SPEAR-1: Scaling Beyond Robot Demonstrations via 3D Understanding

Nikolay Nikolov Giuliano Albanese Sombit Dey Aleksandar Yanev
Luc Van Gool Jan-Nico Zaech Danda Pani Paudel
{nikolay.nikolov, giuliano.albanese, sombit.dey, aleksandar.yanev
luc.vangool, jan-nico.zaech, danda.paudel}@insait.ai

INSAIT, Sofia University "St. Kliment Ohridski"

Abstract

Robotic Foundation Models (RFMs) hold great promise as generalist, end-to-end systems for robot control. Yet their ability to generalize across new environments, tasks, and embodiments remains limited. We argue that this stems from their foundations: most RFMs are built by fine-tuning internet-pretrained Vision-Language Models (VLMs). However, these VLMs are trained on 2D image-language tasks and lack the 3D spatial reasoning inherently required for embodied control in the 3D world. Bridging this gap is challenging due to the lack of diverse large-scale robotic data. Instead, we propose to enrich non-robotic image data with 3D annotations and enhance a pretrained VLM with 3D understanding capabilities. We build SPEAR-VLM: a 3D-aware VLM that infers object coordinates in 3D space from a single 2D image. Building on SPEAR-VLM, we introduce our main contribution, ***SPEAR-1****: a robotic foundation model that combines language-instructed embodied control with grounded 3D perception. We train SPEAR-1 on ∼45M frames from 24 Open X-Embodiment datasets and show it outperforms or matches state-of-the-art models such as π_0-FAST and $\pi_{0.5}$ while using 20× fewer robot demonstrations. This training strategy unlocks new VLM capabilities and as a consequence boosts the reliability of embodied control beyond what is achievable with robot-only data. We make our model weights and 3D-annotated datasets publicly available.*

1. Introduction

Vision-Language-Action (VLA) modeling has emerged as a promising paradigm for building generalist, end-to-end systems for robot control. Their success relies on two factors: (1) the strong visual-linguistic understanding inherited from internet-scale pretraining of the underlying VLM, which provides broad "common sense" knowledge and (2) training on large, diverse datasets of robot demonstrations.

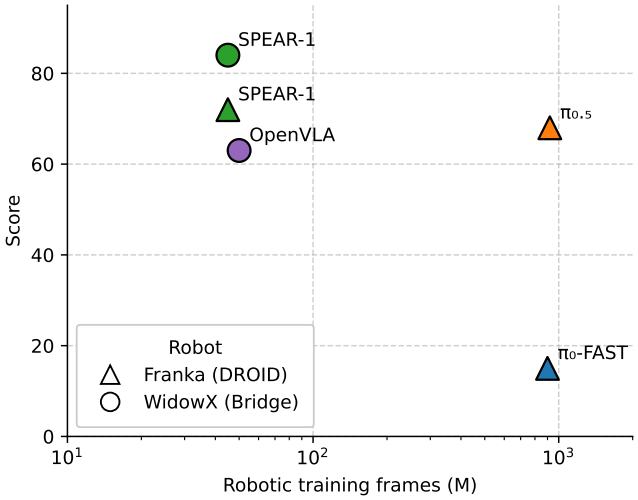

Figure 1. Robotic foundation models performance on different embodiments w.r.t amount of robotic training data. SPEAR-1 outperforms state-of-the-art π_0-FAST [31] and matches $\pi_{0.5}$ [5] on Franka embodiment while using 20× less robot demonstrations data.

However, building effective VLA models (VLAs) still comes with several challenges. First, most off-the-shelf VLMs are trained on 2D image-language tasks and thus lack the 3D spatial reasoning inherently required for embodied control in the 3D world. Second, acquiring and scaling robot demonstration data is costly and time-consuming, making it difficult to reach the data volumes needed for robust generalization [12].

Intuitively, 3D spatial reasoning capabilities can be acquired solely from visual data with 3D annotations, without any need to resort to expensive robot demonstration data. To address this, we integrate a pretrained depth encoder in an existing VLM and train the resulting model, called SPEAR-VLM, on 3D understanding tasks. In particular, in order to

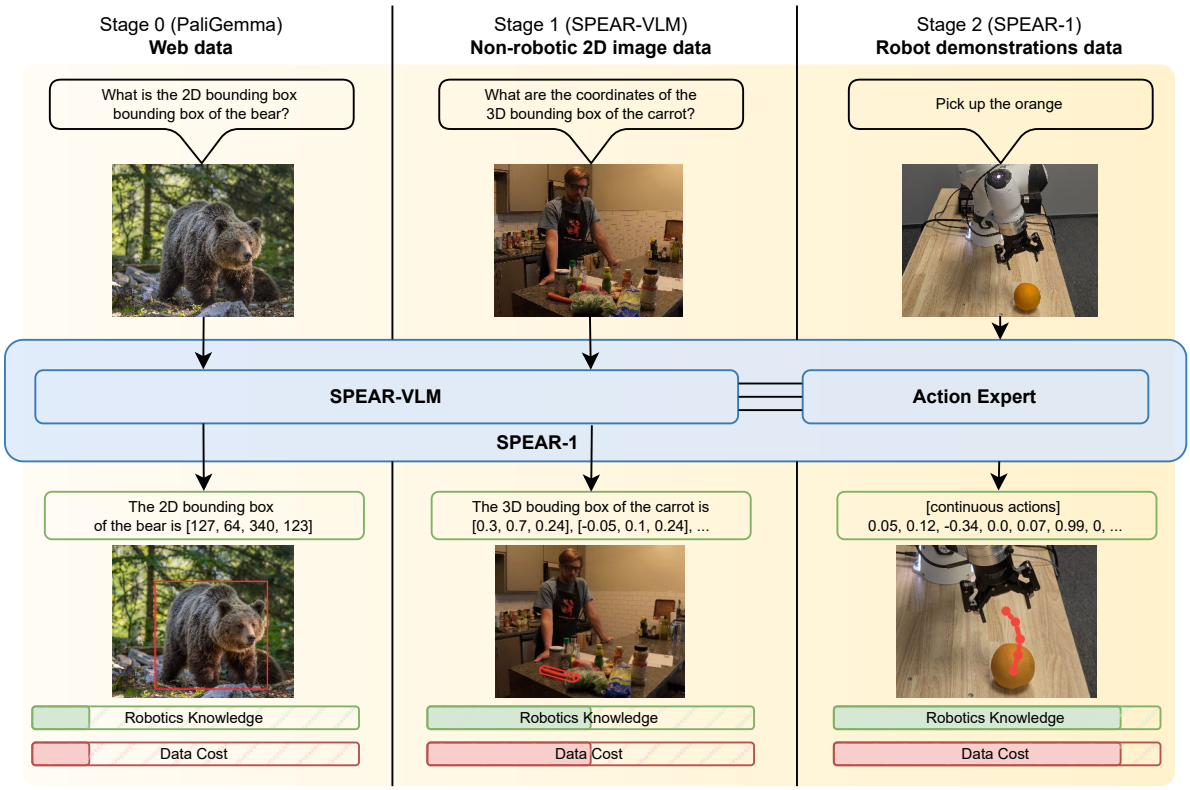

Figure 2. **SPEAR-1 stages of training. Stage 0**: General VLM pretraining on web scale data, *e.g*. PaliGemma. **Stage 1**: Integrate a mono depth vision encoder in PaliGemma VLM to build **SPEAR-VLM** and train it on embodied-inspired VQA tasks, *e.g*. 3D bounding box estimation or object-to-object distance estimation. We use 2D images from non-robotic data, enriched with 3D annotations. **Stage 2**: Add a randomly-initialized *action expert* to our 3D-enhanced SPEAR-VLM to train a generalist robotic foundational model, **SPEAR-1**, on robot demonstration data from OpenX [30]. Each stage boosts the model's robotics-relevant knowledge and capabilities, but simultaneously the abundance and diversity of data decreases.

embed as much control-relevant 3D knowledge in the VLM, we design the tasks in this pretraining stage to be as close as possible to the embodied tasks a VLA needs to learn. For example, SPEAR-VLM is trained to estimate the xyz components of 3D bounding boxes and distances between objects - tasks which intuitively an embodied VLA also needs to solve implicitly for accurate translation control.

Building on SPEAR-VLM, we introduce SPEAR-1, a robotics foundation model that combines language-instructed embodied control with grounded 3D perception. We find that by addressing the 3D understanding problem during the VLM training stage, we can actually reduce robot demonstration data requirements 20$\times$ and achieve superior performance than state-of-the-art robot foundation models such as π_0-FAST [31] and match $\pi_{0.5}$ [5].

Unlike previous works that try to address the challenge of 3D knowledge for robot control, SPEAR-1 demonstrates improvement on foundation level. It is capable of achieving state-of-the-art robot control on multiple robot embodiments solely by fine-tuning for the target embodiment rather than the specific target evaluation environment. Furthermore, SPEAR-1 demonstrates how significant amounts of robot demonstration data can be 'replaced' by non-robotic 3D-annotated image data.

In summary, our work makes the following contributions:

- **SPEAR-VLM**: a VLM with embodied-inspired 3D capabilities, *e.g*. localizing objects in 3D coordinates, trained on enriched 2D-image non-robotic datasets and boosting downstream VLA performance
- **SPEAR-1**: an open-weight robotics foundation model with 3D understanding, which achieves significant im-

provements over state-of-the-art baselines.

- Substantial reduction in reliance on hard-to-collect robotic data: by leveraging only **200k non-robotic 2D images**, SPEAR-1 outperforms state-of-the-art models trained with more than **900M additional frames of robotic demonstrations**

2. Related Work

Spatial Understanding for VLMs. Majority of existing VLMs trained on large-scale datasets have been limited to flat 2D image understanding [3, 17, 25, 37, 39, 42]. Our work builds on top of PaliGemma VLM [3] by integrating the MoGe monocular depth estimator [43] as a supplementary vision backbone and training on manipulation-relevant 3D tasks to enhance the VLM understanding to 3D. Previously, Chen et al. [7] used a similar data annotation approach for training a 3D-aware VLM, but they do not integrate a pretrained depth estimator in the VLM and neither the model, nor the dataset is publicly accessible. Additionally, unlike SpatialVLM [7] or RoboSpatial [36], trained on high-level spatial relationships, our SPEAR-VLM focuses on explicit 3D-coordinate prediction: a pretraining task intuitively much closer to embodied control. SpatialBot [6] also previously proposed a spatially-aware VLM targeting robot control, but their method involves multi-step VLM inference process and was never shown to integrate in a VLA for generalist robotic control.

Vision-Language-Action Models. Recently, multiple works have developed generalist robot policies [4, 5, 9, 19, 30, 31, 47] trained on multiple robot embodiments. Our SPEAR-1 builds on top of the π_0 architecture, but we initialize the underlying VLM from our SPEAR-VLM to integrate pretrained 3D understanding. Previously, SpatialVLA [33] proposed integrating a monocular depth encoder [44] in the VLA, but without any VLM alignment or pretraining and therefore learning 3D capabilities entirely from hard-to-collect robotic data. MolmoAct [20] recently proposed a spatially-aware VLA, but the approach involves 'reasoning' at inference time, rendering the method impractical for real-time control due to high latency. Most closely related, Gemini Robotics 1.0 [40] follows a similar 3D pretraining method to fine-tune the significantly larger Gemini 2.0 [32] and distill into a smaller VLA with reasoning capabilities. With the majority of the method details remaining undisclosed, our work still differs in multiple important aspects: (1) we investigate the benefits of 3D pretraining in isolation, (2) train on substantially smaller and less diverse, but open-access datasets from OpenX [30], (3) train a VLA capable of running inference locally on the robot instead of in the cloud and (4) demonstrate the ability to reduce robotic data requirements with non-robotic 2D images.

3. Method

In this section, we describe SPEAR-1 and its training recipe in detail. In section 3.1 we describe the architecture, data generation pipeline, and training procedure of our 3D-aware SPEAR-VLM. This stage aims to enhance the 3D spatial understanding capabilities of an off-the-shelf VLM through fine-tuning on 3D spatial perception tasks. We then proceed, in section 3.2 to detail the architecture and training procedure of SPEAR-1, which comprises a pre-training and post-training stage. The pre-training stage involves training on a large and diverse mixture of robot demonstration data to acquire general knowledge of robot manipulation. Post-training involves fine-tuning for a specific embodiment.

3.1. SPEAR-VLM

Most recent robotics foundational models are based on Vision-Language-Models (VLMs) pretrained on large corpora of internet-scale text-image data. The architecture of those models usually consists of a vision encoder, a vision-to-text-embedding feature projector, and a LLM. The majority of the tasks on which VLMs are usually trained are limited to 2D space [3, 17, 24], e.g. image captioning, 2D bounding box detection, object segmentation, OCR, visual question answering (VQA). To extend the capabilities of a pretrained VLM to 3D understanding, we propose (1) extending the model architecture by adding a monocular depth encoder and (2) training the VLM on VQA tasks that require explicit 3D reasoning.

VLM Architecture. Our model uses PaliGemma [3] as backbone, but the same method can be used with any late-fusion VLM [1, 10, 26]. PaliGemma consists of three main components: (1) a SigLIP **visual encoder** [45], (2) a linear **projector** that maps the visual tokens predicted by the visual encoder to the language model input space and (3) a Gemma **language model** [38]. To enable the model to perceive depth more accurately, we integrate the MoGe [43] depth encoder as an additional vision encoder. We choose MoGe due to its affine-invariant modeling approach, capable of fitting cameras with different intrinsics. Our intuition is that affine-invariant depth should generalize better across environments thus being better suited for learning feedback control. Similar to MoGe decoder inputs, we concatenate the intermediate features from the last 4 layers of the MoGe ViT encoder along the feature dimension, project them to the LLM embedding space via randomly-initialized linear projector. The visual input to the LLM consists of the averaged outputs of SigLIP and MoGe projectors. To encode 3D information into text we extend the PaliGemma tokenizer with $N = 1024$ 3D tokens (see Appendix A.2.3).

3D pretraining tasks. Given the above architecture, we propose a pre-training scheme to enable the model to leverage the depth information in MoGe's encoder features and acquire 3D spatial understanding capabilities. To embed as

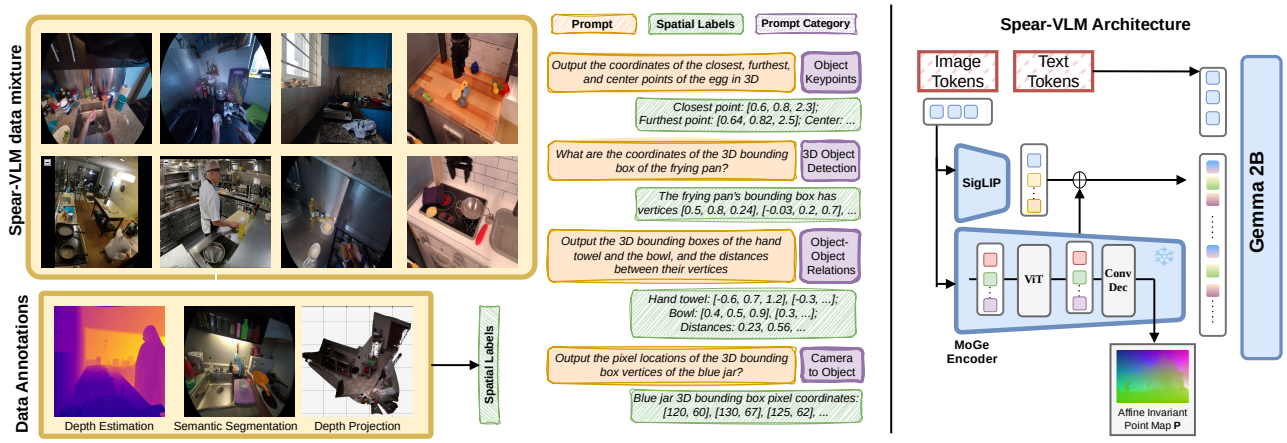

Figure 3. **SPEAR-VLM overview.** Left: Training data mixture, annotations and example question-answer pairs. Right: High-level architecture and fusion between SigLIP and MoGe encoders.

much control-relevant 3D knowledge in the VLM, we design VQA tasks inspired by some of the embodied tasks a VLA needs to learn, *e.g. Output the vertices of the 3D bounding box of object X* or *Output the xyz components of the distance between object X and object Y*. Similar to some VLA tasks, *e.g. Place object X on object Y*, our VLM pre-training tasks require learning semantic 3D localization, object-to-object spatial relations, and 3D coordinate system geometry. For a full list of question-answer pairs, see Appendix A.2.1.

3D Vision-Question-Answering Data. There are few open datasets that contain the annotations needed for the proposed training scheme. We devise the following semi-automatic annotation pipeline to enrich existing datasets with the necessary annotations: *object-level segmentation masks, semantic labels and projected 3D point cloud*. Importantly, our pipeline requires only 2D images as input, enabling the use of large-scale image datasets. We utilize off-the-shelf vision foundation models as follows:

1. Use Gemini [8] to detect 2D bounding boxes and semantic labels for the objects on the image.
2. Prompt SAM2 [34, 35] with the detected bounding boxes to produce instance-level segmentation masks
3. Obtain 3D point cloud annotations for the entire image via MoGe [43]

To construct a training example, we randomly sample a templated prompt and an object (or several) from the image, obtain the object 3D point cloud by filtering the annoted MoGe 3D point cloud with the object segmentation mask, and compute the oriented 3D bounding box as well as any other 3D information needed to construct the question-answer pair.

We focus on indoor environments and annotate the "cooking" and "bike repair" parts of EgoExo4D [13] which already have segmentation masks, resulting in 200k images. For visual diversity, we also annotate 30k frames of the Bridge-V2 [41] robot demonstration dataset, downsampled to 10% in the VLM training data mixture.

Training process. Similar to LLaVa [24], we train the VLM in two stages. In the first stage, we initialize from PaliGemma and MoGe weights, with the MoGe projector and the LLM depth token embeddings initialized randomly. We train only the randomly initialized weights and SigLIP projector, keeping everything else frozen. In the second and longer stage, we keep only SigLIP and MoGe encoders frozen and we scale the next-token-prediction loss for depth tokens by a factor $\lambda = 2$.

3.2. SPEAR-1

SPEAR-1 builds upon SPEAR-VLM by extending it with an action expert module to predict continuous actions via conditional flow matching. We provide a detailed overview of the architecture and formulation in the following.

Preliminaries. Formally, we want to learn a function $\pi(\cdot)$ mapping an observation $\mathbf{o}_t$ to a sequence of robot actions $\mathbf{A_t} = [\mathbf{a}_t, \mathbf{a}_{t+1}, \dots \mathbf{a}_{t+H-1}]$ over an horizon H. The observation is defined as $\mathbf{o}_t = [\mathbf{I}_t^1, \dots, \mathbf{I}_t^n, \mathbf{p}_t, \mathbf{l}_t]$, where $\mathbf{I}_t^i$ is the i-th image observation from an uncalibrated camera , $\mathbf{p}_t$ is a vector containing the robot state comprising of the end-effector pose and gripper state, $\mathbf{l}_t$ is a vector of language tokens representing the language instruction. In this work we focus on learning position control of fixed-base single-arm manipulators. Each action in the sequence is thus composed of an end-effector position control and a gripper control. The end-effector control is defined as a delta with respect to the current end-effector cartesian pose $\Delta_{ee} = [\Delta_{trans}, \Delta_{rot}]$. The translation component, Δ_{trans} is in base frame and the rotation component, Δ_{rot}, is in end-effector frame and is represented as a quaternion. The

gripper action is binary.

Architecture. Similar to π_0 [4], SPEAR-1 combines a VLM, which processes the image-language inputs, with an *action expert* module, which processes robot proprioception observations and predicts the robot action sequence conditioned on the VLM transformer intermediate key-value pairs. The action expert has the same architecture and number of layers as the Gemma [38] transformer, but its hidden size is twice smaller for a total of ∼300M parameters. Corresponding layers in the VLM and the action expert have a shared attention operation with block-wise causal attention over the blocks $[\mathbf{I}_t, \mathbf{l}_t], [\mathbf{p}_t], [\hat{\mathbf{a}}_{t+1}, \dots, \hat{\mathbf{a}}_{t+H-1}]$. Within each block, there is full bidirectional attention and the tokens in each block can attend to tokens in previous blocks, but cannot attend to the tokens in future blocks. During training, only the action sequence prediction is supervised and the gradient updates are propagated back to the VLM parameters through the shared attention layers. For further details, see Appendix A.3.1.

Flow Matching Formulation. The action sequence prediction is supervised via conditional flow matching [22, 23, 27]. Specifically, the model takes as input the observation $\mathbf{o}_t$, the flow-matching step $\tau \in [0, 1]$ and a sequence of noisy actions $\mathbf{A}_t^\tau = [\mathbf{a}_t^\tau, \dots, \mathbf{a}_{t+H-1}^\tau]$ and outputs a denoising vector $\mathbf{v}_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t)$. We denote the decomposed action of translation, rotation and gripper components as $\mathbf{a}_t = [\mathbf{x}_t, \mathbf{q}_t, \mathbf{g}_t]$. For clarity, we use the square brackets operator $[\cdot]$ on the predicted denoising vector $\mathbf{v}_\theta$ and the denoising vector field $\mathbf{u}$ to denote a specific component, *e.g.* $\mathbf{u}[\mathbf{x_t}]$ corresponding to the translation component of the denoising vector field.

We follow a flow matching formulation in linear space for translation and on the $\mathbb{S}^3$ manifold of unit quaternions for rotation. For simplicity, we omit the gripper component as it follows the same linear formulation as translation.

In practice, during training we sample a random timestep $\tau \sim \mathcal{B}(\alpha, \beta)$ and random noise $\mathbf{x}_\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \mathbf{q}_\epsilon \sim \mathcal{U}(\mathbb{S}^3)$. We then compute the "noisy actions" by linear interpolation for translation $\mathbf{x}_t^\tau = \tau \mathbf{x}_t + (1 - \tau)\mathbf{x}_\epsilon$ and spherical linear interpolation on the $\mathbb{S}^3$ manifold for quaternion rotation

$$\mathbf{q}_t^\tau = \frac{\sin\big((1-\tau)\theta\big)}{\sin\theta}\,\mathbf{q}_\epsilon \;+\; \frac{\sin(\tau\theta)}{\sin\theta}\,\mathbf{q}_t, \tag{1}$$

with $\theta = \cos^{-1}(\mathbf{q}_\epsilon \cdot \mathbf{q}_t)$.

We then pass the "noisy action sequence" $\mathbf{A}_t^\tau$ as input to the model and train it to output the denoising vector field $\mathbf{u}(\mathbf{A}_t^\tau | \mathbf{A}_t) = \dfrac{d\mathbf{A}_t^\tau}{d\tau}$.

We apply a conditional flow-matching loss. For translation, this is equivalent to the MSE loss

$$\mathcal{L}_{\mathbb{R}^3}(\theta) = \Big|\Big|\mathbf{v}_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t)[\mathbf{X}_t] - \mathbf{u}(\mathbf{A}_t^\tau | \mathbf{A}_t)[\mathbf{X}_t]\Big|\Big|^2. \tag{2}$$

We use a combination of two losses for rotations, as we experimentally found a single loss to be insufficient on its own. We apply a cosine loss directly on the velocity predictions $\mathcal{L}_t^{\cos}(\theta) = 1 - \mathbf{v}_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t)[\mathbf{q}] \cdot \mathbf{u}(\mathbf{A}_t^\tau | \mathbf{A}_t)[\mathbf{q}]$ and a geodesic loss $\mathcal{L}_t^{\text{geo}}(\theta) = \min|\mathbf{q}_t^{\tau+\delta} \pm \mathbf{q}_{\theta,t}^{\tau+\delta}|$ [11, 14] on a rotation prediction $\mathbf{q}_{\theta,t}^{\tau+\delta} = \mathbf{q}_t^\tau \otimes \mathbf{q}_{\theta,t}^\delta \in \mathbb{S}^3$, where $\mathbf{q}_{\theta,t}^\delta$ is computed by integrating $\mathbf{v}_\theta[\mathbf{q}_t] \in \mathbb{R}^4$ over a randomly sampled integration step $\delta \in (0.01, 1-\tau)$, and $\mathbf{q}_t^{\tau+\delta}$ is computed from (1). The complete rotation loss is thus $\mathcal{L}_{\mathbb{S}^3}(\theta) = \sum_{k=t}^{t+H} \left[\mathcal{L}_k^{\cos}(\theta) + \mathcal{L}_k^{\text{geo}}(\theta)\right]$ and the final translation and rotation loss is

$$\mathcal{L}(\theta) = \mathbb{E}_{p(\mathbf{A}_t|\mathbf{o}_t), q(\mathbf{A}_t^\tau|\mathbf{A}_t)}\left[\mathcal{L}_{\mathbb{R}^3}(\theta) + \mathcal{L}_{\mathbb{S}^3}(\theta)\right]. \tag{3}$$

During inference, we generate actions by integrating the learned vector field from $\tau = 0$ to $\tau = 1$, starting with random noise $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \mathbf{q}_0 \sim \mathcal{U}(\mathbb{S}^3)$ and using Euler integration in linear space for translations

$$\mathbf{x}_t^{\tau+\delta} = \mathbf{x}_t^\tau + \delta \mathbf{v}_\theta^{\mathbf{x}}(\mathbf{A}_t^\tau, \mathbf{o}_t), \tag{4}$$

and on the $\mathbb{S}^3$ manifold for rotations

$$\mathbf{q}_t^{\tau+\delta} = \mathbf{q}_t^\tau \otimes \mathbf{q}_t^\delta(\mathbf{v}_\theta^{\mathbf{q}}(\mathbf{A}_t^\tau, \mathbf{o}_t)). \tag{5}$$

For further details, see Appendix A.3.2.

4. Experimental evaluation

We evaluate the performance of SPEAR-1 as a generalist policy for robot manipulation and compare it to open-weights and open-source state-of-the-art VLA models. Concretely, our experiments aim to answer the following research questions:

1. Does 3D VLM pretraining improve the performance of SPEAR-1 on robot control tasks?
2. How well does SPEAR-1 compare against state-of-the-art VLA models?

To answer these questions, we evaluate SPEAR-1 on a variety of manipulation tasks in both simulation and multiple real-world environments.

4.1. Implementation details

VLM training. We train SPEAR-VLM with a batch size of 512 for 2k steps during the first alignment stage and 10k steps for the second stage, for a total of 18 hours on 16 Nvidia H200 GPUs.

VLA pre-training. For VLA training, we start from SPEAR-VLM and randomly initialized action expert weights. We provide two camera views as inputs to the model: external, with resolution 280x210, and wrist, with resolution 112x112. When the wrist camera view is not available, we feed a black image. We train on 32 H200

Method	Carrot on Plate (Dist)	Carrot on Plate (Elev.)	Marker in Cup (Dist)	Average
π_0-PaliGemma (DROID)	0.0	0.32	67.0	0.34
π_0-SPEAR-VLM (DROID)	0.42	0.52	43.0	0.46

Table 1. **Comparison of a VLA based on PaliGemma [3] and a VLA based on SPEAR-VLM on the DROID dataset [18].** Scores indicate task progress (higher is better). SPEAR-VLM achieves noticeable improvement on average. Results on "Carrot on Plate" task, which is unseen during DROID training, indicate that SPEAR-VLM leads to better generalization to 3D positions of the target objects.

Model	Put Spoon on Towel	Put Carrot on Plate	Stack Green Block on Yellow Block	Put Eggplant in Yellow Basket	Overall
OpenVLA	0%	0%	0%	4.1%	1.0%
SpatialVLA	16.7%	25.0%	29.2%	100.0%	42.7%
SPEAR-1 (ours)	62.5%	58.3%	45.82%	62.5%	**57.3%**

Table 2. **SIMPLER [21] simulation evaluations.** SpatialVLA numbers copied from [33]. SIMPLER results tend to be indicative of A/B performance in the real world, but not necessarily of absolute performance.

GPUs with batch size 2048 for 300k steps (∼6 days) on a data mixture comprising 24 datasets (see Table 3) from the Open X-Embodiment (OXE) collection [30].

VLA post-training. For WidowX real-world and SIMPLER simulation experiments and for Franka real-world experiments, we additionally fine-tune our OXE pre-trained SPEAR-1 for 50k steps on the Bridge V2 [41] and DROID [18] datasets respectively. We refer to these versions as **SPEAR-1 (Bridge)** and **SPEAR-1 (DROID)** respectively.

4.2. Ablation study: SPEAR-VLM vs PaliGemma

We first evaluate whether 3D VLM pretraining improves VLA performance on downstream robot control tasks. We instantiate two VLA models from the same π_0 style architecture: one initialized from the base PaliGemma VLM and the other from our 3D-aware SPEAR-VLM. Due to the cost of pre-training on the entire OXE mixture (Table 3), we train both models directly on DROID for 100k steps and batch size 2048. We refer to the resulting models as π_0-PaliGemma (DROID) and π_0-SPEAR-VLM (DROID) respectively. We then compare the performance of both VLAs on three of the four tasks from the Franka experiments. The results are reported in Table 1. We can observe that π_0-SPEAR-VLM (DROID) is able to outperform the baseline by more than 10% on average. We note that the task "Carrot on plate" is not seen in the DROID training dataset, showing the improved generalization capabilities of SPEAR-VLM. We hypothesize that the lower scores of both models on the variation of the task on the tabletop vs the variation with different elevations is due to workspace 3D position being out-of-distribution compared to the training data. Even in this case, π_0-SPEAR-VLM (DROID) is able to successfully complete the task in some cases while π_0-PaliGemma (DROID) fails every time.

4.3. Simulation experiments

We evaluate SPEAR-1 on the tasks of the WidowX Robot environment of the SimplerEnv simulation benchmark [21], and compare it with OpenVLA [19] and SpatialVLA [33]. For OpenVLA, we use the publicly available weights trained on OXE, whereas for SpatialVLA we use the publicly available weights pre-trained on OXE and fine-tuned on BridgeV2.

We report the results in Table 2. Our model is able to outperform the baselines by more than 10%.

In our experience, we found SIMPLER simulation results to be indicative of A/B performance of the models on the real WidowX robot, but not necessarily of absolute performance. Therefore, we focus on real-world evaluations.

4.4. Real-world experiments

We conduct evaluations on a total of 8 manipulation tasks across two robot platforms: WidowX and Franka Research 3 (Franka). The tasks are designed to assess the ability of the evaluated models to generalize to unseen environments and objects. Following [2], we design the tasks to be quite difficult for the evaluated models, targeting policy success rates around 50%. This is often achieved through the addition of distractors and by varying the visual background.

Evaluation protocol. For each task we define M initial conditions by varying the starting position of the objects in the scene. Following previous works [2, 30], we define a scoring rubric with partial scoring for each task. For more details about the scoring rubrics, see Appendix A.4. We execute N trials for each initial condition, for a total of NxM trials per task. We report the average score for each task and across tasks.

WidowX experiments. Our hardware setup for this set of experiments closely matches the original Bridge V2 setup [41], with a single Logitech C920 external camera po-

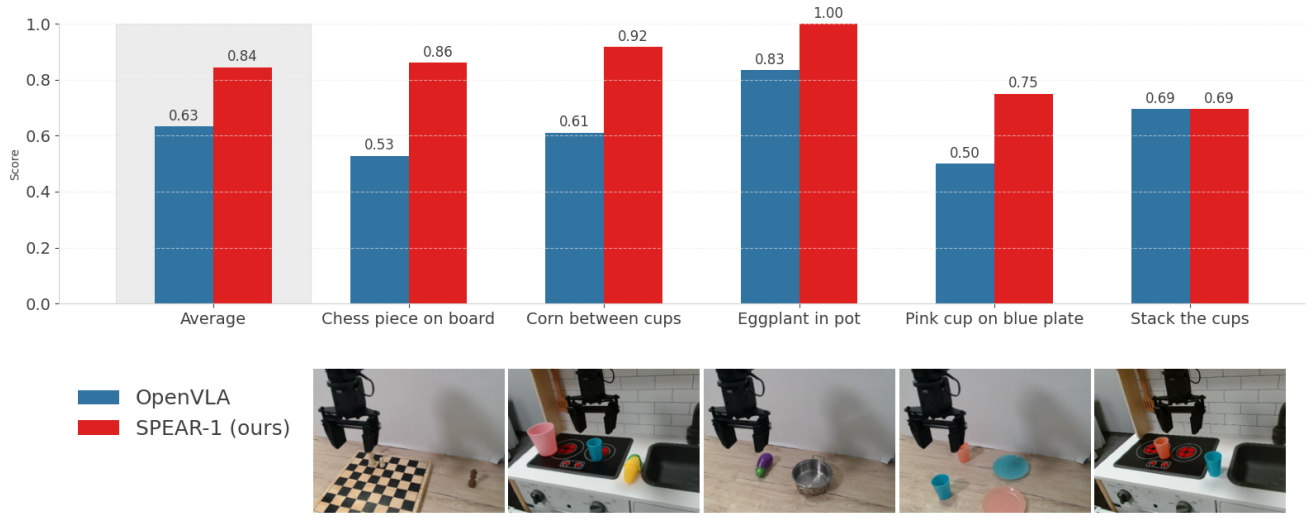

Figure 4. **Real world evaluation on WidowX.** SPEAR-1 is able to achieve 20% higher average task progress across all tasks than OpenVLA, a strong baseline in this setting.

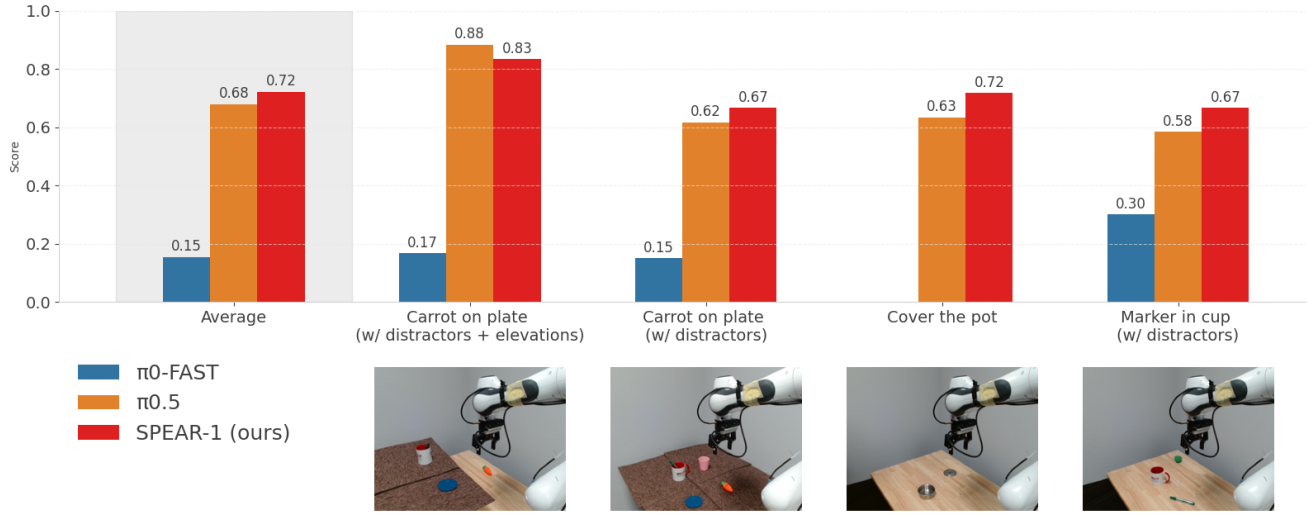

Figure 5. **Real world evaluation on Franka.** Scores indicate task progress (higher is better). We find that without any fine-tuning on the target environment, SPEAR-1 noticeably outperforms π_0-FAST, and matches $\pi_{0.5}$, even though both baselines are trained on 20$\times$ more robotic data from significantly more diverse environments.

sitioned on either side of the robot arm and pointing toward the workspace. For this set of experiments, 4 tasks are evaluated, with M = 4, N = 3. We compare the performance of SPEAR-1 with OpenVLA [19], using the publicly released implementation and model weights. We tried comparing to SpatialVLA [33], but were unable to get the model to work successfully in our setup. In this setting, we do not compare against π_0 [4], π_0-FAST [31] and $\pi_{0.5}$ [5] due to the unavailability of publicly accessible weights for the WidowX platform. The results are reported in Figure 4. SPEAR-1 is able to achieve 20% higher average task progress across all tasks than OpenVLA, a notoriously strong baseline in this setting.

Franka experiments. Our hardware setup for this set of experiments is similar to that of DROID [18]. We design 4 tasks, with M = 5 and N = 3. Tasks 1 and 2, "Carrot on plate (distractors)" and "Carrot on plate (distractors + elevations)", both feature a seen task whose difficulty is increased by the presence of distractors and by varying the elevation of the workspace. Task 3 and 4, "Marker in cup (with distractors)" and "Cover the pot", require the model to be able to precisely grasp small objects and reason about

the correct way to accurately position them to complete the task. We found that the inclusion of the wrist camera view is crucial for training on DROID and deployment on a similar setup. Therefore, to ensure a fair comparison, we only compare against open-weights models for which a wrist camera enabled version finetuned on DROID is publicly available. Specifically, we compare SPEAR-1 with the DROID-finetuned variants of π_0-FAST [4, 31], a strong autoregressive baseline, and $\pi_{0.5}$ [5], one of the latest state-of-the-art robotic foundation model optimized for open-world generalization.

The results of this set of experiments are reported in Figure 5. Without any fine-tuning on the target environment, SPEAR-1 is able to significantly outperform π_0-FAST, and match $\pi_{0.5}$. We note that both baselines do not integrate any sort of specialized 3D-aware training and are trained on at least 900M more robot demonstration frames collected in more diverse environments. In contrast, SPEAR-1 is trained on ∼45M frames, approximately $20\times$ less robotics data. These results indicate the importance of 3D-based knowledge and pretraining for generalization in robot manipulation tasks.

π_0-FAST integrates a specialized action tokenization compared to π_0 and was the first generalist policy trained on the DROID dataset [18] to be successfully evaluated zero-shot in unseen environments, without fine-tuning. In comparison, SPEAR-1, which follows the π_0 architecture, can reach $\sim 4\times$ higher performance than π_0-FAST on our set of tasks without fine-tuning and without the large-scale robotic data used by π_0-FAST.

Apart from architectural enhancements and co-training on top of π_0-FAST, $\pi_{0.5}$ integrates a high-level semantic subtask prediction and robotic data mixture explicitly focused on environment diversity. Qualitatively and quantitatively we find $\pi_{0.5}$ to be much better at environment generalization than π_0-FAST and match SPEAR-1's performance on our set of evaluation tasks. This suggests that 3D VLM pretraining on non-robotic data from diverse environments might be a more scalable way to boost robotic models' generalization capabilities without the need for large-scale robotic data collection in diverse environments.

5. Discussion and Limitations

We introduced **SPEAR-VLM**, a 3D-aware vision-language model derived from PaliGemma and trained on 2D images from non-robotic datasets enriched with 3D annotations. To embed as much control-relevant 3D knowledge in SPEAR-VLM, we train it on VQA tasks inspired by embodied tasks such as 3D bounding box prediction and object-to-object distance prediction. Stepping on this foundation, we built **SPEAR-1**, a robotics foundation model that can be deployed across multiple robot platforms and embodiments, exhibits robustness to 3D variations, and matches or outperforms state-of-the-art foundation models which have been trained on $20\times$ more robot demonstrations data. Ablation studies support our hypothesis that enhancing VLM capabilities with 3D knowledge is the primary factor driving robustness and *reducing dependence on hard-to-collect robot demonstrations data*.

Ablation studies support our hypothesis that enhancing VLM capabilities with 3D knowledge is the primary factor driving robustness and *reducing dependence on hard-to-collect robot demonstrations data*.

While SPEAR-1 is a step forward in building generalist robot foundation models, there are multiple questions yet to be studied. For instance, maybe other embodied capabilities could possibly be enhanced with suitable 3D VLM pretraining tasks. Large-scale 3D VLM pretraining on web-scale data could also further boost downstream robotic performance across multiple environments and visual backgrounds. It also remains to be seen how well SPEAR-1 generalizes to orders of magnitude more tasks and environments against models such as $\pi_{0.5}$ trained on significantly more diverse robot data.

Our 3D approach also bears a number of limitations. For instance, it is not well suited for deformable objects or objects with complex shapes. Perhaps different 3D priors could be used to better capture the geometry of such objects. It is also not immediately clear how to best combine MoGe's affine-invariant depth predictions with existing point cloud datasets in metric space and maybe metric-depth estimators can help resolve this limitation.

Acknowledgments

Project Lead: Nikolay Nikolov, Project Manager: Jan-Nico Zaech, PI: Danda Pani Paudel, Luc Van Gool

We thank Alexander-Marc Spiridonov and Anna-Maria Halacheva for feedback and helpful technical discussions. We also thank Hristo Venev for engineering support and Kamen Pavlov for help with figures and visuals.

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andy Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*, 2022. 3

[2] Jose Barreiros, Andrew Beaulieu, Aditya Bhat, Rick Cory, Eric Cousineau, Hongkai Dai, Ching-Hsin Fang, Kunimatsu Hashimoto, Muhammad Zubair Irshad, Masha Itkina, et al. A careful examination of large behavior models for multitask

dexterous manipulation. *arXiv preprint arXiv:2507.05331*, 2025. 6

[3] Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, Thomas Unterthiner, Daniel Keysers, Skanda Koppula, Fangyu Liu, Adam Grycner, Alexey Gritsenko, Neil Houlsby, Manoj Kumar, Keran Rong, Julian Eisenschlos, Rishabh Kabra, Matthias Bauer, Matko Bošnjak, Xi Chen, Matthias Minderer, Paul Voigtlaender, Ioana Bica, Ivana Balazevic, Joan Puigcerver, Pinelopi Papalampidi, Olivier Henaff, Xi Xiong, Radu Soricut, Jeremiah Harmsen, and Xiaohua Zhai. PaliGemma: A versatile 3B VLM for transfer. *arXiv preprint arXiv:2407.07726*, 2024. 3, 6

[4] 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. *arXiv preprint arXiv:2410.24164*, 2024. 3, 5, 7, 8

[5] 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 *9th Annual Conference on Robot Learning*, 2025. 1, 2, 3, 7, 8

[6] Wenxiao Cai, Iaroslav Ponomarenko, Jianhao Yuan, Xiaoqi Li, Wankou Yang, Hao Dong, and Bo Zhao. Spatialbot: Precise spatial understanding with vision language models. *arXiv preprint arXiv:2406.13642*, 2024. 3

[7] Boyuan Chen, Zhuo Xu, Sean Kirmani, Brain Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14455–14465, 2024. 3

[8] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025. 4

[9] Sombit Dey, Jan-Nico Zaech, Nikolay Nikolov, Luc Van Gool, and Danda Pani Paudel. Revla: Reverting visual domain limitation of robotic foundation models. *arXiv preprint arXiv:2409.15250*, 2024. 3

[10] Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023. 3

[11] Andreas René Geist, Jonas Frey, Mikel Zhobro, Anna Levina, and Georg Martius. Learning with 3d rotations, a hitchhiker's guide to SO(3). In *Forty-first International Conference on Machine Learning*, 2024. 5

[12] Ken Goldberg. Good old-fashioned engineering can close the 100,000-year "data gap" in robotics. *Science Robotics*, 10(105):eaea7390, 2025. 1

[13] Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, et al. Ego-exo4d: Understanding skilled human activity from first-and third-person perspectives. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19383–19400, 2024. 4, 13

[14] Richard Hartley, Jochen Trumpf, Yuchao Dai, and Hongdong Li. Rotation averaging. *International journal of computer vision*, 103(3):267–305, 2013. 5

[15] John Hewitt. Initializing new word embeddings for pretrained language models. https://www.cs.columbia.edu/~johnhew/vocab-expansion.html. 13

[16] HuggingFace. Huggingface transformers documentation. https://huggingface.co/docs/transformers/en/main_classes/model. 13

[17] Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. Prismatic vlms: Investigating the design space of visually-conditioned language models. *arXiv preprint arXiv:2402.07865*, 2024. 3

[18] 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. In *Robotics: Science and Systems*, 2024. 6, 7, 8

[19] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024. 3, 6, 7

[20] Jason Lee, Jiafei Duan, Haoquan Fang, Yuquan Deng, Shuo Liu, Boyang Li, Bohan Fang, Jieyu Zhang, Yi Ru Wang, Sangho Lee, et al. Molmoact: Action reasoning models that can reason in space. *arXiv preprint arXiv:2508.07917*, 2025. 3

[21] Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, Sergey Levine, Jiajun Wu, Chelsea Finn, Hao Su, Quan Vuong, and Ted Xiao. Evaluating real-world robot manipulation policies in simulation. *arXiv preprint arXiv:2405.05941*, 2024. 6

[22] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. 5

[23] Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky TQ Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code. *arXiv preprint arXiv:2412.06264*, 2024. 5

[24] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Advances in Neural Information Processing Systems*, pages 34892–34916. Curran Associates, Inc., 2023. 3, 4

[25] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024. 3

[26] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024. 3

[27] Qiang Liu. Rectified flow: A marginal preserving approach to optimal transport. *arXiv preprint arXiv:2209.14577*, 2022. 5

[28] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023. 13

[29] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 12

[30] Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023. 2, 3, 6

[31] 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. *arXiv preprint arXiv:2501.09747*, 2025. 1, 2, 3, 7, 8

[32] Sundar Pichai, Demis Hassabis, and Koray Kavukcuoglu. Introducing gemini 2.0: our new ai model for the agentic era. https://blog.google/technology/google-deepmind/google-gemini-ai-update-december-2024/. 3

[33] 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. 3, 6, 7, 12

[34] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. 4, 13

[35] Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, Zhaoyang Zeng, Hao Zhang, Feng Li, Jie Yang, Hongyang Li, Qing Jiang, and Lei Zhang. Grounded sam: Assembling open-world models for diverse visual tasks, 2024. 4

[36] Chan Hee Song, Valts Blukis, Jonathan Tremblay, Stephen Tyree, Yu Su, and Stan Birchfield. Robospatial: Teaching spatial understanding to 2d and 3d vision-language models for robotics. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 15768–15780, 2025. 3

[37] Andreas Steiner, André Susano Pinto, Michael Tschannen, Daniel Keysers, Xiao Wang, Yonatan Bitton, Alexey Gritsenko, Matthias Minderer, Anthony Sherbondy, Shangbang Long, et al. Paligemma 2: A family of versatile vlms for transfer. *arXiv preprint arXiv:2412.03555*, 2024. 3

[38] Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024. 3, 5

[39] Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025. 3

[40] Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, et al. Gemini robotics: Bringing ai into the physical world. *arXiv preprint arXiv:2503.20020*, 2025. 3

[41] Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, Vivek Myers, Kuan Fang, Chelsea Finn, and Sergey Levine. Bridgedata v2: A dataset for robot learning at scale. In *Proceedings of the Conference on Robot Learning (CoRL)*, 2023. 4, 6, 13

[42] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024. 3

[43] Ruicheng Wang, Sicheng Xu, Cassie Dai, Jianfeng Xiang, Yu Deng, Xin Tong, and Jiaolong Yang. Moge: Unlocking accurate monocular geometry estimation for open-domain images with optimal training supervision. *arXiv preprint arXiv:2410.19115*, 2024. 3, 4, 13

[44] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. 3

[45] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11975–11986, 2023. 3

[46] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018. 13

[47] Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre

Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Proceedings of The 7th Conference on Robot Learning*, pages 2165–2183. PMLR, 2023. 3

A. Appendix

A.1. Model Training Details

Dataset	Weight
austin_buds_dataset	0.5
austin_sailor_dataset	2.0
austin_sirius_dataset	0.5
berkeley_autolab_ur5	1.0
berkeley_cable_routing	0.1
berkeley_fanuc_manipulation	1.0
bridge	18.0
dlr_edan_shared_control	0.1
droid	35.0
fmb	1.5
fractal20220817_data	12.0
furniture_bench_dataset	1.5
iamlab_cmu_pickup_insert	0.3
kuka	4.0
language_table	1.5
nyu_franka_play_dataset	0.3
roboset (kinesthetic)	2.0
roboset (teleoperation)	5.0
roboturk	3.0
stanford_hydra_dataset	3.0
taco_play	2.0
toto	1.5
ucsd_kitchen_dataset	0.2
utaustin_mutex	3.0
viola	1.0

Table 3. Open X-Embodiment data mixture for SPEAR-1 pre-training

A.2. VLM training

A.2.1. VQA tasks for VLM pre-training

The Visual Question Answering (VQA) tasks used during VLM pre-training are inspired by VLA embodied tasks and aim to embed as much control-relevant 3D information into the VLM as possible. We use templated question-answer pairs grouped in the following categories:

- **3D keypoints prediction**: Output the 3D coordinates of the closest, furthest and center points of an object with respect to the camera frame
- **3D bounding prediction**: Output the vertices of the 3D bounding box of an object
- **Object-to-object distance prediction**: Output the direct distance between object X and object Y in 3D space as well as its xyz components
- **Object-to-object bounding box prediction**: Output the distance between the bounding box vertices and the centers of object X and object Y
- **Backprojection**: Locate the vertices of the 3D bounding box of an object on the 2D image
- **Chain-of-thought comparisons**: What is the distance from the camera to object X? What is the distance from the camera to object Y? Which object is closer to the camera?

To further encourage the model to 'reason' over the information provided and attend to the right objects, in a single training example we use a random number (between 1 and 4) of question-answer pairs corresponding to different prompts and objects in the scene. To resolve ambiguities, if two instances of the same type of object appear in the image, we filter them out and never ask questions about them.

A.2.2. VLM encoder fusion strategies

We experimented with 2 different strategies to combine the outputs of the SigLIP and MoGe encoders:

1. Concatenating the visual features predicted by both encoders and projecting them via a linear layer to the LLM embedding space. In particular, for SigLIP we take only the tokens at the last layer of the vision encoder, while for MoGe we take the tokens at the last 4 layers of the encoder, following the approach used by MoGe architecutre to decode the features to a 3D point cloud.
2. Using MoGe's predicted 3D point cloud $\mathbf{P}$ in the camera ego pose (in an affine-invariant space) and adding them to the SigLIP encoder features, similar to SpatialVLA [33]. In particular, MoGe's 3D point cloud output $\mathbf{P} \in \mathbb{R}^{H \times W \times 3}$ is embedded to $\mathbf{P'} \in \mathbb{R}^{h \times w \times d}$ through a projector $\psi(\cdot)$, composed of normalization, convolution, sinusoidal embedding $\gamma(x) = (x, \sin(2^0 \pi x), \cos(2^0 \pi x), \ldots, \sin(2^{L-1} \pi x), \cos(2^{L-1} \pi x))$ [29] and an MLP. Finally, the features $\mathbf{F'} = \mathbf{F} + \mathbf{P'}$ are fed to PaliGemma's SigLIP linear projector, where $\mathbf{F} \in \mathbb{R}^{h \times w \times d}$ denotes the features at the SigLIP encoder output.

During our preliminary VLM evaluations we found the first strategy to demonstrate qualitatively better performance on bounding box prediction tasks. In particular, models trained with the second approach struggled to consistently output "grammatically" correct bounding boxes, *e.g.* they would output 22 or 23 depth tokens instead of the required 24. We therefore used the first approach for all VLM pre-training experiments in the main paper.

A.2.3. Depth tokenization

To encode 3D information into text we extend the PaliGemma tokenizer with $N = 1024$ 3D tokens, as 3D coordinates are conceptually different from the existing visual and language tokens. This is in line with PaliGemma's approach of extending Gemma's tokenizer to pixel locations. Each 3D token corresponds to a quantized *distance value* in the range $[z_{\min}, z_{\max}]$, where $z_{\min}$ and $z_{\max}$ are emperically computed as the 1st and 99th quantiles of the 3D point cloud distribution along any of the xyz coordinates.

Dataset	Domain / Subset	# Annotated Images	Segmentation Masks
EgoExo4D [13]	Cooking & Bike Repair	∼200k	GT
Bridge [41]	Robot Demonstrations	∼30k	SAM2 Generated
Total		∼230k	

Table 4. Annotated image counts for training dataset construction, with segmentation mask availability.

Task	0.25	0.50	0.75	1.00
Carrot on Plate (w/ distractors & elevations)	Reach carrot	Pick up carrot	Drop on/near plate	Correctly place on plate
Marker in Cup (w/ distractors)	Reach marker	Pick up marker	Drop on/near cup	Place inside cup
Cover the Pot	–	Pick up lid	Drop lid on pot	Correctly cover pot

Table 5. Scoring rubric for Franka evaluation tasks.

We found the *distance values* in the data to approximately follow a Normal distribution. Therefore, to allow for more accurate tokenization, we compute non-uniform bins with fine-grained discretization around the mean and spread out widths near the tails such that the distribution of 3D tokens approximates a uniform.

We initialize the new token embedding weights from a multivariate normal distribution that has the mean and covariance of the pretrained embeddings [15, 16].

A.2.4. VQA data annotation pipeline

We follow the method described in Section 3.1 in order to enrich 2D images with semantics, segmentation masks and 3D point clouds. We also experimented with GroundingDINO [28] instead of Gemini, but we found the semantic labels produced by GroudingDINO to be a lot less accurate and consistent. We found that if we prompt SAM2 [34] with 2D bounding boxes near the target objects, the output segmentation mask would be of high quality.

We also found that MoGe [43] would output depths at different scales depending on the input image size. Therefore, we resized all our images to 840x630 for MoGe point cloud annotations.

For 3D bounding box estimation, after filtering the 3D point cloud with a segmentation mask, we run statistical outlier removal and esitmate an oriented 3D bounding box around the remaining points using Open3D [46]. To facilitate learning, we order all 8 bounding box vertices in a consistent way, starting based on their spatial coordinates with respect to the camera frame.

A.3. VLA training

A.3.1. VLA training details

During VLA training we use an action chunk of size $H = 5$ and frequency of 5Hz. As not all datasets in Open X-Embodiment provide action labels at 5Hz, we downsample or upsample the actions accordingly via linear interpolation. This is done with the goal to encourage the model to share knowledge across datasets with different control frequencies and embodiments instead of 'memorize' each dataset separately.

A.3.2. Flow matching details

To address the inherent double coverage of 3D rotations by the unit quaternion group $\mathbb{S}^3$, we ensure that all quaternions used during training and inference lie in the same half-space defined by $\Re(\mathbf{q}) = \mathbf{q}_w > 0$.

Quaternion integration. Given a unit quaternion $\mathbf{q}_t \in \mathbb{S}^3$ and its time derivative $\dot{\mathbf{q}}_t \in \mathbb{R}^4$, we can compute the angular velocity of rotation via $\boldsymbol{\omega}_t = 2.0 \cdot \Im(\mathbf{q}_t^* \otimes \dot{\mathbf{q}}_t) \in \mathbb{R}^3$. For a small time step Δt, the corresponsing delta rotation is given by a rotation vector around the unit axis $\hat{\boldsymbol{\omega}} = \boldsymbol{\omega}/||\boldsymbol{\omega}||$ over an angle $\Delta\phi = ||\boldsymbol{\omega}||\Delta t$. The corresponding delta quaternion is given by

$$\Delta\mathbf{q} = \left[\cos\left(\frac{\Delta\phi}{2}\right), \hat{\boldsymbol{\omega}}\sin\left(\frac{\Delta\phi}{2}\right)\right] \tag{6}$$

The integrated unit quaternion is then given by $\mathbf{q}_{t+\Delta t} = \mathbf{q}_t \otimes \Delta\mathbf{q} \in \mathbb{S}^3$,

A.4. Real-world robot task progression scoring

We provide the detailed task progression scoring for all real-world evaluations on the Franka robot in Table 5.