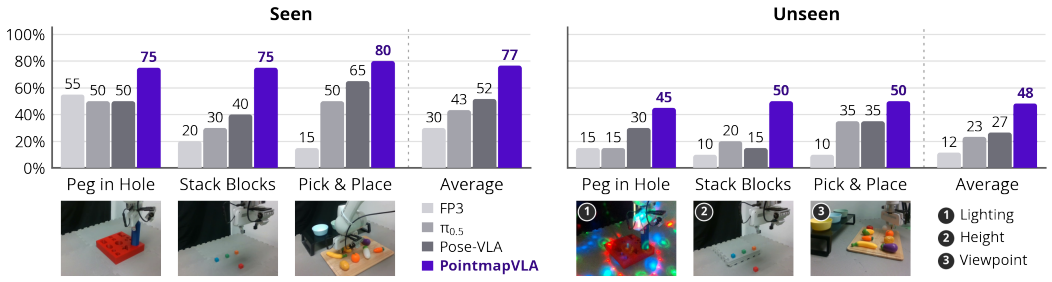



Figure 4: **Real-robot experiments.** Success rates under seen and unseen conditions are shown on the left and right, respectively. Examples of each task are shown below.

45% success on Peg in hole with severe visual disturbances, while Pose-VLA reaches 30%, indicating visual robustness. With unseen target heights and a novel camera viewpoint, PointmapVLA reaches 50% on both Stack blocks and Pick-and-place, while the strongest baselines reach 20% and 35%, respectively, highlighting the benefit of metric 3D geometry for adapting placement and maintaining spatial grounding across viewpoints. Overall, these results show that robot-centric 3D geometry improves both precise manipulation and robustness to environmental changes. Videos of the real-robot rollouts are also provided in the supplementary material.

### 4.3 Analysis

**Robustness to Camera Occlusion.** Robot-base pointmaps unify multi-view observations in a shared 3D coordinate frame, allowing visible geometry from remaining views to support manipulation under occlusion. To evaluate robustness to camera-view occlusion, we physically occlude either the left camera alone or both the left and right cameras at test time, while keeping the wrist view possible. As shown in Table 3, occluding even a single camera view sharply reduces Pose-VLA's success rate from 65% to 25%, whereas PointmapVLA retains 65% success compared with 80% without occlusion. When two views are occluded, PointmapVLA still achieves 50% success, while Pose-VLA's performance degrades to 20%. The qualitative 3D grounding results show a similar failure, where Pose-VLA fails to output 3D bounding boxes even under single-view occlusion in the illus-

Table 3: **View occlusion.** Pick-and-place success rates (%) under increasing occluded camera views.

| Method | None | 1 view occ. | 2 views occ. |
|---|---|---|---|
| Pose-VLA | 65 | 25 | 20 |
| **PointmapVLA** | **80** | **65** | **50** |

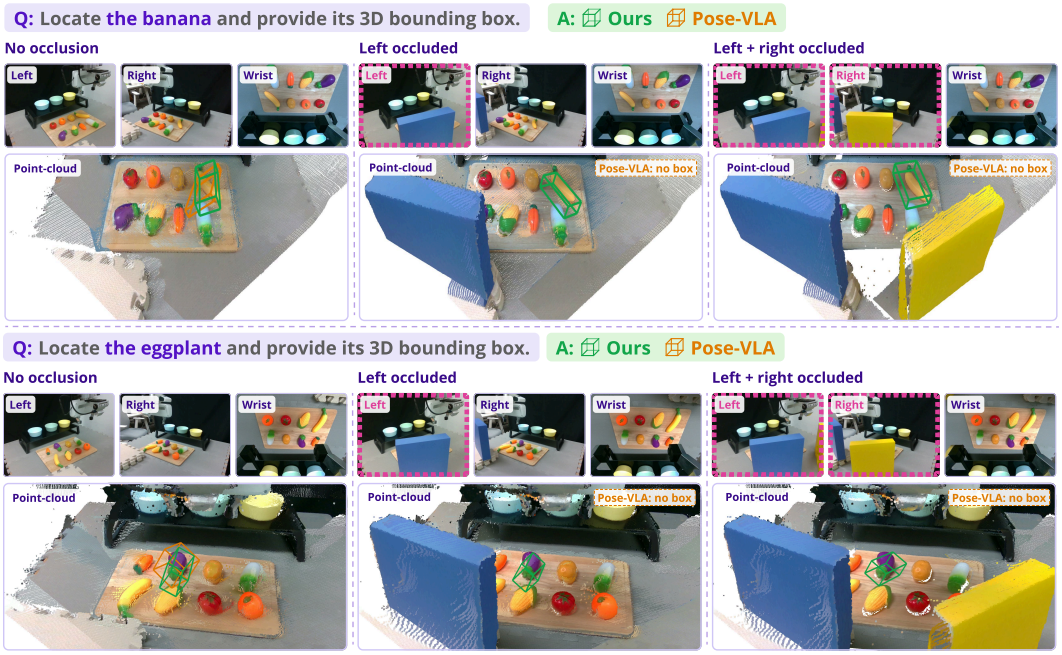



Figure 5: **3D grounding under camera-view occlusion.** Qualitative predictions as camera views are progressively occluded. From left to right: no occlusion, left view occluded, and both left and right views occluded. PointmapVLA (green) continues to predict accurate 3D bounding boxes under occlusion, whereas Pose-VLA (orange) fails to output 3D bounding boxes. See Appendix C.5.

trated examples, while PointmapVLA continues to successfully localize targets from the remaining views (Figure 5). Unlike our robot-base pointmaps, Pose-VLA relies on depth observations without explicitly unifying multi-view geometry in a shared robot-centric frame, limiting its ability to integrate information across views. These results highlight the importance of robot-base pointmaps for multi-view spatial understanding and manipulation under camera occlusion. Additional qualitative 3D grounding examples are provided in Appendix C.5, and the corresponding rollout videos are included in the supplementary material.

**3D Spatial Understanding.** We evaluate PointmapVLA on the quantitative split of SpatialRGPT-Bench (Cheng et al., 2024) to assess metric 3D spatial reasoning (Table 4). Pretrained on large-scale 3D spatial VQA data, PointmapVLA achieves 70.43% average accuracy, outperforming Qwen3.5-4B (42.44%) and RGB-D models specialized for spatial reasoning, despite joint training with robot demonstrations. These results highlight the effectiveness of pointmaps in enabling strong 3D spatial understanding in PointmapVLA. We further show in Table 7 that such spatial knowledge consistently improves robot manipulation performance. Full benchmark results are provided in Appendix C.

### 4.4 Ablation Studies

We ablate three components of PointmapVLA, including the geometric input modality, pointmap encoder design, and 3D spatial VQA pretraining. Due to computational constraints, the geometric input and encoder studies are performed without pretraining, with all variants fine-tuned directly from Qwen3.5-4B under the same setup.

**Geometric Input Modality.** We compare several geometric input representations and reference frames on RoboCasa (Table 6). Adding depth to the RGB-only baseline improves success from 67.9% to 70.0%. Adding camera ray directions, following Pose-VLA, further improves performance to 70.9%, while camera-frame pointmaps, as adopted by Flex-$\pi$, achieve 71.2%. However, camera-frame pointmaps represent each view in its own coordinate frame, so the same 3D point has different coordinates across cameras. Transforming the pointmaps into the robot-base frame places all views in a shared coordinate system and achieves the highest success rate of 73.6%. These results highlight the benefit of representing multi-view geometry in a shared robot-centric frame.

Table 4: **SpatialRGPT-Bench.** Quantitative accuracy (%).

| Model | Input | Avg. |
|---|---|---|
| InternVL3.5-8B | RGB | 30.53 |
| Qwen3.5-4B | RGB | 42.44 |
| RoboRefer-8B-SFT | RGB-D | 39.25 |
| SpatialRGPT-8B | RGB-D | 56.22 |
| MM-Spatial-3B | RGB-D | 68.70 |
| PointmapVLA | RGB-Pointmap | **70.43** |

Table 5: **Pointmap encoder.** RoboCasa success rates (%).

| Encoder | Pretrained | RGB | Pointmap |
|---|---|---|---|
| SigLIP2 (2025) | ✓ | 65.3±1.1 | 70.2±0.6 |
| DINOv3 (2025) | ✓ | 66.8±0.6 | 70.8±0.7 |
| MoGe-3 (2026) | ✓ | 68.3±0.6 | 69.8±0.5 |
| Linear embed. | × | – | 66.6±0.5 |
| Qwen3.5 (2026) | × | – | 63.0±0.4 |
| Qwen3.5 (Ours) | ✓ | – | **73.6**±0.3 |

Table 6: **Geometric input.** RoboCasa success rates (%).

| Input | SR (%) |
|---|---|
| None (RGB-only) | 67.9±0.1 |
| Depth | 70.0±0.6 |
| Depth + ray | 70.9±0.3 |
| Pointmap (camera) | 71.2±0.9 |
| Pointmap (robot base) | **73.6**±0.3 |

Table 7: **3D spatial VQA pretraining.** Success rates (%) across pretraining strategies.

| Pretrain | Simulation | | Real |
|---|---|---|---|
| | RoboCasa | LIBERO-Plus | Seen/Unseen |
| None | 73.6±0.3 | 82.6±0.1 | – |
| Robot | 76.4±0.1 | 88.2±0.9 | 70 / 38 |
| Robot + VQA | **78.2**±0.9 | **92.4**±0.3 | **77 / 48** |

**Pointmap Encoder.** We examine both the design of the pointmap encoder and whether its gains arise from explicit geometry rather than additional model capacity (Table 5). A shallow linear embedding and a randomly initialized Qwen3.5 visual encoder achieve 66.6% and 63.0%, while pretrained SigLIP2 (Tschannen et al., 2025), DINOv3 (Siméoni et al., 2025), and MoGe-3 (Kong et al., 2026) encoders improve performance to 70.2%, 70.8%, and 69.8%, respectively. Initializing the pointmap encoder from Qwen3.5's pretrained RGB visual encoder performs best at 73.6%, suggesting that starting from a representation already aligned with the language backbone facilitates adaptation to pointmap inputs. Importantly, feeding RGB rather than pointmaps to the same auxiliary encoders yields lower performance, with SigLIP2, DINOv3, and MoGe-3 reaching 65.3%, 66.8%, and 68.3%. These matched-capacity comparisons indicate that the improvement comes primarily from the geometric information in pointmaps rather than from adding an auxiliary encoder.

**Effect of 3D Spatial VQA Pretraining.** We study whether explicit 3D spatial supervision provides additional gains beyond action supervision from robot demonstrations. We compare no pretraining, pretraining on pointmap-enriched robot demonstrations alone, and joint pretraining with additional pointmap-paired 3D spatial and embodied VQA, followed by the same downstream finetuning (Table 7). Robot-only pretraining improves RoboCasa and LIBERO-Plus by 2.8 and 5.6 points. Adding 3D spatial VQA further raises performance by 1.8 and 4.2 points, while increasing real-world success by 7 points under seen conditions and 10 points under unseen conditions. These gains indicate that direct 3D spatial supervision complements action supervision during pretraining.

## 5 Conclusion

In this work, we present PointmapVLA, a VLA that unifies 3D spatial understanding and precise manipulation through large-scale pretraining on robot-base-frame pointmaps. Joint pretraining on 76.3M pointmap-enriched robot frames and 10.7M pointmap-conditioned spatial VQA examples teaches the model to interpret metric geometry and use it for action prediction. Our two-stage recipe retains spatial and discrete-action supervision while introducing a continuous action expert. Experiments demonstrate state-of-the-art performance on LIBERO-Plus and strong results on RoboCasa and real-world manipulation, while ablations show that spatial VQA improves control beyond robot-only pretraining. However, our approach relies on depth and camera calibration, requiring additional deployment requirements. Advances in metric depth estimation and automated camera-to-robot calibration could reduce this burden and facilitate broader deployment of pointmap-based VLAs. We hope this work provides a foundation for scaling VLAs that learn to understand and act in the physical world through shared geometric representations.

### AI USE STATEMENT

In this work, we employed generative AI tools to translate author-written drafts into English, to edit and rewrite text for readability and concision, to give editorial feedback on the clarity of our exposition, and to implement and edit portions of the data processing and training code. We did not use generative AI tools to propose hypotheses, design experiments, generate synthetic datasets, or interpret results, and tasks related to mathematical claims and proofs are not applicable to this work. We have reviewed at all AI-assisted work. All AI-edited text was verified for fidelity to the original drafts and technical accuracy by the authors. AI-assisted code was verified and tested for correctness by three authors. All reported numbers were verified against the original experiment logs by the authors. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

### ETHICS STATEMENT

This work uses publicly released datasets (DROID, PointWorld-DROID, InternData-M1, RoboInter-VQA, Robo2VLM, WildDet3D, and HiSpatial, including Objects365 and CA-1M images, together with the RGB-only co-training sources of ST4VLA), benchmarks (LIBERO, LIBERO-Plus, RoboCasa, and SpatialRGPT-Bench), and pretrained models in accordance with their licenses. Several sources restrict use to non-commercial research (e.g., InternData-M1, RoboInter-VQA, and CA-1M), and we will release derived artifacts only under terms compatible with each source. Real-world demonstrations were collected by the authors in a laboratory setting and contain no personal or identifying information. We are not aware of other ethical concerns specific to this work.

### REPRODUCIBILITY STATEMENT

We describe the model architecture and training recipe in Sections 3.1 and 3.2, and provide state and action representations, architecture specifications, prompt formats, optimization hyperparameters, and the data mixture in Appendix A. The construction of the pointmap-augmented robot demonstrations and the spatial and embodied QA data, including reference-frame alignment, is detailed in Appendix B. Benchmark protocols and full per-suite, per-perturbation, and per-task results are reported in Section 4.1 and Appendix C, and the real-world hardware setup, fine-tuning settings, evaluation conditions, and success criteria are described in Appendix D. We plan to release the training code and model weights for non-commercial research use upon publication. For the training data, we will release our processing scripts and, where source licenses permit, the derived annotations and pointmaps under each source's license; third-party images and data derived from CA-1M, whose license prohibits derivatives, are not redistributed and must be obtained from the original providers.

## REFERENCES


Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert 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. In *Conference on Robot Learning*, 2025a. 1, 4, 6, 28, 29, 30, 33

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. $\pi_0$: A vision-language-action flow model for general robot control. *Robotics: Science and Systems*, 2025b. 28, 29, 30

Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. Univla: Learning to act anywhere with task-centric latent actions. *Robotics: Science and Systems*, 2025. 28, 29

Wenxiao Cai, Iaroslav Ponomarenko, Jianhao Yuan, Xiaoqi Li, Wankou Yang, Hao Dong, and Bo Zhao. Spatialbot: Precise spatial understanding with vision language models. In *IEEE International Conference on Robotics and Automation*, 2025. 3

Boyuan Chen, Zhuo Xu, Sean Kirmani, Brian Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. SpatialVLM: Endowing vision-language models with spatial reasoning capabilities. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 14455–14465, 2024. 3

Kaiyuan Chen, Shuangyu Xie, Zehan Ma, Pannag R Sanketi, and Ken Goldberg. Robo2VLM: Visual question answering from large-scale in-the-wild robot manipulation datasets. *arXiv preprint arXiv:2505.15517*, 2025a. 5, 20

Pingyi Chen, Yujing Lou, Shen Cao, Jinhui Guo, Lubin Fan, Yue Wu, Lin Yang, Lizhuang Ma, and Jieping Ye. Sd-vlm: Spatial measuring and understanding with depth-encoded vision-language models. *Advances in Neural Information Processing Systems*, 38:33938–33968, 2025b. 3

Ronghan Chen, Yandan Yang, Zuojin Tang, Dongjie Huo, Tong Lin, Haoning Wu, Haoyun Liu, Yuzhi Chen, Lulu Zheng, Botai Yuan, Tianlun Li, Mingxin Wang, Dekang Qi, Bin Hu, Wei Mei, Yuze Xuan, Haolong Yang, Yanqing Zhu, Mu Xu, Zhiheng Ma, and Xinyuan Chang. Abot-m0.5: Unified mobility-and-manipulation world action model. *arXiv preprint arXiv:2607.00678*, 2026. 7, 28, 29

Xinyi Chen, Yilun Chen, Yanwei Fu, Ning Gao, Jiaya Jia, Weiyang Jin, Hao Li, Yao Mu, Jiangmiao Pang, Yu Qiao, et al. InternVLA-M1: A spatially guided vision-language-action framework for generalist robot policy. *arXiv preprint arXiv:2510.13778*, 2025c. 5, 19

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. *Advances in Neural Information Processing Systems*, 2024. 3, 8

An-Chieh Cheng, Yang Fu, Yukang Chen, Zhijian Liu, Xiaolong Li, Subhashree Radhakrishnan, Song Han, Yao Lu, Jan Kautz, Pavlo Molchanov, et al. 3d aware region prompted vision language model. In *International Conference on Learning Representations*, volume 2026, pp. 123286–123308, 2026. 3

Xianzhe Fan, Shengliang Deng, Xiaoyang Wu, Yuxiang Lu, Zhuoling Li, Mi Yan, Yujia Zhang, Zhizheng Zhang, He Wang, and Hengshuang Zhao. Any3d-vla: Enhancing vla robustness via diverse point clouds. In *ICML*, 2026. 1, 3

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, et al. Libero-plus: In-depth robustness analysis of vision-language-action models. *arXiv preprint arXiv:2510.13626*, 2025. 2, 6, 29

NVIDIA GEAR. GR00T N1.5: An improved open foundation model for generalist humanoid robots. https://research.nvidia.com/labs/gear/gr00t-n1_5/, June 2025a. 30

NVIDIA GEAR. GR00T N1.6: An improved open foundation model for generalist humanoid robots. https://research.nvidia.com/labs/gear/gr00t-n1_6/, December 2025b. 1, 28, 30

Javier Alejandro Lopetegui Gonzalez, Paul Pacaud, and Cordelia Schmid. Spatially aware world action model via geometric latent diffusion. *arXiv preprint arXiv:2609.02531*, 2026. 3, 7, 30

Jun Guo, Qiwei Li, Peiyan Li, Zilong Chen, Nan Sun, Yifei Su, Heyun Wang, Yuan Zhang, Xinghang Li, and Huaping Liu. Unified 4d world action modeling from video priors with asynchronous denoising. *arXiv preprint arXiv:2604.26694*, 2026. 3, 6, 30

Jisang Han, Seonghu Jeon, Jaewoo Jung, René Zurbrügg, Honggyu An, Tifanny Portela, Marco Hutter, Marc Pollefeys, Seungryong Kim, and Sunghwan Hong. Geometric action model for robot policy learning. *arXiv preprint arXiv:2606.17046*, 2026. 3, 7, 28, 29, 30

Junjie He, Junfeng Li, Zhide Zhong, Haodong Yan, Ruixin Li, Yangyang Zheng, Jiaguan Zhu, Tianran Zhang, Yuqiao Du, Wen Chen, et al. Sg-wam: Text-grounded and spatial-aware semantic guidance for world-action models. *arXiv preprint arXiv:2608.08839*, 2026. 3

Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. 3d-llm: Injecting the 3d world into large language models. *Advances in Neural Information Processing Systems*, 2023. 3

Weikai Huang, Jieyu Zhang, Sijun Li, Taoyang Jia, Jiafei Duan, Yunqian Cheng, Jaemin Cho, Matthew Wallingford, Rustin Soraki, Chris Dongjoo Kim, Shuo Liu, Donovan Clay, Taira Anderson, Winson Han, Ali Farhadi, Bharath Hariharan, Zhongzheng Ren, and Ranjay Krishna. Wilddet3d: Scaling promptable 3d detection in the wild, 2026a. URL https://arxiv.org/abs/2604.08626. 5, 20

Wenlong Huang, Yu-Wei Chao, Arsalan Mousavian, Ming-Yu Liu, Dieter Fox, Kaichun Mo, and Li Fei-Fei. PointWorld: Scaling 3d world models for in-the-wild robotic manipulation. *arXiv preprint arXiv:2601.03782*, 2026b. 3, 5, 19

Dongyoon Hwang, Byungkun Lee, Dongjin Kim, Hyojin Jang, Hoiyeong Jin, Jueun Mun, Minho Park, Hojoon Lee, Hyunseung Kim, and Jaegul Choo. 3d hamster: Bridging planning and control in hierarchical vision language action models through 3d trajectory guidance. *arXiv preprint arXiv:2606.31329*, 2026. 3

Xiaogang Jia, Qian Wang, Anrui Wang, Han A. Wang, Balázs Gyenes, Emiliyan Gospodinov, Xinkai Jiang, Ge Li, Hongyi Zhou, Weiran Liao, Xi Huang, Maximilian Beck, Moritz Reuss, Rudolf Lioutikov, and Gerhard Neumann. Pointmappolicy: Structured point cloud processing for multimodal imitation learning. In *Advances in Neural Information Processing Systems*, 2025. 1, 3

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. DROID: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024. 5, 19

Dongyoung Kim, Huiwon Jang, Myungkyu Koo, Suhyeok Jang, Taeyoung Kim, Beomjun Kim, Byungjun Yoon, Changsung Jang, Daewon Choi, Dongsu Han, et al. Rldx-1 technical report. *arXiv preprint arXiv:2605.03269*, 2026a. 7, 28, 29, 30

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 *Conference on Robot Learning*, 2024. 1, 28, 29

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *Robotics: Science and Systems*, 2025. 28, 29

Moo Jin Kim, Yihuai Gao, Tsung-Yi Lin, Yen-Chen Lin, Yunhao Ge, Grace Lam, Percy Liang, Shuran Song, Ming-Yu Liu, Chelsea Finn, et al. Cosmos policy: Fine-tuning video models for visuomotor control and planning. *International Conference on Learning Representations*, 2026b. 7, 28, 29, 30

Lingyu Kong, Ruicheng Li, Ruicheng Wang, Sicheng Xu, Chengtang Yao, Jianfeng Xiang, and Jiaolong Yang. Fine-detail monocular geometry estimation with self-guided sparse volumetric refinement. *arXiv preprint arXiv:2607.17967*, 2026. 9

Byungkun Lee, Dongyoon Hwang, Dongjin Kim, Hojoon Lee, Hyunseung Kim, Jaegul Choo, and Minho Park. See like a robot: Robot-centric pointmaps for vision-language-action models. *arXiv preprint arXiv:2607.11498*, 2026. 1, 3, 7, 30

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. LLaVA-OneVision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024. 21

Chengmeng Li, Junjie Wen, Yaxin Peng, Yan Peng, and Yichen Zhu. Pointvla: Injecting the 3d world into vision-language-action models. *IEEE Robotics and Automation Letters*, 11(3):2506–2513, 2026a. 1, 3

Fuhao Li, Wenxuan Song, Han Zhao, Jingbo Wang, Pengxiang Ding, Donglin Wang, Long Zeng, and Haoang Li. Spatial forcing: Implicit spatial representation alignment for vision-language-action model. In *International Conference on Learning Representations*, 2026b. 1, 3, 6, 7, 28, 29

Hao Li, Zi-han Ding, Shuai Yang, Yilun Chen, Yang Tian, Xiaolin Hu, Tai Wang, Dahua Lin, Feng Zhao, Si Liu, et al. Robointer: A holistic intermediate representation suite towards robotic manipulation. In *International Conference on Learning Representations*, 2026c. 5, 20

Shuang Li, Yihuai Gao, Dorsa Sadigh, and Shuran Song. Unified video action model. *arXiv preprint arXiv:2503.00200*, 2025. 30

Huizhi Liang, Yichao Shen, Yu Deng, Sicheng Xu, Zhiyuan Feng, Tong Zhang, Yaobo Liang, and Jiaolong Yang. HiSpatial: Taming hierarchical 3d spatial understanding in vision-language models. *arXiv preprint arXiv:2603.25411*, 2026. 1, 3, 5, 20, 32

Yue Liao, Pengfei Zhou, Siyuan Huang, Donglin Yang, Shengcong Chen, Yuxin Jiang, Yue Hu, Si Liu, Jianlan Luo, Liliang Chen, et al. Genie envisioner: A unified world foundation platform for robotic manipulation. In *International Conference on Learning Representations*, 2026. 28

Haitao Lin, Hanyang Yu, Jingshun Huang, He Zhang, Yonggen Ling, Ping Tan, Xiangyang Xue, and Yanwei Fu. Posevla: Universal pose pretraining for generalizable vision-language-action policies. *Robotics: Science and Systems*, 2026a. 3, 6, 33

Tao Lin, Yuxin Du, Jiting Liu, Nuobei Zhu, Yunhe Li, Yuqian Fu, Yinxinyu Chen, Hongyi Cai, Zewei Ye, Bing Cheng, et al. Evo-depth: A lightweight depth-enhanced vision-language-action model. *arXiv preprint arXiv:2605.14950*, 2026b. 28, 29

Xuewu Lin, Tianwei Lin, Yun Du, Hongyu Xie, Yiwei Jin, Jiawei Li, Shijie Wu, Qingze Wang, Mengdi Li, Mengao Zhao, et al. Holobrain-0 technical report. *arXiv preprint arXiv:2602.12062*, 2026c. 28, 29

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 2023. 2, 6, 28

Haoxiang Ma, Junhao Cai, Xiaoxu Xu, Hao Li, Yuyin Yang, Yang Tian, Jiafei Cao, Hongrui Zhu, Zherui Qiu, Yuqiang Yang, et al. Internvla-a1. 5: Unifying understanding, latent foresight, and action for compositional generalization. *arXiv preprint arXiv:2607.04988*, 2026. 4, 7, 28, 29

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. Robocasa: Large-scale simulation of everyday tasks for generalist robots. In *Robotics: Science and Systems*, 2024. 2, 6, 30

Nikolay Nikolov, Giuliano Albanese, Sombit Dey, Aleksandar Yanev, Luc Van Gool, Jan-Nico Zaech, and Danda Pani Paudel. Spear-1: Scaling beyond robot demonstrations via 3d understanding. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2026. 3

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. FAST: Efficient action tokenization for vision-language-action models. *Robotics: Science and Systems*, 2025. 4, 28, 29, 30

Jingjing Qian, Boyao Han, Chen Shi, Lei Xiao, Long Yang, Shaoshuai Shi, and Li Jiang. Geopredict: Leveraging predictive kinematics and 3d gaussian geometry for precise vla manipulation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2026. 3, 7, 30

Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Yan Ding, Zhigang Wang, JiaYuan Gu, Bin Zhao, Dong Wang, et al. Spatialvla: Exploring spatial representations for visual-language-action model. *arXiv preprint arXiv:2501.15830*, 2025. 1, 3

Jiaxin Shi, Xidong Zhang, Fucai Zhu, Zhe Li, Siyu Zhu, and Weihao Yuan. 3dthinkvla: Endowing vision-language-action models with latent 3d priors via 3d-thinking-guided co-training. *arXiv preprint arXiv:2606.04436*, 2026. 3

Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025. 9

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 *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15768–15780. IEEE, 2025. 3

Jingwen Sun, Wenyao Zhang, Zekun Qi, Shaojie Ren, Zezhi Liu, Hanxin Zhu, Guangzhong Sun, Xin Jin, and Zhibo Chen. Vla-jepa: Enhancing vision-language-action model with latent world model. In *European Conference on Computer Vision*, 2026. 29

Lin Sun, Bin Xie, Yingfei Liu, Hao Shi, Tiancai Wang, and Jiale Cao. Geovla: Empowering 3d representations in vision-language-action models. *arXiv preprint arXiv:2508.09071*, 2025. 1, 3

Huajie Tan, Enshen Zhou, Zhiyu Li, Yijie Xu, Yuheng Ji, Xiansheng Chen, Cheng Chi, Pengwei Wang, Huizhu Jia, Yulong Ao, et al. Robobrain 2.5: Depth in sight, time in mind. *arXiv preprint arXiv:2601.14352*, 2026. 3

Shuhan Tan, Kairan Dou, Yue Zhao, and Philipp Krähenbühl. Interactive post-training for vision-language-action models. *arXiv preprint arXiv:2505.17016*, 2025. 28, 29

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5. 1, 6, 9, 17, 33

Xiaomi Robotics Team, Jun Guo, Piaopiao Jin, Jason Li, Peiyan Li, Yingyan Li, Futeng Liu, Wanli Peng, Optimus Qin, Yifei Su, et al. Xiaomi-robotics-1: Scaling vision-language-action models with over 100k hours of real-world trajectories. *arXiv preprint arXiv:2607.15330*, 2026. 7, 30

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, et al. Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025. 9

An Dinh Vuong, Tuan Van Vo, Abdullah Sohail, Haoran Ding, Liang Ma, Xiaodan Liang, Anqing Duan, Ivan Laptev, and Ian Reid. World2act: Latent action post-training from world model dynamics. *arXiv preprint arXiv:2603.10422*, 2026. 30

Bowen Wen, Shaurya Dewan, and Stan Birchfield. Fast-foundationstereo: Real-time zero-shot stereo matching. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2026. 18, 33

Ge Yan, Jinghao Liu, Yuzhi Fan, Lei Cai, Minwen Liao, Jesse Zhang, and Dieter Fox. Flex-$\pi$: A multi-stream world-action model with compute flexibility. *arXiv preprint arXiv:2608.10860*, 2026. 3, 6, 7, 28, 29

Rujia Yang, Geng Chen, Chuan Wen, and Yang Gao. Fp3: A 3d foundation policy for robotic manipulation. *arXiv preprint arXiv:2503.08950*, 2025. 3, 6, 33

Yandan Yang, Shuang Zeng, Tong Lin, Xinyuan Chang, Dekang Qi, Junjin Xiao, Haoyun Liu, Ronghan Chen, Yuzhi Chen, Dongjie Huo, et al. Abot-m0: Vla foundation model for robotic manipulation with action manifold learning. *arXiv preprint arXiv:2602.11236*, 2026. 7, 28, 29

Jinhui Ye, Fangjing Wang, Ning Gao, Junqiu Yu, Yangkun Zhu, Bin Wang, Jinyu Zhang, Weiyang Jin, Yanwei Fu, Feng Zheng, et al. St4vla: Spatially guided training for vision-language-action models. *International Conference on Learning Representations*, 2026a. 4, 6, 21

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, et al. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026b. 7, 30

Haoqi Yuan, Zhixuan Liang, Anzhe Chen, Ye Wang, Haoyang Li, Pei Lin, Yiyang Huang, Zixing Lei, Tong Zhang, Jiazhao Zhang, et al. Qwen-robotmanip technical report: Alignment unlocks scale for robotic manipulation foundation models. *arXiv preprint arXiv:2606.17846*, 2026a. 7, 28, 29

Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-wam: Do world action models need test-time future imagination? *arXiv preprint arXiv:2603.16666*, 2026b. 28, 29

Mengqi Zhang, Sahil Khose, Simar Kareer, Yuchen Song, Unnat Jain, and Judy Hoffman. Deva: Decoupled video-action model with physical guidance for robot policy learning. *arXiv preprint arXiv:2607.24159*, 2026a. 7, 30

Yuyang Zhang, Wenyao Zhang, Zekun Qi, He Zhang, Haitao Lin, Jingbo Zhang, Yao Mu, Xiaokang Yang, Wenjun Zeng, and Xin Jin. Imagewam: Do world action models really need video generation, or just image editing? *arXiv preprint arXiv:2606.19531*, 2026b. 28, 29

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, et al. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. In *International Conference on Learning Representations*, 2026a. 28

Yupeng Zheng, Xiang Li, Songen Gu, Yuhang Zheng, Shuai Tian, Weize Li, Linbo Wang, Chaoyue Li, Qichao Zhang, Haoran Li, et al. Gift: Guided intermediate feature training via action-oriented structural supervision for robotic manipulation. *arXiv preprint arXiv:2609.04193*, 2026b. 3, 7, 28, 29

Linqing Zhong, Yi Liu, Yifei Wei, Ziyu Xiong, Si Liu, and Guanghui Ren. Acot-vla: Action chain-of-thought for vision-language-action models. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2026. 28, 29

Enshen Zhou, Jingkun An, Cheng Chi, Yi Han, Shanyu Rong, Chi Zhang, Pengwei Wang, Zhongyuan Wang, Tiejun Huang, Lu Sheng, et al. Roborefer: Towards spatial referring with reasoning in vision-language models for robotics. *Advances in Neural Information Processing Systems*, 38:28404–28481, 2025a. 3

Enshen Zhou, Cheng Chi, Yibo Li, Jingkun An, Jiayuan Zhang, Shanyu Rong, Yi Han, Yuheng Ji, Mengzhen Liu, Pengwei Wang, et al. Robotracer: Mastering spatial trace with reasoning in vision-language models for robotics. *arXiv preprint arXiv:2512.13660*, 2025b. 3

Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 17

Chuning Zhu, Raymond Yu, Siyuan Feng, Benjamin Burchfiel, Paarth Shah, and Abhishek Gupta. Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets. *arXiv preprint arXiv:2504.02792*, 2025. 7, 30

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

# APPENDIX


**A Implementation and Training Details** **17**

A.1 State and Action Representation . . . . . 17

A.2 Architecture Details . . . . . 17

A.3 Prompt and Supervision Format . . . . . 17

A.4 Pretraining Details . . . . . 17

A.5 Downstream Implementation . . . . . 18

A.6 Inference . . . . . 18

**B Training Data Construction Details** **19**

B.1 Pointmap-Augmented Robot Demonstrations . . . . . 19

B.2 Spatial and Embodied QA Construction . . . . . 19

B.3 Reference-Frame Alignment . . . . . 21

B.4 Qualitative QA Examples . . . . . 21

**C Simulation Benchmarks** **28**

C.1 LIBERO . . . . . 28

C.2 LIBERO-Plus . . . . . 29

C.3 RoboCasa . . . . . 30

C.4 SpatialRGPT-Bench (quantitative) . . . . . 32

C.5 Qualitative 3D Grounding . . . . . 33

**D Real-World Experiments** **33**

D.1 Experimental Setup . . . . . 33

D.2 Fine-Tuning and Deployment . . . . . 33

D.3 Evaluation Protocol . . . . . 33

D.4 Detailed Results . . . . . 34

# A Implementation and Training Details

Here, we provide additional implementation details, including state and action representations, architecture specifications, prompt formats, and training hyperparameters.

## A.1 State and Action Representation

During pretraining, both the robot state and action use a 17-dimensional representation consisting of 3D end-effector translation, 6D rotation (Zhou et al., 2019), one gripper value, and seven joint positions. The state is expressed in absolute coordinates, while actions are represented as relative end-effector and joint commands with respect to the state at the beginning of each action chunk. Specifically, translation and joint positions are encoded as offsets, rotation as a relative 6D orientation in the robot-base frame, and the gripper command remains absolute. The policy predicts chunks of $H = 16$ actions. To account for differences across robot datasets, actions are normalized per dataset using the 1st and 99th percentiles and clipped to $[-2.2, 2.2]$. FAST tokenization further clips the normalized actions to $[-1, 1]$, while robot states are normalized using per-dataset statistics.

## A.2 Architecture Details

We use Qwen3.5-4B (Team, 2026) as the vision-language backbone. The pointmap encoder follows the same architecture as the pretrained RGB visual encoder and takes raw metric XYZ coordinates without additional centering or normalization. For continuous control, we use a 32-layer DiT action expert with hidden width 1024 and 16 attention heads. Each DiT layer is conditioned on the corresponding language-model hidden states through a learned $2560 \to 1024$ projection. The expert takes one robot-state token followed by 16 action tokens as input. All components are trained end-to-end throughout pretraining.

## A.3 Prompt and Supervision Format

All samples follow the Qwen3.5 chat template with thinking disabled. Robot samples use

```
Task: {instruction} Mode: <end_effector> Pointmap_origin: <robot>
```

while pointmap-paired VQA samples use

```
Task: {question} Mode: <vqa> Pointmap_origin: <robot|camera>.
```

The origin tag indicates the reference frame in which the pointmap and coordinate-based targets are expressed. RGB-only VQA samples omit this field.

For the discrete action objective used during pretraining, robot samples append the discretized robot state and FAST-encoded action chunk to the assistant sequence as

$$\texttt{[STATE]}\ b_1\ \cdots\ b_{17}\ \texttt{[ACTION]}\ \texttt{<FAST tokens>},$$

where $b_i$ denotes the 256-bin index of the normalized state component. The state prefix is provided as conditioning context but is masked from the next-token loss, so only the FAST action tokens are supervised. For VQA samples, supervision is applied only to the answer tokens.

During Stage 2, the continuous action expert additionally conditions on the VLM representations of the visual observations and instruction, with the robot state provided through the expert's state embedding. The teacher-forced state prefix and FAST targets are excluded from the expert context to prevent access to the target actions during continuous action prediction.

## A.4 Pretraining Details

**Data Mixture.** We pretrain PointmapVLA for 50K updates in each stage. Stage 1 uses 112 robot and 256 VQA samples per update, placing greater emphasis on spatial supervision. Stage 2 reverses the mixture to 256 robot and 112 VQA samples as training shifts toward continuous action learning. Figure 6 shows the source-level composition used in each stage. Further details on dataset construction and statistics are provided in Appendix B.

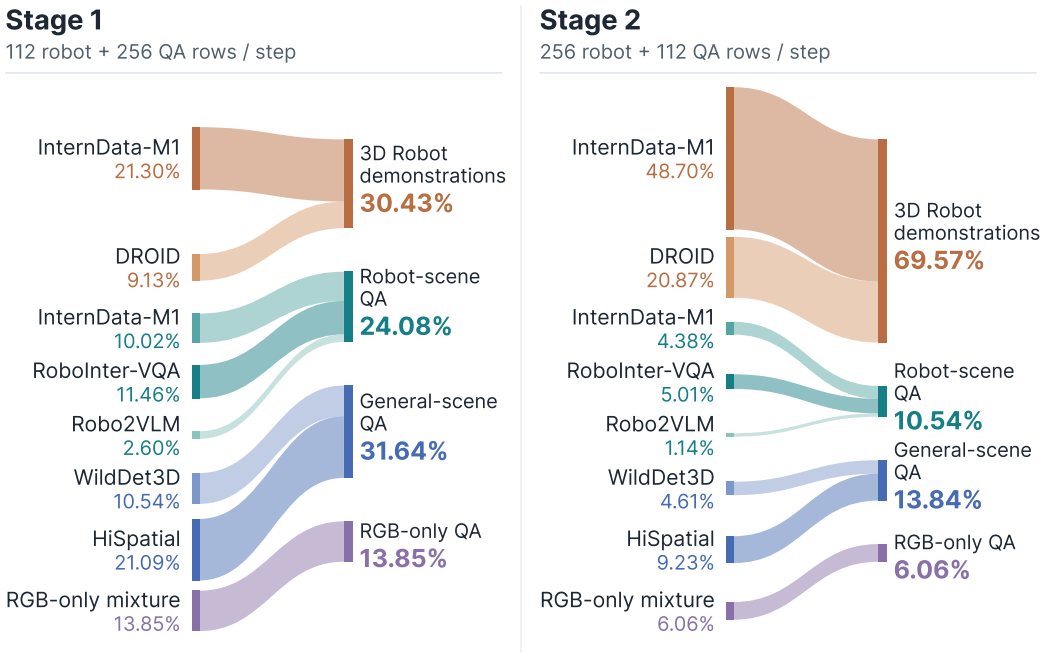



Figure 6: **Training data mixture across the two pretraining stages.** Stage 1 emphasizes VQA supervision, while Stage 2 shifts toward robot demonstrations. Percentages denote the sample share of each source.

**Optimization.** We train each pretraining stage for 50K updates. The backbone and pointmap encoder use a learning rate of 1e-5 in both stages, while the action expert and VLM projections introduced in Stage 2 use 1e-4. We apply linear warmup for 2.5K and 10K updates in Stages 1 and 2, respectively, followed by a constant learning rate. Both stages use AdamW with $\beta = (0.9, 0.999)$, weight decay 1e-2, and gradient clipping at 1.0, together with bfloat16 precision, DeepSpeed ZeRO-2, and FlashAttention-2. During training, we apply RGB color jitter with brightness 0.3, contrast 0.4, saturation 0.5, and hue 0.08, using the same augmentation across all views while leaving pointmaps unchanged, and randomly permute the camera views while preserving RGB–pointmap correspondence. To improve robustness to missing modalities, we remove all pointmaps for 10% of robot samples and remove the RGB features of one view for another 10% while retaining its pointmap.

### A.5 Downstream Implementation

**Fine-Tuning.** We fine-tune PointmapVLA for 30K steps on LIBERO and 20K steps on RoboCasa with a global batch size of 128. For real-world experiments, we collect 60 demonstrations per task across three tasks (180 in total) and fine-tune all models for 50K steps with a global batch size of 64.

**Observation Processing.** In simulation, we construct robot-base pointmaps using ground-truth metric depth and camera extrinsics provided by the simulator. For real-world experiments, we estimate depth from stereo images using Fast-FoundationStereo (Wen et al., 2026) and obtain camera extrinsics through calibration.

### A.6 Inference

At inference, the VLM context is computed once from the visual observations and language instruction. The continuous action expert integrates the predicted flow using 4 Euler steps and outputs a 16-step action chunk. The executed horizon is specified separately for each evaluation protocol.

Table 8: **Pretraining dataset statistics.** Counts are rounded and reported after preprocessing, before training-time filtering; K and M denote thousands and millions, respectively. HiSpatial uses answer-dependent reference frames. Sampling proportions are provided in Appendix A.4.

*(a) Robot demonstrations*

| **Dataset** | **Domain** | **Episodes** | **Frames** |
|---|---|---|---|
| InternData-M1 | Sim | 176.6K | 67.49M |
| DROID | Real | 33.7K | 8.84M |
| **Total** | | **210.3K** | **76.33M** |

*(b) Visual Question answering*

| **Dataset** | **Reference frame** | **QA pairs** |
|---|---|---|
| *Robot-scene QA* | | |
| InternData-M1 | Robot base | 3.69M |
| RoboInter-VQA | Robot base | 878.3K |
| Robo2VLM | Robot base | 199.4K |
| *General-scene QA* | | |
| WildDet3D | Pseudo-base | 994.1K |
| HiSpatial | Pseudo-base / camera origin | 4.91M |
| *Total (pointmap-paired)* | | *10.67M* |
| Supplementary RGB-only QA | — | 4.09M |
| **Total** | | **14.76M** |

# B Training Data Construction Details

Our pretraining data is composed of pointmap-augmented robot demonstrations for action learning and spatial VQA for scene understanding. The QA stream combines *robot-scene QA* from InternData-M1, RoboInter-VQA, and Robo2VLM with *general-scene QA* from WildDet3D and HiSpatial, supplemented by RGB-only QA. Table 8 reports the available training inventory: 210,349 robot episodes (76.3M frames), 10.7M pointmap-paired QA examples, and 4.09M RGB-only QA examples.

## B.1 Pointmap-Augmented Robot Demonstrations

Robot demonstrations pair manipulation instructions and action targets with three camera views. We use the physical robot-base frame pointmaps, providing a shared spatial reference across views.

**InternData-M1.** InternData-M1 (Chen et al., 2025c) provides simulated trajectories, scene assets, object poses, and camera parameters. We retain the original RGB observations and re-render metric depth and instance masks in Isaac Sim. Episodes with invalid task-relevant object annotations are excluded. The resulting geometry supports both action learning and the robot-scene QA described below.

**DROID.** We use DROID (Khazatsky et al., 2024) episodes with refined depth from PointWorld-DROID (Huang et al., 2026b), paired with their RGB observations and robot states. Wrist-camera poses use an episode-specific calibration correction, with a calibration-session estimate as fallback. The action stream includes a wrist pointmap only when its held-out alignment residual against exterior-camera geometry is at most 20 mm; otherwise, wrist RGB is retained and its pointmap is masked.

## B.2 Spatial and Embodied QA Construction

We utilize source annotations along with self-generated additional geometric targets using scene geometry and recorded robot states. Source-specific preprocessing procedures are described below.

Figure 3 illustrates the supervision families. Pointmap-paired QA retains text and image-space answers alongside 3D targets. At training time, its RGB and pointmap inputs are aligned on a $320 \times 256$ letterbox canvas.

### B.2.1 ROBOT-SCENE QA

**InternData-M1.** We generate six task families from the original 3D object annotations, manipulation instructions, and end-effector states: detection of all graspable objects, referring 3D detection, end-effector–object distances, nearest-object queries, instruction-conditioned pick targets, and instruction-conditioned place targets. Each example supplies three camera views, with 3D boxes expressed in the robot-base frame. The QA split reserves 7,420 examples from 354 episodes, including a 2,097-example test subset from 300 episodes. These episodes are excluded from QA training but their demonstrations remain in the action stream.

**RoboInter-VQA.** We select the DROID training subset of RoboInter-VQA (Li et al., 2026c). A representative image identifies the corresponding DROID episode; the episode's QA examples then use their declared frame indices to retrieve depth and robot states. Single-image questions use an exterior camera, while clip questions retain images 0, 4, and 7 of the original eight-image sequence in source order. We preserve text answers and normalize 2D box, point, and trajectory coordinates to the 0–1000 convention.

We additionally construct 3D QA examples from this source, retaining 166,439 examples after task-specific consistency filtering. Object and contact-region boxes are lifted through the pointmap into axis-aligned boxes of the observed surface, and contact points become 3D points. Gripper boxes and future trajectories instead use recorded end-effector poses, with the source 2D waypoints providing trajectory alignment. These five tasks supplement the 711,812 retained source QA examples, yielding 878,251 examples in total.

**Robo2VLM.** We separately preprocess the DROID subset of Robo2VLM (Chen et al., 2025a). Because source frame indices are unavailable, single images are retrieved against the two exterior-camera recordings, and composite images are split and matched panel by panel against exterior and wrist views. Retained multi-camera panels belong to one episode and have matched-frame spreads of at most 15 frames; each panel uses the depth and geometry of its own matched frame. Existing answers cover trajectory direction, task success, object relations, and instruction matching. We replace camera-relative depth-ordering questions with robot-base- or gripper-relative questions, recomputing point distances from the paired geometry. This produces 199,385 examples. QA wrist panels are retained when a valid camera pose is available.

### B.2.2 GENERAL-SCENE QA

**WildDet3D.** WildDet3D (Huang et al., 2026a) supplies 3D boxes, MoGe-2 depth, and WildCamera intrinsics. We use these released depth and intrinsics to construct pointmaps, and estimate gravity with GeoCalib. The resulting QA covers referring 3D detection, lifting a specified 2D box to its annotated 3D box, and querying a pixel's 3D location. Box targets come from the source annotations; point-query targets are computed from depth and intrinsics. Geometry and targets are transformed into a shared pseudo-base frame for each image.

**HiSpatial.** From HiSpatial (Liang et al., 2026), we select seven task families over Objects365 and CA-1M images: point querying, depth ordering, localization, size, distance, direction, and comparison. Processing depends on answer form. We retain source scalar, choice, and text answers; generate point-query and depth-ordering QA from metric depth; and transform 3D points, boxes, and direction vectors into pseudo-base coordinates. Depth-ordering labels are recomputed by comparing the camera-frame depths of two sampled points. Scalar, choice, and text examples use an upright camera-origin frame, with pitch-based alignment where available and a level-camera assumption otherwise. We create an image-disjoint validation split, leaving 4,905,218 training examples.

### B.2.3 SUPPLEMENTARY RGB-ONLY QA

We reconstruct the Stage-1 co-training source list of ST4VLA (Ye et al., 2026a) from publicly released datasets, yielding 4,085,926 examples of general VQA and captioning, 2D box grounding, pointing, and trajectory QA. This supplementary mixture includes LLaVA-OneVision subsets (Li et al., 2024), with per-source sizes determined by our reconstruction. These examples receive no pointmap; images retain their aspect ratio with the longest side at most 320 pixels.

## B.3 REFERENCE-FRAME ALIGNMENT

Robot demonstrations and robot-scene QA use physical robot-base frames. For general-scene QA with 3D coordinate or vector answers, we transform pointmaps and their targets into randomized pseudo-base frames. This varies the spatial reference while preserving scene geometry, encouraging the model to interpret spatial targets in the coordinate frame specified by the pointmap.

WildDet3D uses one transform for all QA examples associated with an image, whereas HiSpatial samples a transform for each coordinate- or vector-answer example. Points and boxes receive the full rigid transform, while direction vectors receive only its rotation. HiSpatial's scalar, choice, and text examples instead use upright camera-origin geometry.

Origin tags distinguish camera-origin from robot-style-origin geometry, with the latter covering both physical robot-base and pseudo-base frames. Coordinate and vector answers are expressed in the pointmap frame. Source text answers are retained where applicable; RoboInter-VQA image-space coordinates are normalized to the 0–1000 convention.

Representative robot-scene and general-scene QA examples are collected in Appendix B.4, with RGB observations, pointmaps, and visualizations of the scene geometry and selected spatial targets.

## B.4 QUALITATIVE QA EXAMPLES

Figures 7–9 illustrate robot-scene QA, and Figures 10–12 illustrate general-scene QA. Each figure presents RGB observations, pointmaps, and representative question–answer pairs, alongside a point-cloud visualization of the scene geometry and selected spatial targets.

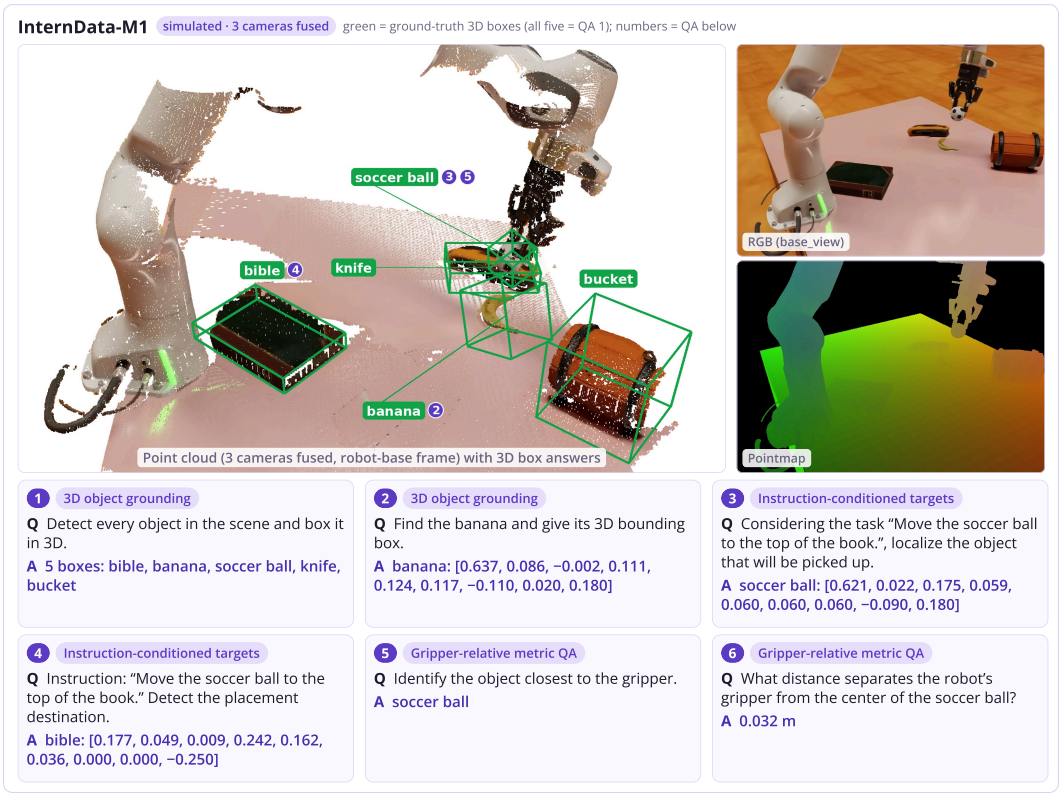



Figure 7: **Robot-scene QA examples from InternData-M1.** Examples cover all-object and referring 3D grounding, instruction-conditioned pick and place targets, nearest-object queries, and gripper–object distance estimation. The point-cloud visualization fuses three camera views in the robot-base frame, with green boxes indicating 3D object targets.

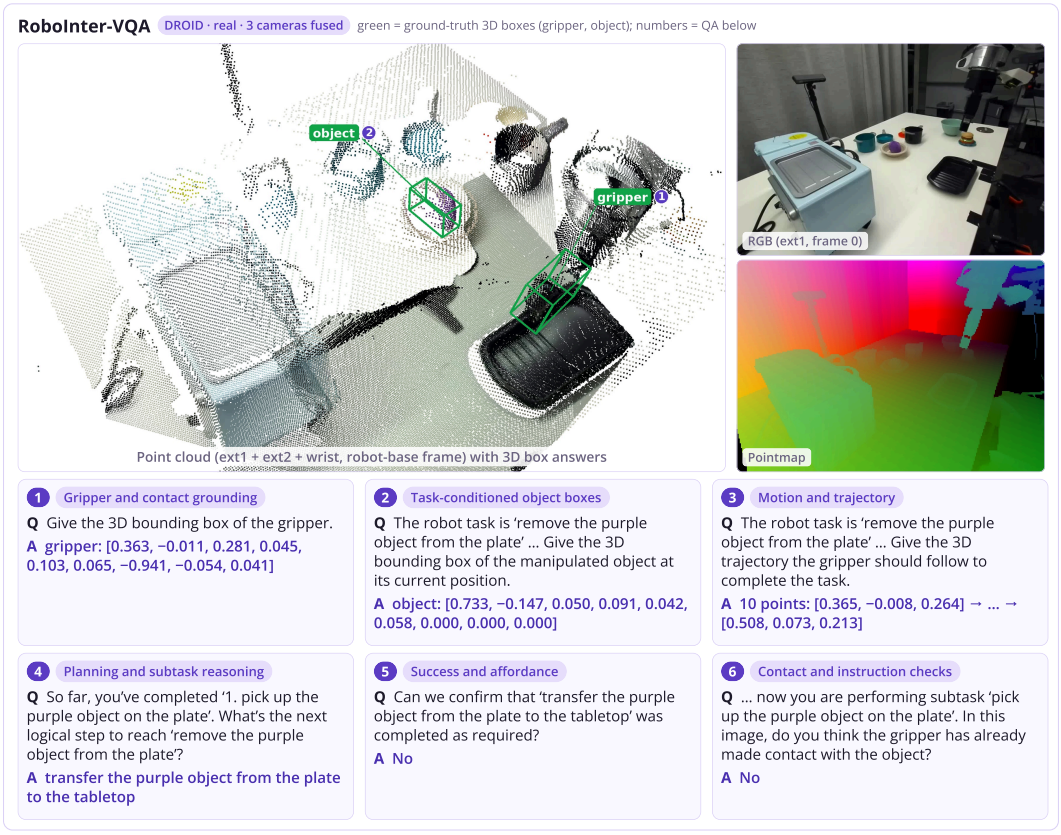



Figure 8: **Robot-scene QA examples from the DROID subset of RoboInter-VQA.** Examples include 3D gripper and manipulated-object boxes, a gripper trajectory, next-subtask reasoning, task-completion assessment, and gripper–object contact checking. The point cloud and 3D box targets are expressed in the robot-base frame.

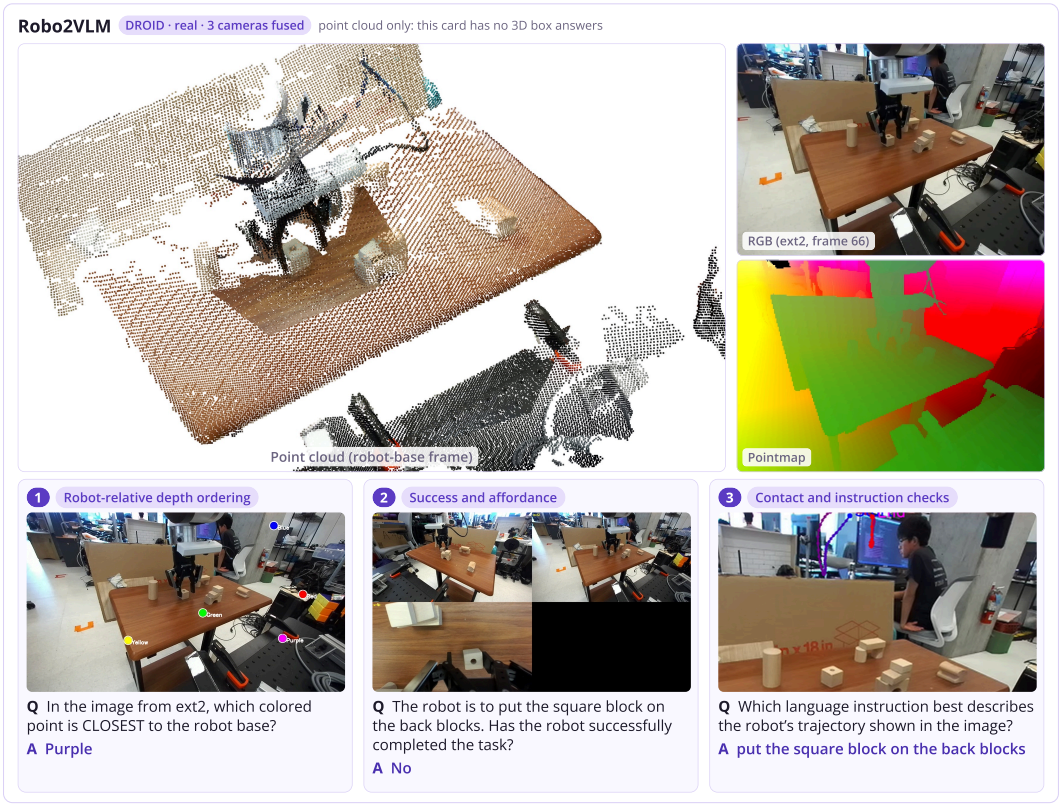



Figure 9: **Robot-scene QA examples from the DROID subset of Robo2VLM.** Examples combine robot-base-relative point ordering, task-completion assessment, and matching a depicted trajectory to a language instruction. Each question retains its corresponding image or multi-camera panel; the point-cloud visualization fuses three camera views in the robot-base frame.

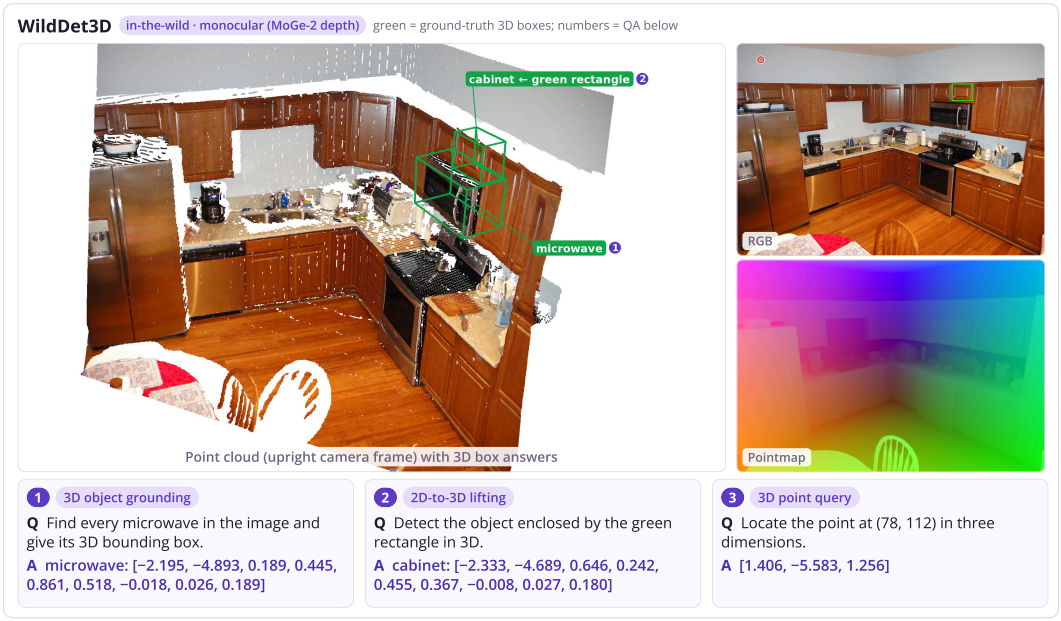



Figure 10: **General-scene QA examples from WildDet3D.** Examples illustrate referring 3D object grounding, lifting a specified 2D box to a 3D box, and querying the 3D position of an image point. Green boxes visualize the 3D object targets, with scene geometry displayed in an upright camera frame.

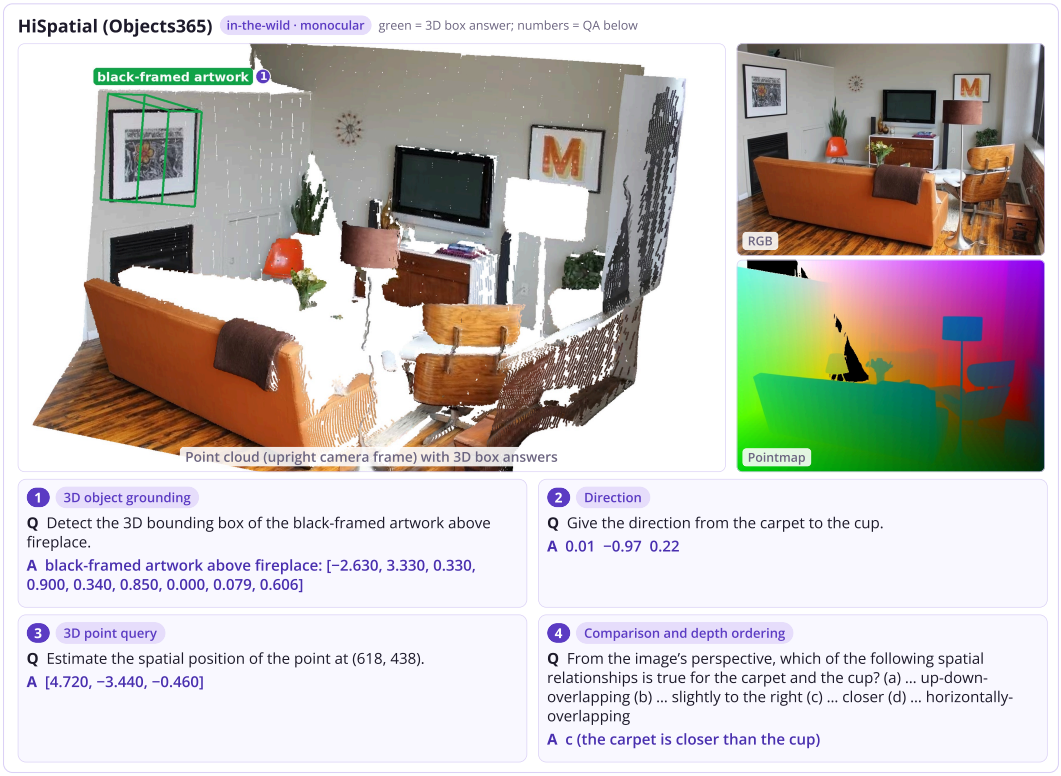



Figure 11: **General-scene QA examples from HiSpatial on Objects365 images.** Examples cover referring 3D object grounding, object-to-object direction, 3D point querying, and relative depth comparison. The scene and 3D box target are visualized in an upright camera frame.

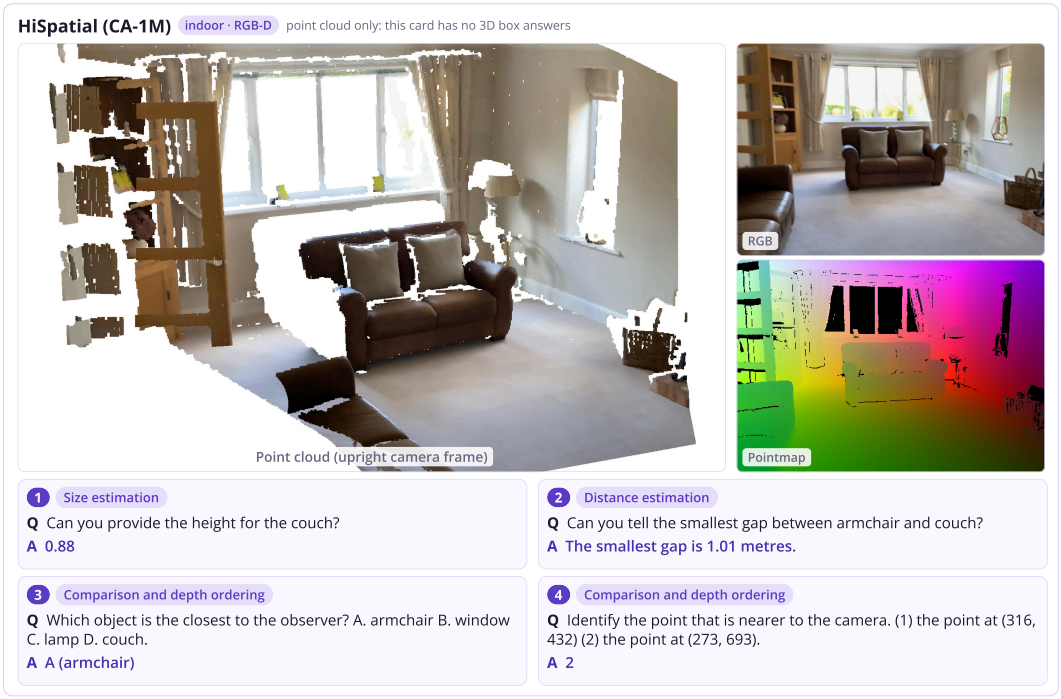



Figure 12: **General-scene QA examples from HiSpatial on CA-1M images.** Examples cover object height, inter-object distance, nearest-object selection, and point-based depth ordering. The point cloud shows the scene in an upright camera frame, while the QA examples provide scalar, textual, and choice answers.

## C Simulation Benchmarks

We report additional baseline comparisons and per-category results on the LIBERO suites, LIBERO-Plus perturbations, RoboCasa tasks, and the quantitative split of SpatialRGPT-Bench, followed by qualitative 3D grounding comparisons.

**Simulation Evaluation Protocol.** For RoboCasa, we fine-tune on 300 demonstrations per task across 24 tasks (7,200 demonstrations in total). We evaluate PointmapVLA on LIBERO and RoboCasa over 50 episodes per task and report average success rates. LIBERO-Plus is evaluated without additional fine-tuning using the LIBERO-fine-tuned checkpoints.

### C.1 LIBERO

Table 9 extends the main comparison with additional WAM, VLA, and geometry-aware baselines and reports per-suite success rates. Performance on LIBERO is highly saturated among recent methods, with PointmapVLA achieving consistently high success across all four suites and 99.0% on average.

Table 9: **LIBERO.** Success rates (%) on LIBERO (Liu et al., 2023).

| Method | Spatial | Object | Goal | Long | Avg. |
|---|---|---|---|---|---|
| *WAMs* | | | | | |
| GE-Act (Liao et al., 2026) | 98.2 | 97.6 | 95.8 | 94.4 | 96.5 |
| Fast-WAM (Yuan et al., 2026b) | 98.2 | **100.0** | 97.0 | 95.2 | 97.6 |
| ImageWAM (Zhang et al., 2026b) | 97.2 | 99.2 | 98.8 | 98.4 | 98.4 |
| Cosmos-Policy (Kim et al., 2026b) | 98.1 | **100.0** | 98.2 | 97.6 | 98.5 |
| ABot-M0.5 (Chen et al., 2026) | **100.0** | 99.8 | **99.4** | 98.4 | **99.4** |
| *VLAs* | | | | | |
| OpenVLA (Kim et al., 2024) | 84.7 | 88.4 | 79.2 | 53.7 | 76.5 |
| $\pi_0$-FAST (Pertsch et al., 2025) | 96.4 | 96.8 | 88.6 | 60.2 | 85.5 |
| $\pi_0$ (Black et al., 2025b) | 98.0 | 96.8 | 94.4 | 88.4 | 94.4 |
| UniVLA (Bu et al., 2025) | 96.5 | 96.8 | 95.6 | 92.0 | 95.2 |
| $\pi_{0.5}$ (Black et al., 2025a) | 98.8 | 98.2 | 98.0 | 92.4 | 96.9 |
| GR00T N1.6 (GEAR, 2025b) | 97.7 | 98.5 | 97.5 | 94.4 | 97.0 |
| OpenVLA-OFT (Kim et al., 2025) | 97.6 | 98.4 | 97.9 | 94.5 | 97.1 |
| RIPT-VLA (Tan et al., 2025) | 98.6 | 98.6 | 99.0 | 93.8 | 97.5 |
| RLDX-1 (Kim et al., 2026a) | 98.0 | 99.3 | 98.4 | 95.3 | 97.8 |
| X-VLA (Zheng et al., 2026a) | 98.2 | 98.6 | 97.8 | 97.6 | 98.1 |
| ACoT-VLA (Zhong et al., 2026) | 98.6 | 99.0 | **99.4** | 97.0 | 98.5 |
| InternVLA-A1.5 (Ma et al., 2026) | 98.6 | 99.8 | 98.6 | 98.4 | 98.9 |
| Qwen-RobotManip (Yuan et al., 2026a) | - | - | - | - | 99.1 |
| Qwen-RobotManip-Ctx (Yuan et al., 2026a) | - | - | - | - | 99.2 |
| *Geometry-aware WAMs* | | | | | |
| GAM (Han et al., 2026) | 98.6 | 99.6 | 97.4 | 94.6 | 97.6 |
| GIFT-WAM-IDM (Zheng et al., 2026b) | 99.0 | **100.0** | 98.6 | 96.4 | 98.5 |
| SG-WAM (**?**) | 99.4 | 99.4 | 98.2 | 97.8 | 98.7 |
| Flex-$\pi$ (Yan et al., 2026) | 99.6 | 99.8 | 98.6 | 96.0 | 98.5 |
| Flex-$\pi^*$ (Yan et al., 2026) | 99.2 | 99.8 | 99.0 | **98.6** | 99.2 |
| *Geometry-aware VLAs* | | | | | |
| Evo-Depth (Lin et al., 2026b) | 95.6 | 99.2 | 95.6 | 91.3 | 95.4 |
| HoloBrain-0-GD (Lin et al., 2026c) | 97.8 | 98.2 | 95.2 | 95.6 | 96.7 |
| GIFT-VLA (Zheng et al., 2026b) | 99.0 | 99.2 | 98.4 | 94.8 | 97.9 |
| Spatial Forcing (Li et al., 2026b) | 99.4 | 99.6 | 98.8 | 96.0 | 98.5 |
| ABot-M0 (Yang et al., 2026) | 98.8 | 99.8 | 99.0 | 96.6 | 98.6 |
| **PointmapVLA (Ours)** | 99.4 | 99.5 | 99.3 | 97.9 | 99.0 |

### C.2 LIBERO-PLUS

Table 10 provides the full zero-shot comparison on LIBERO-Plus across all perturbation types. Unlike standard LIBERO, LIBERO-Plus introduces substantial visual, spatial, and linguistic shifts, resulting in larger differences among methods. PointmapVLA achieves 92.4% overall success and remains particularly robust to language, lighting, background, noise, and layout perturbations.

Table 10: **LIBERO-Plus.** Zero-shot success rates (%) on LIBERO-Plus (Fei et al., 2025).

| Method | Camera | Robot | Lang. | Light | Backgr. | Noise | Layout | Total |
|---|---|---|---|---|---|---|---|---|
| *WAMs* | | | | | | | | |
| Fast-WAM (Yuan et al., 2026b) | 16.4 | 44.5 | 68.9 | 78.2 | 53.7 | 37.7 | 60.7 | 51.5 |
| VLA-JEPA (Sun et al., 2026) | 63.3 | 67.1 | 85.4 | 95.6 | 93.6 | 66.3 | 85.1 | 79.5 |
| Cosmos-Policy (Kim et al., 2026b) | 75.8 | 63.3 | 81.7 | 96.5 | 88.9 | 92.7 | 82.2 | 82.2 |
| ImageWAM (Zhang et al., 2026b) | 80.8 | 50.3 | 91.4 | 98.1 | 85.5 | 93.8 | 80.5 | 83.1 |
| ABot-M0.5 (Chen et al., 2026) | 70.5 | 87.4 | 88.6 | 94.0 | 89.7 | 75.5 | 85.2 | 83.4 |
| *VLAs* | | | | | | | | |
| OpenVLA (Kim et al., 2024) | 0.8 | 3.5 | 23.0 | 8.1 | 34.8 | 15.2 | 28.5 | 15.6 |
| UniVLA (Bu et al., 2025) | 1.8 | 46.2 | 69.6 | 69.0 | 81.0 | 21.2 | 31.9 | 42.9 |
| $\pi_0$ (Black et al., 2025b) | 13.8 | 6.0 | 58.8 | 85.0 | 81.4 | 79.0 | 68.9 | 53.6 |
| $\pi_0$-FAST (Pertsch et al., 2025) | 65.1 | 21.6 | 61.0 | 73.2 | 73.2 | 74.4 | 68.8 | 61.6 |
| RIPT-VLA (Tan et al., 2025) | 55.2 | 31.2 | 77.6 | 88.4 | 91.6 | 73.5 | 74.2 | 68.4 |
| OpenVLA-OFT (Kim et al., 2025) | 56.4 | 31.9 | 79.5 | 88.7 | 93.3 | 75.8 | 74.2 | 69.6 |
| $\pi_{0.5}$ (Black et al., 2025a) | 78.4 | 73.6 | 80.8 | 96.2 | 94.1 | 89.0 | 84.5 | 84.4 |
| InternVLA-A1.5 (Ma et al., 2026) | 83.1 | 55.1 | 86.9 | 96.4 | 98.2 | 95.6 | 85.2 | 84.8 |
| ACoT-VLA (Zhong et al., 2026) | 72.6 | 82.6 | 87.5 | 97.7 | 96.5 | 87.8 | 88.1 | 86.6 |
| RLDX-1 (Kim et al., 2026a) | 75.0 | **91.8** | 92.2 | 97.1 | 95.5 | 80.3 | 80.7 | 86.7 |
| Qwen-RobotManip (Yuan et al., 2026a) | 87.2 | 75.5 | 85.6 | 96.6 | 97.7 | 97.7 | 87.3 | 89.0 |
| Qwen-RobotManip-Ctx (Yuan et al., 2026a) | **89.9** | 83.9 | 86.5 | 98.6 | **99.9** | 97.9 | 87.5 | 91.4 |
| *Geometry-aware WAMs* | | | | | | | | |
| SG-WAM (**?**) | 54.8 | 78.3 | 81.7 | 96.5 | 88.9 | 90.7 | 85.3 | 81.3 |
| GAM (Han et al., 2026) | 83.1 | 70.0 | 84.8 | 97.2 | 94.3 | 95.3 | 79.1 | 85.5 |
| GIFT-WAM-IDM (Zheng et al., 2026b) | 78.7 | 80.5 | 90.2 | **99.0** | 89.2 | 97.2 | 83.3 | 87.8 |
| Flex-$\pi$ (Yan et al., 2026) | 89.1 | 72.7 | 84.5 | 94.6 | 96.3 | 98.9 | 87.3 | 88.6 |
| *Geometry-aware VLAs* | | | | | | | | |
| Spatial Forcing (Li et al., 2026b) | 20.1 | 13.4 | 40.9 | 29.1 | 33.4 | 25.7 | 39.3 | 29.1 |
| Evo-Depth (Lin et al., 2026b) | 47.2 | 49.2 | 78.9 | 88.1 | 76.4 | 77.6 | 69.6 | 69.6 |
| HoloBrain-0-GD (Lin et al., 2026c) | 65.5 | 58.2 | 78.7 | 88.1 | 90.3 | 66.9 | 79.5 | 74.0 |
| GIFT-VLA (Zheng et al., 2026b) | 57.7 | 63.7 | 88.6 | 98.5 | 96.9 | 82.5 | 80.3 | 79.6 |
| ABot-M0 (Yang et al., 2026) | 60.4 | 67.9 | 86.4 | 96.2 | 91.6 | 86.4 | 82.6 | 80.5 |
| **PointmapVLA (Ours)** | 88.0 | 83.7 | **92.6** | **99.0** | 98.8 | **99.3** | **88.9** | **92.4** |

### C.3 RoboCasa

Table 11 provides additional baseline comparisons on RoboCasa, while Table 12 reports the per-task success rates of PointmapVLA across the 24 atomic kitchen tasks.

We additionally report X-WAM (Guo et al., 2026) for completeness, but do not treat it as a directly comparable baseline in the main text. X-WAM includes 56,771 RoboCasa MimicGen episodes (282.4 hours) in its large-scale pretraining corpus before benchmark-specific fine-tuning. Because X-WAM uses RoboCasa data during pretraining, its result is not directly comparable to methods trained without such exposure. We therefore mark it separately in the table and do not use it as the primary baseline in the main comparison.

Table 11: **RoboCasa.** Average success rate (%) over the 24 atomic kitchen tasks (Nasiriany et al., 2024). †X-WAM uses RoboCasa data during pretraining and is therefore not directly comparable.

| Method | Avg. Success (%) |
|---|---|
| *VLAs* | |
| $\pi_{0.5}$ (Black et al., 2025a) | 62.1 |
| $\pi_0$ (Black et al., 2025b) | 62.5 |
| $\pi_0$-FAST (Pertsch et al., 2025) | 63.6 |
| GR00T N1.5 (GEAR, 2025a) | 64.1 |
| GR00T N1.6 (GEAR, 2025b) | 66.2 |
| RLDX-1 (Kim et al., 2026a) | 70.6 |
| Xiaomi-Robotics-1 (Team et al., 2026) | 74.5 |
| *WAMs* | |
| UVA (Li et al., 2025) | 50.0 |
| UWM (Zhu et al., 2025) | 60.8 |
| DreamZero (Ye et al., 2026b) | 62.4 |
| Cosmos Policy (Kim et al., 2026b) | 67.1 |
| World2Act (Vuong et al., 2026) | 72.6 |
| *Geometry-aware WAMs* | |
| GAM (Han et al., 2026) | 69.4 |
| DeVA (Zhang et al., 2026a) | 72.0 |
| SA-WAM (Gonzalez et al., 2026) | 76.6 |
| X-WAM† (Guo et al., 2026) | 79.2 |
| *Geometry-aware VLAs* | |
| GeoPredict (2026) | 52.4 |
| SeeR-VLA (2026) | 61.5 |
| **PointmapVLA** | **78.2** |

Table 12: **RoboCasa, per-task results.** Success rate (%) of PointmapVLA on each of the 24 atomic kitchen tasks.

| Task | Success (%) |
|---|---|
| CloseDoubleDoor | 96.7±1.2 |
| CloseDrawer | 97.3±1.2 |
| CloseSingleDoor | 100.0±0.0 |
| CoffeePressButton | 92.7±1.2 |
| CoffeeServeMug | 85.3±4.2 |
| CoffeeSetupMug | 46.0±2.0 |
| OpenDoubleDoor | 68.0±2.0 |
| OpenDrawer | 93.3±1.2 |
| OpenSingleDoor | 78.7±3.1 |
| PnPCabToCounter | 65.3±3.1 |
| PnPCounterToCab | 66.7±1.2 |
| PnPCounterToMicrowave | 40.0±7.2 |
| PnPCounterToSink | 81.3±2.3 |
| PnPCounterToStove | 80.7±6.1 |
| PnPMicrowaveToCounter | 44.0±3.5 |
| PnPSinkToCounter | 84.7±5.0 |
| PnPStoveToCounter | 79.3±1.2 |
| TurnOffMicrowave | 100.0±0.0 |
| TurnOffSinkFaucet | 94.7±1.2 |
| TurnOffStove | 40.7±8.3 |
| TurnOnMicrowave | 88.7±2.3 |
| TurnOnSinkFaucet | 83.3±1.2 |
| TurnOnStove | 80.7±5.0 |
| TurnSinkSpout | 88.0±2.0 |
| **Average** | **78.2**±0.9 |

Table 13: **Demonstration efficiency on RoboCasa.** Average success rates (%) over 24 tasks with different numbers of demonstrations per task. Both variants start from the same Stage 2 checkpoint, pretrained with RGB and pointmap inputs. The RGB-only variant removes the pointmap encoder before fine-tuning and uses only RGB observations during fine-tuning and evaluation. PointmapVLA retains the pointmap encoder and uses both RGB and pointmaps throughout fine-tuning and evaluation. Parentheses indicate percentage-point gains over RGB-only at the same demonstration budget.

| | Demonstrations per task | | |
|---|---|---|---|
| Input | 30 | 100 | 300 |
| RGB-only | 57.5 | 68.9 | 71.6 |
| **PointmapVLA** | **68.3** (+10.8) | **76.5** (+7.6) | **78.2** (+6.6) |

### C.3.1 Demonstration Efficiency

We investigate how retaining pointmap inputs during downstream fine-tuning affects demonstration efficiency (Table 13). Both variants are initialized from the same checkpoint after Stage 2 pretraining with RGB and pointmap inputs. For the RGB-only variant, we remove the pointmap encoder before fine-tuning while retaining the pretrained weights of the remaining modules. This variant uses only RGB observations during fine-tuning and evaluation, whereas PointmapVLA continues to use both RGB and pointmaps. Thus, RGB-only denotes the downstream input configuration; both variants share the preceding pointmap-based pretraining.

We compare three demonstration budgets on the same 24 RoboCasa tasks: MG30, MG100, and MG300, containing 30, 100, and 300 MimicGen demonstration episodes per task, respectively. Each demonstration is a complete episode comprising a sequence of observations and actions. These settings correspond to 720, 2,400, and 7,200 episodes in total. We fine-tune both variants under the same 20K-step schedule at each demonstration budget.

PointmapVLA achieves higher success at every demonstration budget. With 100 demonstrations per task, it reaches 76.5% success, exceeding RGB-only trained with 300 demonstrations (71.6%) by 4.9 percentage points while using one-third as many demonstrations. With 30 demonstrations per task, it achieves 68.3% success, just 0.6 percentage points below RGB-only trained with 100 demonstrations (68.9%). These results show that retaining pointmap inputs during fine-tuning and evaluation improves downstream demonstration efficiency under a fixed training schedule, even when both variants start from the same pointmap-pretrained checkpoint.

### C.4 SPATIALRGPT-BENCH (QUANTITATIVE)

Table 14 reports detailed results on the quantitative split of SpatialRGPT-Bench. Baseline results are taken from HiSpatial (Liang et al., 2026), with † denoting reproduced results.

We additionally report HiSpatial-3B for completeness, but do not treat it as a directly comparable baseline in the main text. PointmapVLA is trained using only the publicly released subset of the HiSpatial training data, which accounts for 24% of the data used to train HiSpatial-3B. Moreover, PointmapVLA is jointly optimized for spatial VQA and robot manipulation rather than spatial VQA alone. Because of these differences in training data and objectives, we report HiSpatial-3B as a reference and focus the main comparison on models trained under more comparable settings.

Table 14: Accuracy (%) on SpatialRGPT-Bench (quantitative). † denotes reproduced results.

| **Model** | **Input** | Width | Height | Direct Dist. | Horiz. Dist. | Vert. Dist. | **Avg.** |
|---|---|---|---|---|---|---|---|
| GPT-4o | RGB | 51.10 | 68.40 | 29.70 | 25.40 | 33.00 | 41.50 |
| GPT-5 | RGB | 59.09 | 67.66 | 29.72 | 25.21 | 13.72 | 40.47 |
| Gemini-2.5-Pro | RGB | 37.20 | 43.51 | 16.78 | 17.39 | 15.53 | 26.57 |
| PaliGemma2-3B | RGB | 20.30 | 27.07 | 20.27 | 15.57 | 22.64 | 21.18 |
| Qwen3VL-8B | RGB | 27.06 | 40.60 | 31.10 | 31.96 | 18.86 | 30.37 |
| InternVL3.5-8B | RGB | 27.81 | 41.35 | 31.08 | 31.96 | 17.92 | 30.53 |
| Qwen3.5-2B† | RGB | 43.61 | 51.13 | 32.43 | 21.31 | 8.49 | 31.39 |
| Qwen3.5-4B† | RGB | 49.62 | 66.92 | 35.14 | 30.33 | 30.19 | 42.44 |
| RoboRefer-8B-SFT | RGB-D | 17.29 | 60.15 | 49.32 | 37.70 | 28.30 | 39.25 |
| SpatialRGPT-8B | RGB-D | 48.90 | 61.70 | 45.90 | 68.00 | 56.60 | 56.22 |
| MM-Spatial-3B | RGB-D | 55.60 | <u>83.50</u> | 59.50 | <u>82.00</u> | 63.20 | 68.70 |
| HiSpatial-3B | RGB-XYZ | <u>69.17</u> | **84.96** | **79.73** | **86.89** | **75.47** | **79.28** |
| PointmapVLA | RGB-XYZ | **69.92** | 70.68 | <u>72.97</u> | 63.11 | **75.47** | <u>70.43</u> |

### C.5 Qualitative 3D Grounding

Figure 13 compares 3D bounding-box predictions of PointmapVLA, Pose-VLA (Lin et al., 2026a), and Qwen3.5-4B (Team, 2026) for two referring queries on a real-robot scene from our Pick-and-place setup. Each predicted box is projected onto the left, right, and wrist views. Because PointmapVLA predicts boxes in the robot-base frame shared by all views, a single prediction encloses the target consistently across the three cameras, whereas the baseline boxes are displaced from the target in one or more views.

## D Real-World Experiments

### D.1 Experimental Setup

We evaluate PointmapVLA on three tabletop manipulation tasks: Peg in hole, Stack blocks, and Pick-and-place with spatial instructions. The setup uses a Franka Research 3 robot, two fixed Intel RealSense D435i cameras, and a wrist-mounted RealSense D405. Depth is estimated from stereo observations using Fast-FoundationStereo (Wen et al., 2026), and combined with calibrated camera extrinsics to construct robot-base pointmaps. Fixed-camera extrinsics directly transform observations into the robot-base frame, while the wrist-camera transform combines the robot pose with hand-eye calibration. Figures 14 and 15 show the camera configuration and task examples.

For Peg in Hole, the robot manipulates shape-specific pegs, including arch-shaped, cylindrical, and hexagonal pegs, and inserts each peg into the correspondingly shaped hole. A representative task prompt is

```
Task: Insert the hexagonal pillar into the matching hole.
```

For Stack Blocks, the robot picks up a red cube and places it on top of the cube with the color specified by the instruction. During training, the target cube appears at one- or three-block heights, while evaluation includes unseen target heights. A representative task prompt is

```
Task: Place the red block on top of the orange tower.
```

For Pick-and-Place, the robot follows a spatially grounded instruction that specifies both the object to pick and the destination bowl, choosing among the left, middle, and right bowls. A representative task prompt is

```
Task: Pick up the carrot from the counter and place it in the middle
bowl.
```

We collect 60 demonstrations per task, yielding 180 demonstrations in total. RGB observations are recorded at $640 \times 480$ resolution, and robot trajectories and action targets are recorded at 20 Hz.

### D.2 Fine-Tuning and Deployment

We compare PointmapVLA with $\pi_{0.5}$ (Black et al., 2025a), FP3 (Yang et al., 2025), and Pose-VLA (Lin et al., 2026a). All models are fine-tuned on the same 180 demonstrations for 50K updates with a global batch size of 64.

Table 15 summarizes the observation inputs. PointmapVLA uses RGB images and pixel-aligned pointmaps in the robot-base frame. $\pi_{0.5}$ uses RGB images, Pose-VLA uses RGB, depth, and camera rays, and FP3 uses colored point clouds. Every method uses three views (left, right, and wrist) except for FP3. Following FP3 s original configuarion, we find that FP3 works best when only using the left and wrist views. We found that adding the right view degraded its performance in our experiments.

### D.3 Evaluation Protocol

For each policy, we evaluate 20 rollouts under the seen condition and 20 rollouts under the unseen condition for each task, resulting in 120 rollouts in total. All policies are evaluated using the same set of initial configurations and task instructions. Each rollout terminates upon success, failure, or

Table 15: **Real-world policy observations.** FP3 uses the left and wrist views, while the other policies additionally use the right view.

| Policy | Input | Views |
|---|---|---|
| $\pi_{0.5}$ | RGB | 3 |
| FP3 | Colored point clouds | 2 |
| Pose-VLA | RGB, depth, and rays | 3 |
| PointmapVLA | RGB and pointmaps | 3 |

reaching the time limit. Success rate is computed over the 20 rollouts for each task and condition. Table 16 summarizes the distribution shift and success criterion for each task.

Table 16: **Real-world evaluation protocol.** Each task is evaluated with 20 seen and 20 unseen rollouts per policy.

| Task | Seen | Unseen | Success criterion |
|---|---|---|---|
| Peg in Hole | Standard lighting | Rotating colored illumination with ceiling lights off | Insert and release the peg in the matching hole. |
| Stack Blocks | Target cubes at one- or three-block heights | Target cubes at unseen heights | Place the red cube on the instructed target cube. |
| Pick-and-Place | Training camera viewpoints | Novel external viewpoint replacing the left camera | Place the instructed object in the instructed bowl. |

### D.4 Detailed Results

Table 17 reports the task-level success rates shown in Figure 4. PointmapVLA achieves 77% average success under seen conditions and 48% under unseen conditions, compared with 52% and 27% for Pose-VLA, the strongest baseline on average. It attains the highest success rate in each task under both conditions.

Table 17: **Per-task real-world success rates (%).** S and U denote seen and task-specific unseen conditions. Each task–condition entry uses the latest scored outcomes for 20 cases. Average success rates are rounded to the nearest whole percent, following Figure 4.

| | Peg in hole | | Stack blocks | | Pick-and-place | | Average | |
|---|---|---|---|---|---|---|---|---|
| Policy | S | U | S | U | S | U | S | U |
| $\pi_{0.5}$ | 50 | 15 | 30 | 20 | 50 | 35 | 43 | 23 |
| FP3 | 55 | 15 | 20 | 10 | 15 | 10 | 30 | 12 |
| Pose-VLA | 50 | 30 | 40 | 15 | 65 | 35 | 52 | 27 |
| **PointmapVLA** | **75** | **45** | **75** | **50** | **80** | **50** | **77** | **48** |

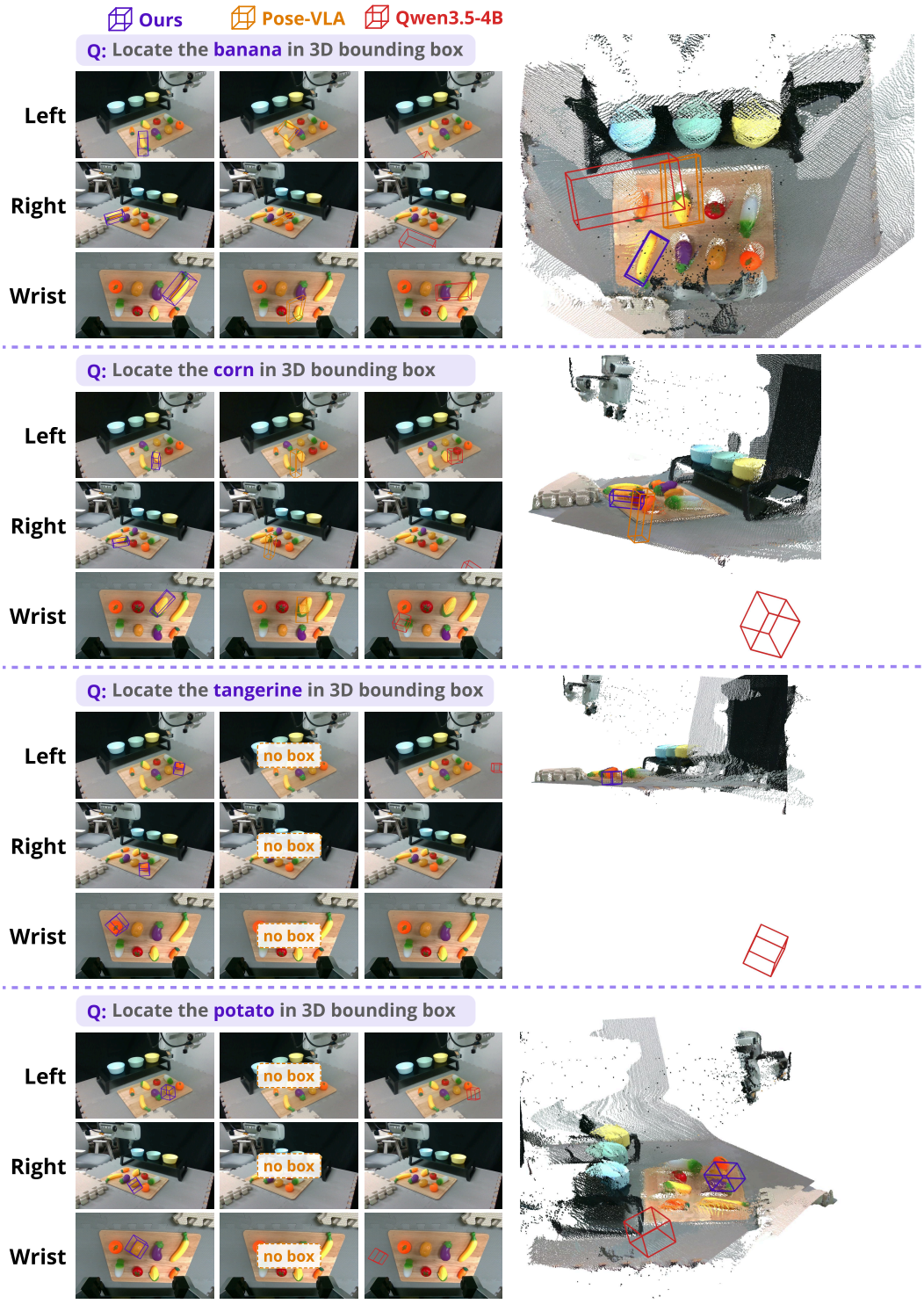



Figure 13: **Qualitative 3D grounding on a real-robot scene.** 3D bounding-box predictions for "banana", "corn", "tangerine" and "potato" projected onto the left, right, and wrist views.

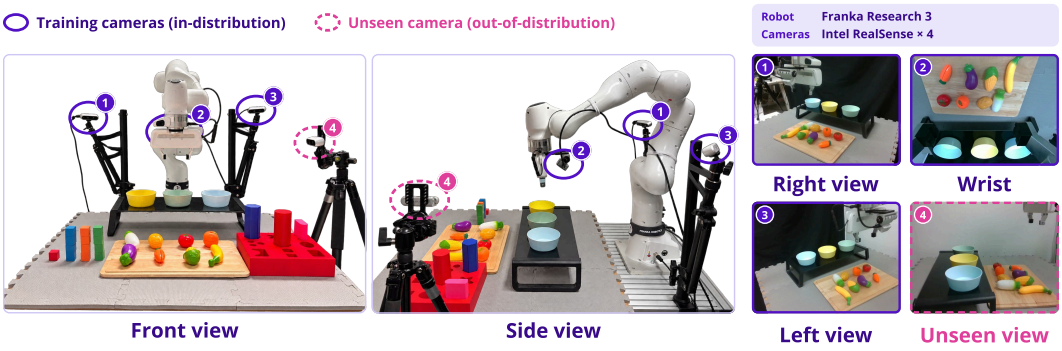



Figure 14: **Real-robot camera setup.** The workspace is observed by two fixed cameras and a wrist-mounted camera. Cameras 1–3 provide the right, wrist, and left views, respectively. Camera 4 replaces the left-camera input in the unseen Pick-and-place condition.

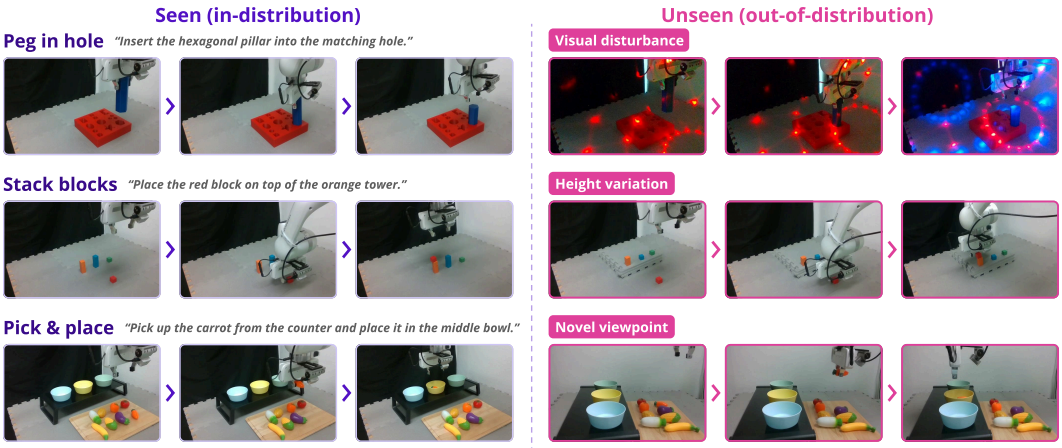



Figure 15: **Real-world tasks and evaluation conditions.** Representative sequences for Peg in hole, Stack blocks, and Pick-and-place with spatial instructions. Seen conditions are shown on the left and unseen conditions on the right, including visual disturbances, unseen target heights, and a novel camera viewpoint, respectively.