
Vision-Language-Action Instruction Tuning: From Understanding to Manipulation

Shuai Yang^{2,3†*} **Hao Li**^{1,3*} **Yilun Chen**³ **Bin Wang**^{2,3} **Yang Tian**³ **Tai Wang**³
Hanqing Wang³ **Feng Zhao**¹ **Yiyi Liao**² **Jiangmiao Pang**³

¹University of Science and Technology of China, ²Zhejiang University,

³Shanghai Artificial Intelligence Laboratory

Abstract

To operate effectively in the real world, robots must integrate multimodal reasoning with precise action generation. However, existing vision-language-action (VLA) models often sacrifice one for the other, narrow their abilities to task-specific manipulation data, and suffer catastrophic forgetting of pre-trained vision-language capabilities. To bridge this gap, we introduce **InstructVLA**, an end-to-end VLA model that preserves the flexible reasoning of large vision-language models (VLMs) while delivering leading manipulation performance. InstructVLA introduces a novel training paradigm, *Vision-Language-Action Instruction Tuning (VLA-IT)*, which employs multimodal training with mixture-of-experts adaptation to jointly optimize textual reasoning and action generation on both standard VLM corpora and a curated 650K-sample VLA-IT dataset. On in-domain SimplerEnv tasks, InstructVLA achieves 30.5% improvement over SpatialVLA. To evaluate generalization, we introduce SimplerEnv-Instruct, an 80-task benchmark requiring closed-loop control and high-level instruction understanding, where it outperforms a fine-tuned OpenVLA by 92% and an action expert aided by GPT-4o by 29%. Additionally, InstructVLA surpasses baseline VLMs on multimodal tasks and exhibits inference-time scaling by leveraging textual reasoning to boost manipulation performance in both simulated and real-world settings. These results demonstrate InstructVLA’s potential for bridging intuitive and steerable human-robot interaction with efficient policy learning. [Project website](#).

1 Introduction

Large-scale pre-training has produced versatile foundation models in computer vision (CV) [1, 2, 3, 4, 5, 6] and natural language processing (NLP) [7, 8, 9, 10, 11]. Inspired by this success, recent Vision-Language-Action (VLA) models [12, 13, 14, 15, 16, 17, 18] initialize from large vision-language models (VLMs) [19, 20, 21, 22, 23, 24] and train on large-scale embodied data [25, 26] to enhance generalization in robotic manipulation. While these VLAs demonstrate strong performance in robotic manipulation tasks, they are susceptible to *catastrophic forgetting* [27, 28], which gradually diminishes the rich multimodal reasoning capabilities inherited from their web-scale pre-trained vision-language backbones. Two challenges contribute to this issue: (1) existing large-scale real-world robotic datasets mostly lack diverse human instructions across varied task scenarios, restricting training to simple, templated commands (e.g., “open the drawer”); and (2) training solely on domain-specific robotic data accelerates the erosion of general multimodal understanding, limiting the model’s ability to handle diverse inputs, user feedback, and free-form instructions [29].

To mitigate catastrophic forgetting when finetuning VLMs into VLAs, prior work primarily adopts two strategies. The first aims to jointly preserve general multimodal capabilities while learning diverse manipulation skills. Models such as ChatVLA [30] and Magma [31] follow this approach by jointly

*Equal contribution, †the work is completed during an internship at³, email: shuaiyang2003@gmail.com

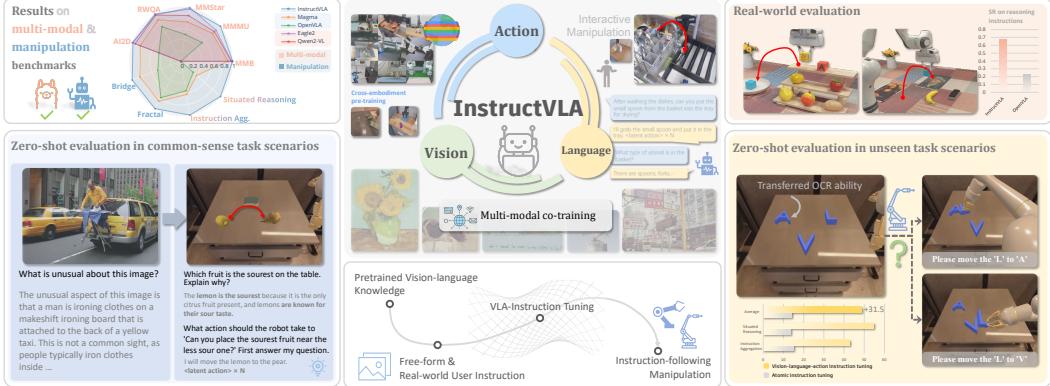


Figure 1: **Method overview.** InstructVLA integrates robust multimodal understanding with precise instruction-driven robotic control, leveraging the world knowledge of VLMs. The core training strategy, vision-language-action instruction tuning, enhances manipulation by enabling the model to perform vision language reasoning before generating actions.

training on vision-language and manipulation data. However, this approach often neglects complex embodied reasoning. The second strategy focuses on tightly integrating embodied reasoning into manipulation datasets to transfer VLM capabilities. Methods such as ECoT [14] and Emma-X [32] embed chain-of-thought (CoT) reasoning into manipulation datasets. However, these methods are built on action-pretrained architectures [12] and structured reasoning patterns (plan, subtask, etc.), which inherently constrain general multimodal capabilities. The extent to which the VLM capabilities translate into action generation in embodied contexts remains largely unexplored.

To tackle this issue, we propose **InstructVLA**, a generalist VLA model that extends pretrained VLM for accurate action generation while retaining strong multimodal understanding, as illustrated in Figure 1. InstructVLA adopts a training paradigm specifically designed to bridge vision-language understanding with action generation by treating language-steered action generation as an integral component of instruction following. To this end, we curate the Vision-Language-Action Instruction Tuning (VLA-IT) dataset, consisting of 650K human-robot interactions annotated with diverse instructions, scene captions, and question-answer pairs grounded in high-quality manipulation tasks [33, 34]. The training process follows a two-stage paradigm: (1) *Action Pretraining*, which trains a VLM-driven action expert using latent action representations distilled from language-based motion descriptions; and (2) *Vision-Language-Action Instruction Tuning*, which unifies language and latent action generation through a trainable mixture-of-experts(MoE) adaptation framework. This framework is jointly trained on multimodal datasets [19, 35, 36], manipulation datasets, and the curated VLA-IT corpus, enabling the automatic switch between textual reasoning and action generation, thereby effectively leveraging vision-language understanding for action execution.

To validate the generalist performance of InstructVLA, we introduce the SimplerEnv-Instruct benchmark, a manually designed evaluation suite featuring 80 zero-shot manipulation tasks. It encompasses both closed-loop manipulation tasks and high-level instruction reasoning, involving either situated understanding or decomposition into actionable subtasks. With its thinking ability during manipulation, InstructVLA outperforms the fine-tuned OpenVLA baseline by 92% and achieves a 29% improvement over an action expert model assisted by GPT-4o on SimplerEnv-Instruct, demonstrating its effectiveness in instruction following and task decomposition. Furthermore, InstructVLA surpasses similarly sized VLMs in multimodal performance and shows a 27% improvement over Magma in closed-loop manipulation [37]. Our contributions can be summarized as follows:

- We propose **InstructVLA**, a VLA architecture and training pipeline that **emphasizes the importance of language capability in VLAs** by efficiently preserving pretrained vision-language knowledge from VLMs while integrating manipulation as a component of instruction following.
- We design a **practical data and evaluation pipeline** for vision-language-action instruction following, supported by 650K tailored VLA-IT annotations and a manually curated benchmark suite, enabling evaluation of VLAs’ instruction generalization capabilities.
- InstructVLA achieves leading performance across robotic manipulation tasks, multimodal benchmarks, and real-world deployments, enabling intuitive and controllable manipulation.

2 Related Works

Policy learning at scale. Following the success of CV [1, 2] and NLP [11, 9], recent research [38, 34, 13, 39, 40] shows that robot policies improve when trained in large heterogeneous datasets. RT1 [34] and RT-2 [13], trained in large-scale real-world demonstrations, achieve strong in-domain accuracy and zero-shot transfer. Works such as Octo [41] and RT-X [25] extend this approach by aggregating the largest open-source manipulation datasets [25]. Some methods, such as LAPA [42], Seer [43], and Moto [44], use video generation and inverse dynamics to learn scalable motor representations. In the VLA domain, models are typically initialized from pretrained vision-language models [12, 15, 13] leveraging prior visual-linguistic alignment instead of learning from scratch. Further, methods such as RT-Trajectory [45] and GraspVLA [46] jointly train intermediate manipulation representations such as trajectories or bounding boxes using a combination of real, simulated, and web data to guide action generation and enhance generalization.

Vision-language-action models. Recent foundation models [13, 12, 15, 47, 44, 48, 49] integrate perception, language, and robot manipulation into a single network, using two main architectures. Autoregressive models treat actions as discrete tokens: RT-2 [13] co-trains a web-scale VLM on robot trajectories, transferring semantic knowledge to manipulation, while OpenVLA [12] and SpatialVLA [15] follow a similar token-based control approach. FAST tokenization [49] compresses motion sequences to manage length. In contrast, flow-based VLAs avoid discretization; for example, π_0 [47] and GR00T [48] generate actions through continuous flow matching [50], while CogACT [17] and CronusVLA [18] use diffusion [51]. Hybrid approaches, like RoboDual [52], combine generalist action models with specialist action experts. Although flow-based methods [47, 48, 18, 17] often achieve superior performance, they typically neglect the integration of autoregressive text reasoning [13], which is crucial for leveraging the VLM’s semantic capabilities. In contrast, our model unifies autoregressive VLM language generation with the flow-based action generation, demonstrating efficient co-training of language and action at scale.

Robot policies with hierarchical decision making. Training multitasking policies for decision making and planning in complex environments remains a significant challenge. Leveraging the capabilities of pretrained VLMs and LLMs provides a simple yet effective solution. For example, SayCan [53] uses a frozen LLM to identify subtasks. RT-H [54], Steer [55], and Hi-robot [29] ground actions through language, training a language-steerable VLA model. RT-Trajectory [45] and RoboGround [56] bridge planning and action generation with intermediate representations. These methods use language, trajectories, or bounding boxes as interfaces between high-level understanding and low-level action generation. Designing a general interface without ambiguity remains difficult; with LCB [57] and Helix [58] inserting learnable latent tokens between high-level reasoning backbones and low-level policy heads, enabling end-to-end finetuning without manual skill libraries. However, these methods do not integrate textual reasoning and action planning in a single model, and designs of dual models [29] often incur additional computational costs. Our approach preserves the VLM’s reasoning capabilities while enabling dynamic switching between reasoning and action execution, resulting in efficient and controllable manipulation.

3 InstructVLA

We propose **InstructVLA** (Figure 2), a unified model for joint language and action generation. Section 3.1 details the architecture with dynamic reasoning and execution switching, along with inference strategies, while Section 3.2 outlines the training paradigm for VLA instruction following.

3.1 Architecture

Embodied VLM for textual and latent action generation. We propose a unified framework that enables simultaneous multimodal reasoning and language-steered latent action planning using a single VLM (Figure 2 (1) and (2)). The model produces textual outputs to preserve the strong language understanding and multimodal inference capabilities of the pretrained VLM, while subsequently generating latent action representations for downstream manipulation. To support action planning, we introduce N learnable action queries $Q \in \mathbb{R}^{N \times D}$, which attend to the VLM’s hidden states and extract task-relevant latent action $C \in \mathbb{R}^{N \times D}$, where D is the VLM hidden dimension. Our implementation builds on the compact and efficient Eagle2-2B backbone [59], with a tailored training

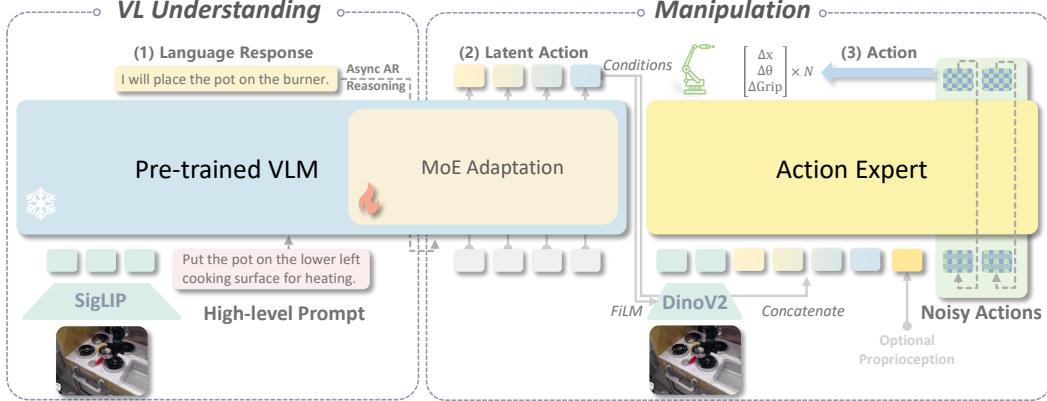


Figure 2: **Overview of the InstructVLA.** InstructVLA integrates the multimodal reasoning capabilities of a vision-language model with robotic manipulation. Generation consists of three steps: (1) asynchronous auto-regressive reasoning by the VLM, (2) latent action generation, and (3) action decoding. A MoE adaptation enables the VLM to alternate between reasoning and latent action prediction. The flow matching action expert decodes the final actions, conditioned on latent actions.

strategy described in Section 3.2. The model is supervised with cross-entropy on language output with loss \mathcal{L}_{LM} .

Mixture of adaptation experts for language-steered latent action. A key challenge is enabling the model to seamlessly alternate between reasoning and manipulation at inference time. To this end, we adopt a Mixture-of-Experts (MoE) design [60], which allows adaptive reweighting of expert modules based on input context and reasoning mode, thereby integrating multimodal reasoning with language-steered latent action. Specifically, LoRA [61] modules are employed as experts within the LLM backbone, preserving pretrained capabilities while ensuring efficient inference. A scale head [62] predicts gating coefficients λ_i for each expert by classifying the hidden state, enabling the model to adaptively blend their outputs. The resulting hidden states for K experts are computed as $h = W_0 x + \sum_{i=0}^K B_i A_i x \cdot \alpha_i \cdot \lambda_i$, where W_0 is the original weight, x denotes input, $A_i \in \mathbb{R}^{r \times d}$ and $B_i \in \mathbb{R}^{d \times r}$ are the LoRA parameters, α_i is the LoRA scaling factor.

Flow model as an efficient action expert. To further decouple low-level control from high-level understanding, the action expert is designed to generate actions from image observations conditioned on VLM-derived intentions. It takes image features from DINOv2 [1], latent actions from the VLM, noisy action embeddings and optional information such as proprioception, and fuses these with a simple transformer architecture [11] with block-wise causal attention. Specifically, non-causal attention is applied within each input, and causal attention between input types. The DINOv2 vision encoder, further enhanced with feature-wise linear modulation (FiLM) [63], plays a crucial role in directing actions to spatial and contextual input. The flow matching objective [47] is used to supervise action learning, as detailed in Appendix E.2.

Inference. InstructVLA integrates language and action generation in a single model. The following techniques are designed for InstructVLA to enhance speed. (1) *Decoding strategies.* To mitigate the latency of autoregressive decoding, textual responses are generated via greedy search until the first action query token appears. The remaining action queries are then decoded in parallel within a single forward pass of the VLM. (2) *Language response and latent action caching.* We decouple language response from action generation by caching textual outputs across multiple action steps, leveraging their temporal stability. InstructVLA also supports cache latent actions, which reduces the number of VLM forward with minimal performance impact compared with ECoT [14] (see Appendix A.3).

3.2 Two-Stage Training Recipe

The first stage involves efficient pretraining of an action expert aligned with latent action embeddings from the VLM via *Action Pretraining*, followed by *Vision-Language-Action Instruction Tuning* to bootstrap the action generation process by reactivating the VLM’s multimodal reasoning abilities.

Original Dataset	Embodied Scene Understanding		Instruction Understanding and Planning	
	Scenario Caption	Question Answering	Command Rewriting	Context Creation
	A kitchen counter with various objects, including colorful plastic food items, a cup, and utensils. There is also a stove with some kitchen tools scattered around.	What surface is the cutting board placed on? The stove.	Place the cutting tool on the cutting board. I'll put the knife on the cutting board.	I'm about to prepare a sandwich. Could you get the tool ready on the cutting board? I'll place the knife on the cutting board. Situational
	A table with a cloth, a spoon, and various kitchen items including a microwave and cans	What is the utensil made of that is on the table? The spoon is made of metal and plastic.	Place the tool made of metal and plastic onto the fabric. I will put the spoon on the cloth.	After you are done with washing the spoon, please dry it on the cloth. I will place the spoon on the cloth. Material
	A drawer is open, revealing a Coke can inside, while other items are placed nearby on the table.	What beverage is in the middle drawer? A Coke can.	Retrieve the red can from the drawer and set it on the counter. I will get the soda can from the drawer.	Please open the middle drawer, take the Coke, and place it on the table. The middle drawer is open. I will take the Coke out of the drawer and set it on the table. Appearance Noval Action Long horizon

Figure 3: **Vision-language-action instruction tuning data examples.** Annotations focus on: (1) improving scene understanding and (2) learning instruction following and planning.

Stage 1: Action pre-training. InstructVLA is pre-trained using heterogeneous manipulation data [34, 33]. To distill the knowledge from the VLM for manipulation, the model is trained to predict both actions and rule-based annotated language motion (Section 4.1), with the latter supervised via cross-entropy loss. Due to the stability of flow matching and the next token prediction, the final loss is the direct sum of both losses as $\mathcal{L} = \mathcal{L}_{LM} + \mathcal{L}_{FM}$. During this stage, only the input and output embedding of the action queries and action LoRA adapter on the LLM backbone are tuned, consisting of 650M parameters. The model trained in this stage is named the “Expert”.

Stage 2: Vision-language-action instruction tuning. We extend the concept of visual instruction tuning [19] with a simple approach to train InstructVLA. Our observation is that once the action expert is pretrained to follow the latent actions from the VLM, further adapting the LLM backbone enables the model to handle manipulation tasks with more complex instructions and generate appropriate responses. In this stage, the action expert remains frozen, a new language LoRA adapter and scale head of the MoE-adaptation are added. The MoE module is the only trainable parts, comprising 220M parameters. We detail the data pipeline for vision-language-action instruction tuning in Section 4.1; this data bridges pretrained vision-language capabilities with embodied task scenarios. We further co-train the model using multimodal datasets [36, 19, 35] to bootstrap multimodal understanding. The resulting model is referred to as the “Generalist”, reflecting its combined vision-language and manipulation capabilities.

4 VLA Dataset and Benchmark

4.1 InstructVLA Tuning Dataset

We curate diverse hierarchical language annotations from large-scale manipulation datasets [34, 33], including language motion [54] and rule-based labels for pretraining, along with the VLA-IT (Vision-Language-Action Instruction Tuning) dataset for instruction tuning and reasoning transferring.

Language motion pre-training data. Language motion [54] provides intuitive linguistic descriptions of basic end-effector movements, which can be distilled into latent actions. We compute the relative displacement of the end-effector between the t -th and $(t + W)$ -th steps, using a window size W . The final labels, such as “move right and open gripper,” provide supervision for VLM.

Vision-language-action instruction tuning data. To enable language-steerable VLA models, it is essential to curate diverse instructions, model responses, and reasoning patterns. We categorize our data into four types as illustrated in Figure 3. For embodied scene understanding: (1) *Scenario captioning* provides descriptions of the robot’s environment (2) *Question answering* targets scene understanding through consistent QA pairs across an episode. Together, they bridge vision-language annotations with embodied scenes. For instruction understanding and latent action planning: (3) *Command rewriting* introduces instructional diversity through paraphrasing, attribute-based references and varied vocabulary. (4) *Context creation* generates implicit user goals or progress cues in multi-step tasks, requiring the robot to infer intent. These annotations support joint VLA reasoning.

We use GPT-4o [64] to annotate data with three frames from each episode, along with the corresponding instruction. Ground-truth instruction is crucial for annotation accuracy, emphasizing that even

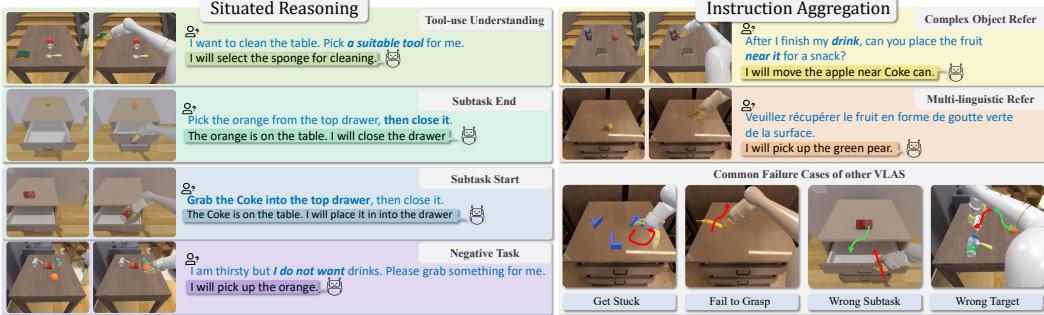


Figure 4: **Simpler-Instruct.** We visualize six representative test cases, with instructions and responses from InstructVLA during evaluation. Top four failure modes of other VLAs are listed.

state-of-the-art VLMs can make errors in embodied tasks, leading to a performance gap when using GPT-4o as an instruction interpreter for such tasks. Additional details of the dataset analysis and prompt templates are provided in Appendix C.

4.2 SimplerEnv-Instruct

Building upon the real-to-sim SimplerEnv platform [37], we propose SimplerEnv-Instruct, a **manually designed** benchmark for evaluating the instruction-following and reasoning capabilities of VLA models in a **zero-shot setting**. The benchmark comprises two hierarchical levels: instruction aggregation (50 tasks) and situated reasoning (30 tasks), totaling 1.1K trials, as shown in Figure 4.

Instruction aggregation. The instruction aggregation tasks focus on command diversity, including new verbs, multilingual expressions, object references, sentence rephrasing, and novel objects. **Situated reasoning.** Situated reasoning tasks evaluate the model’s ability to infer intent when instructions are implicit. For example, “I want to clean the table. Pick a suitable tool for me.” requires the model to identify and retrieve the correct object (e.g. a sponge) through reasoning. We further incorporate subtask identification, where each subgoal is conditioned on both the instruction and the environment, capturing the complexity of long-horizon tasks.

The benchmark task design adheres to two principles: (1) tasks should evaluate the transfer of in-domain manipulation skills to novel scenarios, and (2) instructions must be human-interpretable. To assess the first, we filter basic tasks and objects to ensure models exhibit correct intent, and then annotate novel instructions to increase task difficulty. To validate the second, we conduct cross-checks among human annotators to ensure instruction clarity and naturalness.

5 Experiment

Benchmarks. (a) *Multimodal*. We use the automatic evaluation from VLMEvalKit [65] including MMU(Val) [66], MMStar [67], MME [68], OCRBench [69], HallB(Avg) [70], MMB(Dev En V1.1) [71], TextVQA [72], DoCVQA [73], InfoVQA [74], AI2D [75], ChartQA [76] and RWQA [77]. These benchmarks collectively evaluate diverse multimodal capabilities, including general visual question answering, document, infographic and chart understanding, OCR reasoning, and hallucination robustness. (b) *SimplerEnv* [37] features real-to-sim evaluation on large-scale manipulation datasets [34, 33] with visual matching and variance aggregation settings to evaluate generalization ability. (c) *SimplerEnv-Instruct*, detailed in Section 4.2, extends the SimplerEnv with more novel objects, tasks, and instructions, providing a broader testbed to evaluate the instruction generalization of VLAs. We additionally evaluate language-capable baselines as detailed in Appendix A.1.

Training details. The VLM is trained with a resolution of 448×448 following [59], while the action expert operates at 224×224 as in [12], using a fixed learning rate of 5e-5 without warm-up. The action expert employs a 12-layer transformer backbone with a hidden size of 768. Following [47], a β distribution is used to enhance accuracy on the noisier time steps. During Stage 2 finetuning, manipulation and multimodal understanding are trained in an interleaved manner. Owing to InstructVLA’s design,

Table 1: **Multimodal understanding.** #Params is the size of LLM backbone. S. denotes robot state.

Methods	#Params	Multi-modal Understanding Benchmarks							VQA Benchmarks						
		MMMU ^{Val}	MM-Vet	MMStar	MME ^P	OCRBench	HallB	MMB	TextVQA	DocVQA	InfoVQA	AI2D	ChartQA	RWQA	
LLaVA-OV [78]	8B	47.9	50.6	61.9	1993.6	622	31.6	80.9	-	-	-	82.4	80.9	69.9	
Bunny [36]	8B	43.4	39.1	45.4	1987.7	444	37.7	72.9	-	-	-	69.4	30.1	60.4	
Eagle2 [59]	2B	43.1	53.8	56.4	1572.1	818	45.8	74.9	79.1	88.0	65.8	79.3	82.3	63.1	
Qwen2-VL [79]	2B	41.1	51.5	48.0	1872.0	809	41.7	74.9	74.9	88.6	61.4	74.7	73.5	62.9	
OpenVLA [12]	7B	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
OpenVLA (FT)	7B	26.0	9.1	28.2	87.6	2.5	8.4	18.9	2.5	29.2	43.4	35.8	1.4	47.2	
ECoT [14]	7B	16.2	0.0	19.1	0.0	0.0	3.1	0.9	0.0	2.2	0.0	0.0	0.0	29.8	
Magma [81]	8B	38.8	34.1	41.3	1496.5	518	38.0	69.7	66.5	65.4	45.2	66.1	61.8	56.5	
Generalist	2B	44.8	47.5	54.9	1611.2	795	47.0	76.6	75.6	84.4	63.8	78.1	79.7	64.4	
Generalist(S.)	2B	44.2	51.4	55.6	1612.6	816	43.4	77.7	76.6	85.5	64.7	78.9	81.5	63.7	

Table 2: **Robotic manipulation.** Google Robot and WidowX Robot denote two embodiments in SimplerEnv. For SimplerEnv-Instruct, we focus on two reasoning levels instead of embodiments. Magma[†] denotes evaluation with sampling.

Methods	Google Robot						WidowX Robot						Avg	SimplerEnv-Instruct			
	Open/Close Drawer		Put in Drawer		Pick Coke Can		Move Near		Put Spoon		Put Carrot			Instruction Aggregation	Situated Reasoning	Avg	
	VM	VA	VM	VA	VM	VA	VM	VA	VM		VM						
RT-1-X [25]	59.7	29.4	21.3	10.1	56.7	49.0	31.7	32.3	0.0	4.2	0.0	26.8	-	-	-	-	
RT-2-X [25]	25.0	35.5	3.7	20.6	78.7	82.3	77.9	79.2	-	-	-	-	-	-	-	-	
Octo-Base [41]	22.7	1.1	0.0	0.0	17.0	0.6	4.2	3.1	15.8	12.5	0.0	7.0	-	-	-	-	
RoboVLMs-2B [16]	43.5	10.6	27.8	0.0	77.3	75.6	61.7	60.0	45.8	20.8	4.2	38.8	-	-	-	-	
TraceVLA-4B [80]	35.4	37.5	0.0	0.0	69.7	75.4	70.8	67.8	8.3	0.0	12.5	34.3	-	-	-	-	
OpenVLA-7B [12]	63.0	28.8	0.0	0.0	18.0	60.8	56.3	67.7	4.2	0.0	0.0	27.2	14.8	13.6	14.2		
TraceVLA-3B [80]	63.1	61.6	11.1	12.5	45.0	64.3	63.8	60.6	12.5	16.6	16.6	38.9	-	-	-	-	
SpatialVLA-3B [15]	57.4	41.8	0.9	9.1	86.0	88.0	77.9	72.7	16.7	25.0	29.2	45.9	-	-	-	-	
Expert	47.2	60.6	61.1	40.2	87.7	76.0	68.3	77.3	45.8	20.8	20.8	52.9	20.8	10.4	15.6		
Expert(S.)	46.3	56.1	46.3	69.8	92.7	93.2	70.0	77.9	50.0	50.0	25.0	59.9	-	-	-	-	
Magma-8B [31]	9.7	5.8	0.0	0.0	46.0	46.4	60.0	82.0	45.8	33.3	8.3	30.5	15.5	9.9	12.7		
Magma-8B [†] [31]	56.0	53.4	6.4	18.5	83.7	68.8	65.4	65.7	35.5	31.0	12.7	43.6	26.2	21.4	23.8		
OpenVLA (FT) 7B	63.9	42.6	3.7	6.9	62.3	88.7	65.8	67.7	12.5	33.3	4.2	39.0	28.3	19.5	23.9		
OpenVLA (FT&GPT)	-	-	-	-	-	-	-	-	-	-	-	-	38.8	32.4	35.6		
Generalist	55.6	57.7	50.0	38.1	78.0	91.0	52.1	69.8	33.3	29.2	12.5	49.4	43.3	48.8	46.0		
Generalist(S.)	43.6	52.8	40.3	56.9	90.2	93.9	70.0	78.9	50.0	41.7	12.5	55.4	49.5	42.6	46.1		

multimodal capabilities are retained without requiring tuning of the multimodal-to-manipulation ratio; we use a 1:7 ratio, twice the imbalance used in ECoT (1:3)[14]. Appendix E gives more details.

Baselines. We categorize the baselines into three groups: (1) *Multimodal VLMs*, including LLaVA-OV[78], Bunny[36], Eagle2 [59], and Qwen2-VL [79]; (2) *VLA models*, including RT-1-X and RT-2-X from OXE [25], Octo [41], RoboVLMs [16], SpatialVLA [15], TraceVLA [80], and OpenVLA [12]; (3) *Generalist VLA models*, including Magma [81], OpenVLA fine-tuned (FT) from its official generalist pretrained model on both robotic and multimodal data, and ECoT(Bridge) [14]. For both language and action generation, InstructVLA and other baselines use a temperature of 0 and greedy search without sampling to expedite generation. We re-evaluate Magma using its official checkpoint². For ECoT, we report only its multimodal results due to its real-to-sim domain gap [14].

5.1 Main Results

We present our main results in Tables 1 and 2. In Table 1, using the same generalist model InstructVLA (generalist), it not only outperforms the co-trained baseline Magma-8B [31], but also exceeds its base model Eagle2 [59] and Bunny (VLM data corpus) [36] across multiple multimodal benchmarks including MMMU, MMB, and RWQA. In Table 2, InstructVLA (expert) outperforms expert baselines SpatialVLA by 30.5%, while InstructVLA (generalist) maintains strong performance on SimplerEnv's atomic instructions and further achieves a 29.5% improvement on the SimplerEnv-Instruct over SOTA baseline (OpenVLA with GPT-4o).

However, we observe that finetuning OpenVLA on multimodal and manipulation datasets does not fully restore its original multimodal capabilities, although it does improve task performance. Its performance can be further enhanced by integrating GPT-4o as an API-based system-2 module to rephrase instructions (OpenVLA (FT&GPT)). However, GPT-4o faces the same challenges in accurate instruction rewriting as noted in Section 4.1, and fails to outperform InstructVLA (Generalist).

²We observe a notable performance gain for Magma when using sampling. Accordingly, we report its official score on SimplerEnv and re-evaluate its performance on SimplerEnv-Instruct under the sampling setting.

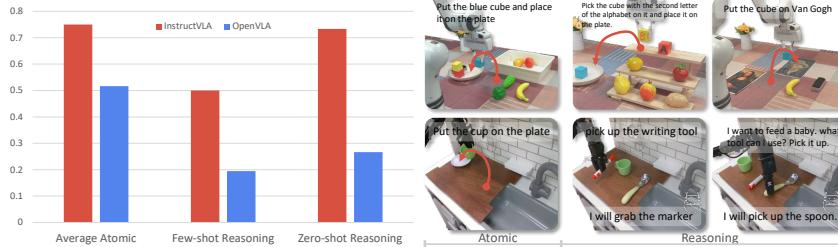


Figure 5: **Real-world experiments.** “Atomic” refers to atomic instructions, while “Reasoning” denotes situated reasoning. For the Bridge settings, InstructVLA’s responses are presented.

Methods such as Magma, which co-train both abilities of the VLM, better preserve multimodal ability, but still fail to match the performance of our approach. ECoT relies solely on textual chain-of-thought reasoning over manipulation datasets and lacks the capability for multimodal question answering. We observe that it consistently generates manipulation-style CoT responses, without demonstrating effective instruction-following ability.

5.2 Real-world Experiments

To evaluate InstructVLA in real-world scenarios, we conduct zero-shot experiments on the WidowX-250 Arm and few-shot experiments on the Franka Research 3 robot, as shown in Figure 5. The few-shot tasks involve spatially demanding pick-and-place from a rack and cluttered tabletop organization. We use GPT-4o to annotate VLA-IT dataset on the collected few-shot data. InstructVLA is fine-tuned using the proposed training recipe, while OpenVLA is jointly trained on both atomic skill and VLA-IT datasets until the action token accuracy reaches 95% [12]. The zero-shot tasks are set in the kitchen environment following Bridge Dataset [33].

Each scenario includes both atomic and reasoning instructions. Atomic settings focus on in-domain objects and instructions with an emphasis on spatial generalization to assess baseline VLA capabilities. Both models perform comparably on direct instruction with in-domain objects; InstructVLA achieves a 23.3% success rate improvement over OpenVLA. For reasoning settings such as celebrity recognition, OCR, and tool-use inference, OpenVLA exhibits a substantial performance drop. In contrast, InstructVLA outperforms it by 41.7% in few-shot and 46.7% in zero-shot settings. Detailed experimental setups are provided in Appendix G.

5.3 Ablation Studies

We conduct ablation studies guided by two central questions: (1) How can manipulation and multimodal understanding be effectively integrated into a single model through architectural design and training strategies? (2) To what extent does vision-language comprehension enhance manipulation performance in complex scenarios? Through targeted ablations, we examine the impact of key architectural and training decisions on these capabilities.

5.3.1 Action Ability Integration

Effects of language motion data for pre-training. As shown in Figure 6 (a), introducing “language motion” (auxiliary textual descriptions of low-level actions) enhances the VLM’s ability to associate visual cues with manipulation primitives, leading to a 10.5% improvement in overall success rate.

Effects of latent action queries. Latent action tokens are a key design component for decoupling high-level VLM planning from low-level action generation. As shown in Figure 6 (b), we vary the number of tokens from 16 to 128. Too few tokens limit behavioral diversity, while too many reduce training efficiency. A setting of 64 offers a good trade-off under our configuration.

Ablation on action expert designs. As shown in Figure 6(c), while the base VLM offers general visual understanding, fine-grained perception for manipulation tasks demands richer representations. Removing the DINOv2-based ViT encoder from the action expert results in a 50.0% performance drop, highlighting its critical role in capturing task-relevant visual cues. Incorporating FiLM enhancement to the ViT encoder yields a further 15.3% improvement by modulating visual features with latent



Figure 6: **Ablation Studies.** Ablation studies are grouped into two perspectives: **(a-d) Action ability integration:** analysis of how design choices in data (language motion), representation (latent action tokens), vision encoders, and finetuning strategies influence manipulation performance. **(e-g) Multimodal ability transfer:** analysis of how vision-language understanding contributes to manipulation by VL-to-action learning, instruction data scaling, and inference time thinking.

actions, enhancing task alignment. As shown in Table 2, the expert model with robot state generally performs better.

Comparison with fully finetuning VLA. As shown in Figure 6 (d), FFT denotes full finetuning of the VLM backbone with latent actions using the same data recipe, but without MoE adaptation or multi-stage training. In contrast, InstructVLA employs our proposed architecture and two-stage training strategy, yielding 12.5% performance gain over Magma on SimplerEnv. This highlights the effectiveness of our design for integrating manipulation capabilities into VLMs.

5.3.2 Multimodal Ability Transfer

Ablation on VL-to-action learning. As shown in Figure 6 (e), to isolate the effect of instruction adaptation, we compare two settings: (i) finetuning only the VLM with the action expert frozen, and (ii) jointly finetuning both components. Freezing the action expert yields performance comparable to joint tuning, while reducing the number of trainable parameters and accelerating training. This suggests that InstructVLA can effectively adapt to complex textual inputs by fine-tuning only the VLM, without altering the pretrained action expert.

Effects of instruction data scaling. As shown in Figure 6 (f), we evaluate performance on the SimplerEnv-Instruct benchmark as we scale the amount of VLA-IT annotations from 25% to 100% examples. Instruction-following accuracy exhibits a logarithmic improvement. Notably, situated reasoning tasks, where the model must ground objects and goals in context, benefit more from larger annotation sets, underscoring the bootstrapped reasoning ability from VLMs. In contrast, pertrained OpenVLA fine-tuned on VLA-IT benefits primarily from increased instruction diversity, but exhibits limited improvement on situated reasoning tasks, due to catastrophic forgetting of its original vision-language capabilities. These results indicate that InstructVLA’s advantage stems not only from additional training data but also from its model design. Further ablations and discussions for OpenVLA are provided in Appendix A.2.

Training and inference strategies in different models. As shown in Figure 6 (g), OpenVLA suffers from catastrophic forgetting, leading to suboptimal performance when directly fine-tuned with VL or VLA-IT dataset. Magma, despite being co-trained on multimodal datasets, demonstrates limited benefits from its vision-language capabilities on reasoning tasks. In contrast, our generalist model, trained on the VLA-IT corpus, outperforms the expert model, which is capable for atomic instructions, on the SimplerEnv-Instruct benchmark. We denote language generation during manipulation as Think. Enabling thinking in the generalist model results in a further 36.1% performance gain over direct instruction execution and surpasses InstructVLA-expert paired with GPT-4o as an external interpreter. Further analysis of the role of thinking is discussed in Appendix A.3.

6 Conclusion

We present InstructVLA, a unified VLA model that integrates multimodal reasoning and action generation. By preserving the generalization capability of pretrained VLMs and aligning language, perception, and control in a cohesive process, InstructVLA offers a solution to catastrophic forgetting and disjoint reasoning for VLAs. While effective, the current implementation leverages only minimal inputs: a single image and instruction. Incorporating additional sensory modalities (e.g., depth) could further enhance performance. Despite this, our end-to-end data and training pipeline enables state-of-the-art performance across manipulation tasks, multimodal benchmarks, and real-world deployments, paving the way for more generalizable, interpretable, and interactive robots.

References

- [1] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al. “Dinov2: Learning robust visual features without supervision”. In: *arXiv preprint arXiv:2304.07193* (2023).
- [2] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer. “Sigmoid loss for language image pre-training”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 11975–11986.
- [3] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. “Learning transferable visual models from natural language supervision”. In: *International conference on machine learning*. PMLR. 2021, pp. 8748–8763.
- [4] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick. “Segment Anything”. In: *arXiv:2304.02643* (2023).
- [5] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, E. Mintun, J. Pan, K. V. Alwala, N. Carion, C.-Y. Wu, R. Girshick, P. Dollár, and C. Feichtenhofer. “SAM 2: Segment Anything in Images and Videos”. In: *arXiv preprint arXiv:2408.00714* (2024). URL: <https://arxiv.org/abs/2408.00714>.
- [6] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan, et al. “Grounded sam: Assembling open-world models for diverse visual tasks”. In: *arXiv preprint arXiv:2401.14159* (2024).
- [7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 2019, pp. 4171–4186.
- [8] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer”. In: *Journal of Machine Learning Research* 21.140 (2020), pp. 1–67. URL: <http://jmlr.org/papers/v21/20-074.html>.
- [9] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. “Language models are unsupervised multitask learners”. In: *OpenAI blog* 1.8 (2019), p. 9.
- [10] J. Bai, S. Bai, S. Yang, S. Wang, S. Tan, P. Wang, J. Lin, C. Zhou, and J. Zhou. “Qwen-vl: A frontier large vision-language model with versatile abilities”. In: *arXiv preprint arXiv:2308.12966* (2023).
- [11] H. Touvron, T. Lavigil, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. “Llama: Open and efficient foundation language models”. In: *arXiv preprint arXiv:2302.13971* (2023).
- [12] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, et al. “OpenVLA: An Open-Source Vision-Language-Action Model”. In: *arXiv preprint arXiv:2406.09246* (2024).
- [13] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, et al. “Rt-2: Vision-language-action models transfer web knowledge to robotic control”. In: *arXiv preprint arXiv:2307.15818* (2023).
- [14] M. Zawalski, W. Chen, K. Pertsch, O. Mees, C. Finn, and S. Levine. “Robotic control via embodied chain-of-thought reasoning”. In: *arXiv preprint arXiv:2407.08693* (2024).

- [15] D. Qu, H. Song, Q. Chen, Y. Yao, X. Ye, Y. Ding, Z. Wang, J. Gu, B. Zhao, D. Wang, et al. “SpatialVLA: Exploring Spatial Representations for Visual-Language-Action Model”. In: *arXiv preprint arXiv:2501.15830* (2025).
- [16] X. Li, P. Li, M. Liu, D. Wang, J. Liu, B. Kang, X. Ma, T. Kong, H. Zhang, and H. Liu. “Towards generalist robot policies: What matters in building vision-language-action models”. In: *arXiv preprint arXiv:2412.14058* (2024).
- [17] Q. Li, Y. Liang, Z. Wang, L. Luo, X. Chen, M. Liao, F. Wei, Y. Deng, S. Xu, Y. Zhang, et al. “Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation”. In: *arXiv preprint arXiv:2411.19650* (2024).
- [18] H. Li, S. Yang, Y. Chen, Y. Tian, X. Yang, X. Chen, H. Wang, T. Wang, F. Zhao, D. Lin, et al. “CronusVLA: Transferring Latent Motion Across Time for Multi-Frame Prediction in Manipulation”. In: *arXiv preprint arXiv:2506.19816* (2025).
- [19] H. Liu, C. Li, Q. Wu, and Y. J. Lee. “Visual instruction tuning”. In: *Advances in neural information processing systems* 36 (2023), pp. 34892–34916.
- [20] H. Liu, C. Li, Y. Li, B. Li, Y. Zhang, S. Shen, and Y. J. Lee. *Llava-next: Improved reasoning, ocr, and world knowledge*. 2024.
- [21] S. Karamcheti, S. Nair, A. Balakrishna, P. Liang, T. Kollar, and D. Sadigh. “Prismatic vlms: Investigating the design space of visually-conditioned language models”. In: *Forty-first International Conference on Machine Learning*. 2024.
- [22] L. Beyer, A. Steiner, A. S. Pinto, A. Kolesnikov, X. Wang, D. Salz, M. Neumann, I. Alabdulmohsin, M. Tschannen, E. Bugliarello, et al. “Paligemma: A versatile 3b vlm for transfer”. In: *arXiv preprint arXiv:2407.07726* (2024).
- [23] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, et al. “Flamingo: a visual language model for few-shot learning”. In: *Advances in neural information processing systems* 35 (2022), pp. 23716–23736.
- [24] Z. Peng, W. Wang, L. Dong, Y. Hao, S. Huang, S. Ma, and F. Wei. “Kosmos-2: Grounding multimodal large language models to the world”. In: *arXiv preprint arXiv:2306.14824* (2023).
- [25] O. X.-E. Collaboration et al. *Open X-Embodiment: Robotic Learning Datasets and RT-X Models*. <https://arxiv.org/abs/2310.08864>. 2023.
- [26] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, et al. “Droid: A large-scale in-the-wild robot manipulation dataset”. In: *arXiv preprint arXiv:2403.12945* (2024).
- [27] J. L. McClelland, B. L. McNaughton, and R. C. O'Reilly. “Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory.” In: *Psychological review* 102.3 (1995), p. 419.
- [28] R. M. French. “Catastrophic forgetting in connectionist networks”. In: *Trends in cognitive sciences* 3.4 (1999), pp. 128–135.
- [29] L. X. Shi, B. Ichter, M. Equi, L. Ke, K. Pertsch, Q. Vuong, J. Tanner, A. Walling, H. Wang, N. Fusai, et al. “Hi robot: Open-ended instruction following with hierarchical vision-language-action models”. In: *arXiv preprint arXiv:2502.19417* (2025).
- [30] Z. Zhou, Y. Zhu, M. Zhu, J. Wen, N. Liu, Z. Xu, W. Meng, R. Cheng, Y. Peng, C. Shen, et al. “Chatvla: Unified multimodal understanding and robot control with vision-language-action model”. In: *arXiv preprint arXiv:2502.14420* (2025).
- [31] J. Yang, R. Tan, Q. Wu, R. Zheng, B. Peng, Y. Liang, Y. Gu, M. Cai, S. Ye, J. Jang, et al. “Magma: A foundation model for multimodal ai agents”. In: *arXiv preprint arXiv:2502.13130* (2025).
- [32] Q. Sun, P. Hong, T. D. Pala, V. Toh, U. Tan, D. Ghosal, S. Poria, et al. “Emma-X: An Embodied Multimodal Action Model with Grounded Chain of Thought and Look-ahead Spatial Reasoning”. In: *arXiv preprint arXiv:2412.11974* (2024).
- [33] F. Ebert, Y. Yang, K. Schmeckpeper, B. Bucher, G. Georgakis, K. Daniilidis, C. Finn, and S. Levine. “Bridge data: Boosting generalization of robotic skills with cross-domain datasets”. In: *arXiv preprint arXiv:2109.13396* (2021).
- [34] A. Brohan, N. Brown, J. Carbalal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, et al. “Rt-1: Robotics transformer for real-world control at scale”. In: *arXiv preprint arXiv:2212.06817* (2022).

- [35] G. H. Chen, S. Chen, R. Zhang, J. Chen, X. Wu, Z. Zhang, Z. Chen, J. Li, X. Wan, and B. Wang. “Allava: Harnessing gpt4v-synthesized data for lite vision-language models”. In: *arXiv preprint arXiv:2402.11684* (2024).
- [36] M. He, Y. Liu, B. Wu, J. Yuan, Y. Wang, T. Huang, and B. Zhao. “Efficient multimodal learning from data-centric perspective”. In: *arXiv preprint arXiv:2402.11530* (2024).
- [37] X. Li, K. Hsu, J. Gu, K. Pertsch, O. Mees, H. R. Walké, C. Fu, I. Lunawat, I. Sieh, S. Kirmani, et al. “Evaluating Real-World Robot Manipulation Policies in Simulation”. In: *arXiv preprint arXiv:2405.05941* (2024).
- [38] L. Wang, X. Chen, J. Zhao, and K. He. “Scaling proprioceptive-visual learning with heterogeneous pre-trained transformers”. In: *Advances in Neural Information Processing Systems 37* (2024), pp. 124420–124450.
- [39] J. Zheng, J. Li, D. Liu, Y. Zheng, Z. Wang, Z. Ou, Y. Liu, J. Liu, Y.-Q. Zhang, and X. Zhan. *Universal Actions for Enhanced Embodied Foundation Models*. 2025. arXiv: 2501 . 10105 [cs.RO]. URL: <https://arxiv.org/abs/2501.10105>.
- [40] L. Wang, J. Zhao, Y. Du, E. H. Adelson, and R. Tedrake. “Poco: Policy composition from and for heterogeneous robot learning”. In: *arXiv preprint arXiv:2402.02511* (2024).
- [41] Octo Model Team, D. Ghosh, H. Walké, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, C. Xu, J. Luo, T. Kreiman, Y. Tan, P. Sanketi, Q. Vuong, T. Xiao, D. Sadigh, C. Finn, and S. Levine. “Octo: An Open-Source Generalist Robot Policy”. In: *Proceedings of Robotics: Science and Systems*. Delft, Netherlands, 2024.
- [42] S. Ye, J. Jang, B. Jeon, S. Joo, J. Yang, B. Peng, A. Mandlekar, R. Tan, Y.-W. Chao, B. Y. Lin, et al. “Latent action pretraining from videos”. In: *arXiv preprint arXiv:2410.11758* (2024).
- [43] Y. Tian, S. Yang, J. Zeng, P. Wang, D. Lin, H. Dong, and J. Pang. “Predictive inverse dynamics models are scalable learners for robotic manipulation”. In: *arXiv preprint arXiv:2412.15109* (2024).
- [44] Y. Chen, Y. Ge, Y. Li, Y. Ge, M. Ding, Y. Shan, and X. Liu. “Moto: Latent motion token as the bridging language for robot manipulation”. In: *arXiv preprint arXiv:2412.04445* (2024).
- [45] J. Gu, S. Kirmani, P. Wohlhart, Y. Lu, M. G. Arenas, K. Rao, W. Yu, C. Fu, K. Gopalakrishnan, Z. Xu, et al. “Rt-trajectory: Robotic task generalization via hindsight trajectory sketches”. In: *arXiv preprint arXiv:2311.01977* (2023).
- [46] S. Deng, M. Yan, S. Wei, H. Ma, Y. Yang, J. Chen, Z. Zhang, T. Yang, X. Zhang, H. Cui, et al. “Graspvla: a grasping foundation model pre-trained on billion-scale synthetic action data”. In: *arXiv preprint arXiv:2505.03233* (2025).
- [47] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, et al. “\pi_0: A Vision-Language-Action Flow Model for General Robot Control”. In: *arXiv preprint arXiv:2410.24164* (2024).
- [48] J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. Fan, Y. Fang, D. Fox, F. Hu, S. Huang, et al. “GR00T N1: An Open Foundation Model for Generalist Humanoid Robots”. In: *arXiv preprint arXiv:2503.14734* (2025).
- [49] K. Pertsch, K. Stachowicz, B. Ichter, D. Driess, S. Nair, Q. Vuong, O. Mees, C. Finn, and S. Levine. “Fast: Efficient action tokenization for vision-language-action models”. In: *arXiv preprint arXiv:2501.09747* (2025).
- [50] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le. “Flow matching for generative modeling”. In: *arXiv preprint arXiv:2210.02747* (2022).
- [51] W. Peebles and S. Xie. “Scalable diffusion models with transformers”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2023, pp. 4195–4205.
- [52] Q. Bu, H. Li, L. Chen, J. Cai, J. Zeng, H. Cui, M. Yao, and Y. Qiao. “Towards Synergistic, Generalized, and Efficient Dual-System for Robotic Manipulation”. In: *arXiv preprint arXiv:2410.08001* (2024).
- [53] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, et al. “Do as i can, not as i say: Grounding language in robotic affordances”. In: *arXiv preprint arXiv:2204.01691* (2022).
- [54] S. Belkhale, T. Ding, T. Xiao, P. Sermanet, Q. Vuong, J. Tompson, Y. Chebotar, D. Dwibedi, and D. Sadigh. “Rt-h: Action hierarchies using language”. In: *arXiv preprint arXiv:2403.01823* (2024).

- [55] L. Smith, A. Irpan, M. G. Arenas, S. Kirmani, D. Kalashnikov, D. Shah, and T. Xiao. “STEER: Flexible Robotic Manipulation via Dense Language Grounding”. In: *arXiv preprint arXiv:2411.03409* (2024).
- [56] H. Huang, X. Chen, Y. Chen, H. Li, X. Han, Z. Wang, T. Wang, J. Pang, and Z. Zhao. “RoboGround: Robotic Manipulation with Grounded Vision-Language Priors”. In: *Proceedings of the Computer Vision and Pattern Recognition Conference*. 2025, pp. 22540–22550.
- [57] Y. Shentu, P. Wu, A. Rajeswaran, and P. Abbeel. “From LLMs to Actions: latent codes as bridges in hierarchical robot control”. In: *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2024, pp. 8539–8546.
- [58] F. AI. *Helix*. 2024. URL: <https://www.figure.ai/news/helix>.
- [59] Z. Li, G. Chen, S. Liu, S. Wang, V. VS, Y. Ji, S. Lan, H. Zhang, Y. Zhao, S. Radhakrishnan, et al. “Eagle 2: Building Post-Training Data Strategies from Scratch for Frontier Vision-Language Models”. In: *arXiv preprint arXiv:2501.14818* (2025).
- [60] Y. Zhou, T. Lei, H. Liu, N. Du, Y. Huang, V. Zhao, A. M. Dai, Q. V. Le, J. Laudon, et al. “Mixture-of-experts with expert choice routing”. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 7103–7114.
- [61] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, et al. “Lora: Low-rank adaptation of large language models.” In: *ICLR* 1.2 (2022), p. 3.
- [62] M. B. E.L. Buehler. “X-LoRA: Mixture of Low-Rank Adapter Experts, a Flexible Framework for Large Language Models with Applications in Protein Mechanics and Design”. In: (2024). URL: <https://arxiv.org/abs/2402.07148>.
- [63] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville. “Film: Visual reasoning with a general conditioning layer”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 32. 1. 2018.
- [64] OpenAI. “Gpt-4 technical report”. In: *arXiv:2303.08774* (2023).
- [65] H. Duan, J. Yang, Y. Qiao, X. Fang, L. Chen, Y. Liu, X. Dong, Y. Zang, P. Zhang, J. Wang, et al. “Vlmevalkit: An open-source toolkit for evaluating large multi-modality models”. In: *Proceedings of the 32nd ACM International Conference on Multimedia*. 2024, pp. 11198–11201.
- [66] X. Yue, Y. Ni, K. Zhang, T. Zheng, R. Liu, G. Zhang, S. Stevens, D. Jiang, W. Ren, Y. Sun, C. Wei, B. Yu, R. Yuan, R. Sun, M. Yin, B. Zheng, Z. Yang, Y. Liu, W. Huang, H. Sun, Y. Su, and W. Chen. “MMMU: A Massive Multi-discipline Multimodal Understanding and Reasoning Benchmark for Expert AGI”. In: *Proceedings of CVPR*. 2024.
- [67] L. Chen, J. Li, X. Dong, P. Zhang, Y. Zang, Z. Chen, H. Duan, J. Wang, Y. Qiao, D. Lin, et al. “Are We on the Right Way for Evaluating Large Vision-Language Models?” In: *arXiv preprint arXiv:2403.20330* (2024).
- [68] C. Fu, P. Chen, Y. Shen, Y. Qin, M. Zhang, X. Lin, J. Yang, X. Zheng, K. Li, X. Sun, Y. Wu, and R. Ji. *MME: A Comprehensive Evaluation Benchmark for Multimodal Large Language Models*. 2024. arXiv: 2306.13394 [cs.CV]. URL: <https://arxiv.org/abs/2306.13394>.
- [69] Y. Liu, Z. Li, M. Huang, B. Yang, W. Yu, C. Li, X.-C. Yin, C.-L. Liu, L. Jin, and X. Bai. “OCRBench: on the hidden mystery of OCR in large multimodal models”. In: *Science China Information Sciences* 67.12 (Dec. 2024). ISSN: 1869-1919. DOI: 10.1007/s11432-024-4235-6. URL: <http://dx.doi.org/10.1007/s11432-024-4235-6>.
- [70] T. Guan, F. Liu, X. Wu, R. Xian, Z. Li, X. Liu, X. Wang, L. Chen, F. Huang, Y. Yacoob, et al. “Hallusionbench: an advanced diagnostic suite for entangled language hallucination and visual illusion in large vision-language models”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 14375–14385.
- [71] Y. Liu, H. Duan, Y. Zhang, B. Li, S. Zhang, W. Zhao, Y. Yuan, J. Wang, C. He, Z. Liu, et al. “Mmbench: Is your multi-modal model an all-around player?” In: *European conference on computer vision*. Springer. 2024, pp. 216–233.
- [72] A. Singh, V. Natarjan, M. Shah, Y. Jiang, X. Chen, D. Parikh, and M. Rohrbach. “Towards VQA Models That Can Read”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 8317–8326.
- [73] M. Matheus, D. Karatzas, and C. Jawahar. “Docvqa: A dataset for vqa on document images”. In: *Proceedings of the IEEE/CVF winter conference on applications of computer vision*. 2021, pp. 2200–2209.

- [74] M. Mathew, V. Bagal, R. Tito, D. Karatzas, E. Valveny, and C. Jawahar. “Infographiccvqa”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2022, pp. 1697–1706.
- [75] A. Kembhavi, M. Salvato, E. Kolve, M. Seo, H. Hajishirzi, and A. Farhadi. “A diagram is worth a dozen images”. In: *European conference on computer vision*. Springer. 2016, pp. 235–251.
- [76] A. Masry, D. X. Long, J. Q. Tan, S. Joty, and E. Hoque. “Chartqa: A benchmark for question answering about charts with visual and logical reasoning”. In: *arXiv preprint arXiv:2203.10244* (2022).
- [77] R. Team. *RealWorldQA*. 2024. URL: <https://x.ai/news/grok-1.5v>.
- [78] B. Li, Y. Zhang, D. Guo, R. Zhang, F. Li, H. Zhang, K. Zhang, P. Zhang, Y. Li, Z. Liu, et al. “Llava-onevision: Easy visual task transfer”. In: *arXiv preprint arXiv:2408.03326* (2024).
- [79] P. Wang, S. Bai, S. Tan, S. Wang, Z. Fan, J. Bai, K. Chen, X. Liu, J. Wang, W. Ge, et al. “Qwen2-vl: Enhancing vision-language model’s perception of the world at any resolution”. In: *arXiv preprint arXiv:2409.12191* (2024).
- [80] R. Zheng, Y. Liang, S. Huang, J. Gao, H. Daumé III, A. Kolobov, F. Huang, and J. Yang. “Tracevla: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies”. In: *arXiv preprint arXiv:2412.10345* (2024).
- [81] J. Yang, R. Tan, Q. Wu, R. Zheng, B. Peng, Y. Liang, Y. Gu, M. Cai, S. Ye, J. Jang, et al. “Magma: A foundation model for multimodal ai agents”. In: *arXiv preprint arXiv:2502.13130* (2025).
- [82] N. Reimers. “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks”. In: *arXiv preprint arXiv:1908.10084* (2019).
- [83] T. Gao, X. Yao, and D. Chen. “Simcse: Simple contrastive learning of sentence embeddings”. In: *arXiv preprint arXiv:2104.08821* (2021).
- [84] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. “Bleu: a method for automatic evaluation of machine translation”. In: *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 2002, pp. 311–318.
- [85] J. C. Young, R. Arthur, and H. T. Williams. “CIDER: Context sensitive sentiment analysis for short-form text”. In: *arXiv preprint arXiv:2307.07864* (2023).
- [86] S. Banerjee and A. Lavie. “METEOR: An automatic metric for MT evaluation with improved correlation with human judgments”. In: *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*. 2005, pp. 65–72.
- [87] J. Liu, M. Liu, Z. Wang, L. Lee, K. Zhou, P. An, S. Yang, R. Zhang, Y. Guo, and S. Zhang. “Robomamba: Multimodal state space model for efficient robot reasoning and manipulation”. In: *arXiv e-prints* (2024), arXiv–2406.
- [88] P. Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, et al. “ π 0. 5: a vision-language-action model with open-world generalization, 2025”. In: URL <https://arxiv.org/abs/2504.16054> 1.2 () , p. 3.
- [89] N. Gao, Y. Chen, S. Yang, X. Chen, Y. Tian, H. Li, H. Huang, H. Wang, T. Wang, and J. Pang. “GENMANIP: LLM-driven Simulation for Generalizable Instruction-Following Manipulation”. In: *Proceedings of the Computer Vision and Pattern Recognition Conference*. 2025, pp. 12187–12198.
- [90] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. “Learning fine-grained bimanual manipulation with low-cost hardware”. In: *arXiv preprint arXiv:2304.13705* (2023).

Appendix

Contents

A More Experiments and Analysis	17
A.1 Vision-Language Evaluation	17
A.2 Data Ablation on OpenVLA	18
A.3 Further discussions	19
B Case Study	22
B.1 Reasoning Cases in SimplerEnv-Instruct	22
B.2 Failure Cases	23
B.3 GPT4o as the Auxiliary System 2	24
C Data Annotation Details and Analysis	25
C.1 Task Diversity Analysis	25
C.2 Prompting	25
C.3 Ground Truth Instruction for Data annotation	27
C.4 Language Motion Examples	28
D Benchmark Visualization	30
E Model Design and Training Details	32
E.1 Instruction Format	32
E.2 Learning Objective and Inference Procedure	33
E.3 Model Parameters	33
E.4 Inference Speed	34
E.5 Experiments Compute Resources	34
F Multimodal Examples	35
G Real-world Experiments Setup and Analysis	36
H Broader Impacts and Future Work	38
H.1 Broader Impacts	38
H.2 Future Work	38

The supplementary material is organized as follows:

- Appendix A presents: (1) additional benchmarks on language responses, (2) finetuning of OpenVLA under the same settings as InstructVLA, and (3) extended analysis of InstructVLA.
- Appendix B provides additional case analysis for InstructVLA, OpenVLA, and GPT-4o System2.
- Appendix C lists data annotation details, including GPT-4o prompt and dataset statistics. We further analyse the distribution of the instructions from two dimensions: task diversity and language diversity.
- Appendix D visualizes the SimplerEnv-Instruct benchmark and the acknowledgements of 3D assets.
- Appendix E details the model architecture, training configurations, inference speeds under different settings, and compute resources used.
- Appendix F shows several multimodal question answering examples.
- Appendix G describes the real-world experimental setup and provides example executions.
- Appendix H discusses the broader impacts and outlines future directions for InstructVLA.

A More Experiments and Analysis

A.1 Vision-Language Evaluation

Table 3: **VLA-IT captioning evaluation.** ‘‘Sentence-BERT’’ and ‘‘SimCSE’’ represent learning-based evaluation methods, while the remaining metrics are traditional n-gram-based evaluations focused on word distribution.

Methods	# Params	Sentence-BERT	SimCSE	BLEU-1	BLEU-4	METEOR	CIDER
Qwen2-VL [79]	2B	61.3	67.5	16.8	1.5	12.4	0.30
GPT4o [64]	-	60.7	67.1	16.3	1.8	16.2	0.09
OpenVLA(VLA-IT) [12]	7B	0.0	0.0	0.0	0.0	0.0	0.00
Magma [31]	8B	59.8	66.7	12.4	1.2	12.3	0.12
InstructVLA(Generalist)	2B	72.0	77.0	44.3	8.2	18.7	0.84

Table 4: **VLA-IT question-answering evaluation.**

Methods	# Params	Sentence-BERT	SimCSE	BLEU-1	BLEU-4	METEOR	CIDER
Qwen2-VL [79]	2B	51.9	53.4	15.3	2.8	17.9	0.82
GPT4o [64]	-	63.6	63.6	29.6	19.9	9.8	1.16
OpenVLA(VLA-IT) [12]	7B	0.0	0.0	0.0	0.0	0.0	0.00
Magma [31]	8B	53.5	54.5	23.7	5.7	21.6	1.04
InstructVLA(Generalist)	2B	64.9	65.9	44.6	17.4	23.5	1.85

Table 5: **VLA-IT instruction response evaluation.** We use ‘‘context creation’’ annotations, as they present a more challenging and diverse set of instructions.

Methods	# Params	Sentence-BERT	SimCSE	BLEU-1	BLEU-4	METEOR	CIDER
Qwen2-VL [79]	2B	52.3	54.0	5.6	1.5	11.6	0.09
GPT4o [64]	-	52.8	54.1	17.8	4.2	20.6	1.02
OpenVLA(VLA-IT) [12]	7B	0.0	0.0	0.0	0.0	0.0	0.00
Magma [31]	8B	10.9	13.6	3.7	0.8	1.6	0.00
InstructVLA(Generalist)	2B	71.6	73.1	50.2	24.1	25.8	2.26

In addition to the multimodal and closed-loop evaluations presented in the main results, we conduct supplementary language evaluations on the proposed VLA-IT dataset. This evaluation uses manually verified VLA-IT annotations on the Bridge dataset [33], chosen for its diversity and distinct validation split. We generate 1,000 annotations following the method described in the VLA-IT dataset generation section. Two evaluation metrics are employed: (1) learning-based methods [82, 83], and (2) traditional metrics [84, 85, 86].

The captioning, question-answering and instruction-following results are presented in Tables 3 to 5. We select Qwen2-VL [79] and GPT-4o [64] as zero-shot VLM baselines, and include Magma [31] (zero-shot) and OpenVLA [12] fine-tuned on the VLA-IT dataset as baselines for VLA models.

Although OpenVLA is fine-tuned on the VLA-IT dataset, it fails to generate complete sentences under the same evaluation setting as InstructVLA, despite the performance on multiple-choice benchmarks reported in our main results. This suggests a significant loss of its free-form dialogue capability. Magma performs well on question answering and captioning tasks. However, it struggles with instruction response (Figure 7), often generating outputs misaligned with the given image. *We hypothesize that this failure stems from the similarity between these instructions and the atomic commands used in finetuning manipulation datasets, which disrupts the coherence of the language latent space near the action latent space.* This suggests a limited capacity to interpret and generalize free-form instructions, hindering effective transfer of vision-language capabilities.

InstructVLA achieves state-of-the-art performance, while GPT4o demonstrates competitive results. We visualize three episodes in Figure 8. GPT-4o generates more detailed captions but occasionally exhibits minor hallucinations. In the instruction response task, InstructVLA produces clearer and more grounded responses compared to GPT-4o, benefiting from the integration of ground truth atomic instructions during the data annotation process, as discussed in Appendix C.3.

	Which fruit is curved and yellow? Banana	QA
	Make sure the banana is accessible by the gripper for the next task. \u0e38\u0e29\u0e22\u093e\u0928\u0938 \u0e41\u0e38\u0fee4\u03bb\u03b1\u03bd \u6d59\u6c5f\u0e38\u0e29\u0e22	Instruction
	Check the contents of the brown box by opening its flap.	Instruction
	When you open the flap of the brown box, you will find a set of kitchen utensils inside. The utensils may include items such as a spatula, a ladle, a grater, and other tools designed for cooking or food preparation. These utensils are typically made of materials like stainless steel, silicone, or plastic, and they are often used together to perform specific tasks in the kitchen. The presence of these utensils in the box suggests that it is a set designed for easy access and organization of commonly used kitchen tools.	
	What color are the handles of the screwdriver? The handles of the screwdriver are red.	QA
	Organize the tools by picking up the rainbow Allen key set first. \u0446\u0456\u0439\u043d\u0430 diren\u65cf\u81ea\u6ccb diren\u8083 diren\u0e38\u0e29\u0e22	Instruction

Figure 7: **Magma results.** Magma’s responses collapse when given instructions resembling those in its manipulation tasks, possibly due to learned actions interfering with its language latent space.

	A laundry area with a white basket containing colorful clothes, positioned next to a washing machine. Please prepare the laundry for washing by putting it in the machine. I will place the clothes into the washing machine.	On the table, there is a washing machine with its door open. Inside the door, there is a visible drum. In front of the washing machine, there is a white basket containing various cloth items. The basket is filled with different colored fabrics, including blue, green, and red. A red cloth piece is draped over the edge of the basket.
	What container is used for holding the cloth pieces on the ground? The basket. A red toolbox filled with various tools, including screwdrivers and wrenches, with a focus on a red and yellow screwdriver. We need to clear the toolbox. Begin by picking up the colorful hex keys set. I will start by picking up the rainbow Allen key set.	Place all the clothes from the basket into the washing machine. The container used for holding the cloth pieces on the ground is typically a basket. The table is covered with various tools, including screwdrivers with different handles, hex keys, drill bits, and other assorted hand tools. The tools are organized in sections within a red drawer, suggesting it’s part of a tool chest or storage unit.
	What color are the handles of the screwdriver? The screwdriver handles are red and yellow. Before we cook, can you ensure the pot is spotless by using the sponge? I will wipe the pot with the sponge. What object is used for cleaning in the scene? The yellow sponge.	Pick up the rainbow Allen key set and arrange it neatly on one side of the drawer, then organize the remaining tools by type and size. The handles of the screwdrivers are yellow with red accents. On the table, there is a metal pot placed upside down, a yellow sponge with a green scrubbing side, a red-capped spice container, a waffle-textured ice cream cone, and a mounted metal clamp extending over the edge of the table. The surface is a wooden countertop, and the background has dark purple and white tiled walls.

(a) InstructVLA

(b) GPT4o

Figure 8: **Comparison with GPT-4o.** We visualize three examples from the VLA-IT language validation set. Each example includes a scenario caption (top), instruction response (middle), and question answering (bottom). The GPT-4o column displays **responses only**, as the instructions are identical across models.

A.2 Data Ablation on OpenVLA

Table 6: **Data ablation on OpenVLA.** “+VL” indicates finetuning OpenVLA with the same multimodal dataset used by InstructVLA. “+VLA-IT” refers to finetuning OpenVLA with the same VLA-IT dataset as InstructVLA. “+GPT4o” denotes using GPT4o as system 2 to translate free-form instructions into atomic ones.

	OpenVLA (OXE)	OpenVLA + VL	OpenVLA + VL + VLA-IT	OpenVLA + VL + GPT4o	InstructVLA
Instruction Aggregation	14.8	28.3	30.5	38.8	43.3
Situated Reasoning	13.6	19.5	17.4	32.4	48.8
Average	14.2	23.9	24.0	35.6	46.0

To investigate whether the performance gain of VLA-IT arises solely from the dataset itself, we reimplement the training procedure of the InstructVLA on OpenVLA [12], which represents a class of models trained under the action-only paradigm. As shown in Table 6, OpenVLA benefits from both vision-language and VLA instruction tuning data, with the latter showing greater improvement in the instruction aggregation setting. This is attributed to exposure to more diverse instructions.

However, performance on the situated reasoning setting remains unchanged, likely due to catastrophic forgetting caused by the action-only training paradigm, which limits OpenVLA’s ability to leverage the VLM’s reasoning ability through simple finetuning.

The greatest performance gain is observed when GPT-4o is introduced as an auxiliary System 2 in both evaluation settings. However, overall performance remains inferior to InstructVLA, as GPT-4o cannot fully ground free-form instructions to the atomic skills on which OpenVLA is pretrained.

A.3 Further discussions

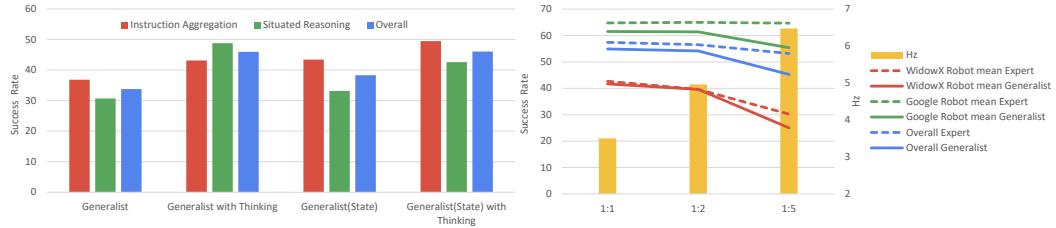


Figure 9: Test-time thinking and dual-frequency evaluation. “Expert” refers to the model after action pretraining, while “Generalist” denotes the model after VLA-IT tuning. For dual-frequency evaluation, the horizontal axis represents the ratio of VLM executions to expert model executions.

Role of VLA-IT training. As shown in Table 2, although the InstructVLA-expert model does not outperform the OpenVLA(OXE) on Situated Reasoning of SimplerEnv-Instruct, which benefits from direct full fine-tuning of the VLM backbone, InstructVLA-expert shows promising scaling ability in understanding complex instructions and performing test-time thinking after stage-2 VLA-IT training. This result reflects a deliberate design choice in InstructVLA, where latent action learning during pretraining focuses on querying from visual and simple instruction features rather than relying on the full semantic space of the VLM too early. This design offers two significant advantages. First, it preserves the original semantic space of the pretrained VLM, maintaining its vision-language capabilities. Second, it enables the model to integrate diverse reasoning contexts during VLA-IT training. These properties contribute to the strong performance gains achieved by our generalist model and demonstrate the effectiveness of this training paradigm.

Test-time thinking. Allowing the model to perform test-time thinking by generating textual analysis of the given instruction can improve performance, particularly on situated reasoning tasks, as shown in Figure 9 (left). Notably, while the model with access to robot state outperforms the one without state when no instruction response is required, it provides limited performance gains when instruction

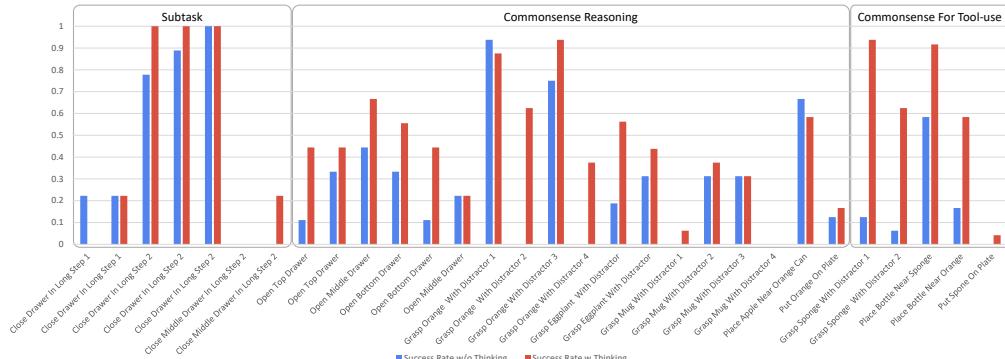


Figure 10: Performance visualization of 30 situated reasoning tasks with and without reasoning enabled. Activating reasoning in our generalist model generally improves performance. For clarity, tasks are grouped into three categories: *Subtask*, involving subtask identification; *Commonsense Reasoning*, requiring broad world knowledge; and *Commonsense for Tool Use*, focusing on tool-related reasoning.

following is involved. We hypothesize that state information helps the model retain manipulation skills but compromises its generalization to OOD environments and instructions.

Dual frequency inference. To further analyze the relationship between latent actions generated by the VLM and the final decoded actions, we decouple the inference frequencies of the VLM and the action expert, as illustrated in Figure 9 right. The results show that performance remains stable at a 1:2 ratio (VLM:expert), but begins to degrade at higher ratios. This suggests that latent actions offer relatively stable guidance to the action expert, reducing the need for frequent VLM queries.

A closer look at reasoning in manipulation tasks. We compare the performance of the generalist model on SimplerEnv-Instruct with and without vision language reasoning, as shown in Figure 10. A clear performance gap emerges in tasks involving commonsense tool use and interaction with articulated objects. This may result from instructions that do not explicitly state the intended actions and objects. For example, retrieving a cleaning tool from a drawer requires the robot to infer whether the prerequisite of an open drawer is satisfied, and to identify the sponge as the appropriate tool among several options. In addition to these cases, the reasoning process also improves performance on other situated reasoning tasks by grounding unfamiliar instructions using the pretrained in-domain knowledge of the vision language model.

VLA instruction tuning for cross-embodiment understanding. To assess whether InstructVLA retains this capability, we evaluate three variants on SimplerEnv-Instruct (see Table 7): InstructVLA-Expert, trained solely on atomic instructions without test-time thinking; InstructVLA Generalist (Bridge), trained with the VLA-IT dataset on Bridge and the original Fractal dataset; and InstructVLA Generalist, trained with the full VLA-IT datasets across both environments. Adding the Bridge dataset results in a 139.4% improvement in Situated Reasoning performance for Generalist (Bridge) over the expert baseline, while Instruction Aggregation performance remains comparable. This discrepancy reflects differing generalization requirements: Instruction Aggregation emphasizes linguistic robustness, whereas Situated Reasoning demands vision-language grounding prior to action. The latter particularly benefits from the preserved reasoning capabilities of the pretrained VLM. As illustrated in Figure 11, the zero-shot model generates more diverse and accurate outputs than its fine-tuned counterpart.

Table 7: **Instruction tuning data ablation.** We evaluate three settings: without VLA-IT data, with data only on Bridge, and with VLA-IT data on both Fractal and Bridge. This ablation examines the contribution of the VLA-IT dataset and the cross-embodiment generalization of InstructVLA on SimplerEnv-Instruct.

Instruction Tuning Data		Name	Insturction Aggregation	Situated Reasoning	Overall
Bridge	Fractal				
✗	✗	Expert	20.8	10.4	15.6
✓	✗	Generalist (Bridge)	18.4	24.9	21.7
✓	✓	Generalist	43.3	48.8	46.0

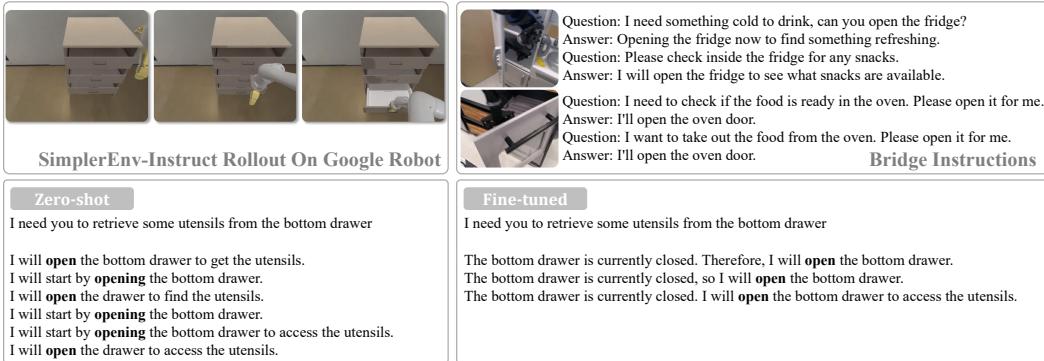


Figure 11: **Case study on cross-embodiment.** Top left: rollouts on SimplerEnv-Instruct. Top right: similar scenarios from the Bridge dataset with corresponding instructions. Bottom left: zero-shot results trained only on Bridge instructions. Bottom right: rollouts from the fine-tuned model.

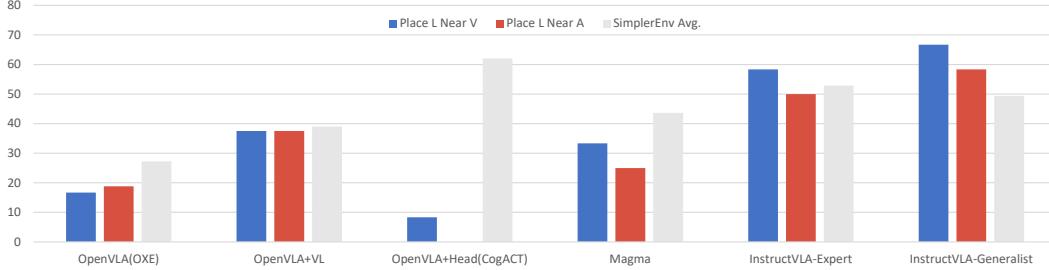


Figure 12: Case study on multimodal capabilities. OCR represents a unique multimodal skill of VLMs that is absent from typical manipulation datasets. We evaluate two tasks from the Instruction Aggregation set in SimplerEnv-Instruct, involving moving one letter to another (see Figure 13(1)). By comparing different finetuning paradigms, we assess how effectively multimodal capabilities are integrated into VLA models.

Case study on multimodal capability transfer. As shown in Figure 12, we compare InstructVLA with OpenVLA [12], Magma [31], and CogACT [17], all using the same input (language instruction and a single image). InstructVLA-Expert, though trained without multimodal datasets, retains the OCR capability of the underlying VLM and achieves the best performance among baselines trained solely on manipulation data. Finetuning InstructVLA-Expert into InstructVLA-Generalist with multimodal and VLA-IT datasets further enhances performance. For autoregressive models such as OpenVLA and Magma, multimodal finetuning improves OCR ability. In contrast, CogACT, when fine-tuned from OpenVLA(OXE) only on manipulation data with an action head, shows improved in-domain performance (on SimplerEnv) but suffers in generalization.

Training at scale. A generalist VLA model with vision-language capabilities should be scalable across both manipulation and multimodal datasets. In this context, we compare datasets used by models claiming generalist abilities, as shown in Table 8. RoboMamba [87] utilizes a limited manipulation dataset compared to other methods, while the dataset for ChatVLA [30] is not reported. $\pi_{0.5}$ [88] employs a significantly larger multimodal dataset than other approaches, though its multimodal performance is not disclosed. Magma uses more robot and multimodal data but achieves slightly worse performance on both multimodal and manipulation benchmarks compared to InstructVLA.

Table 8: Data comparison of different methods. “Trans.” denotes transitions.

	Magma[31]	ChatVLA[30]	RoboMamba[87]	$\pi_{0.5}$ [88]	InstructVLA
Manipulation Data	9.4M Trans.	-	10K Trans.	400 Hours	469 Hours/ 5.9M Trans.
Multimodal Data	1.2M Images + 4M Videos	54K	1.5M	>7M	2M

B Case Study

B.1 Reasoning Cases in SimplerEnv-Instruct

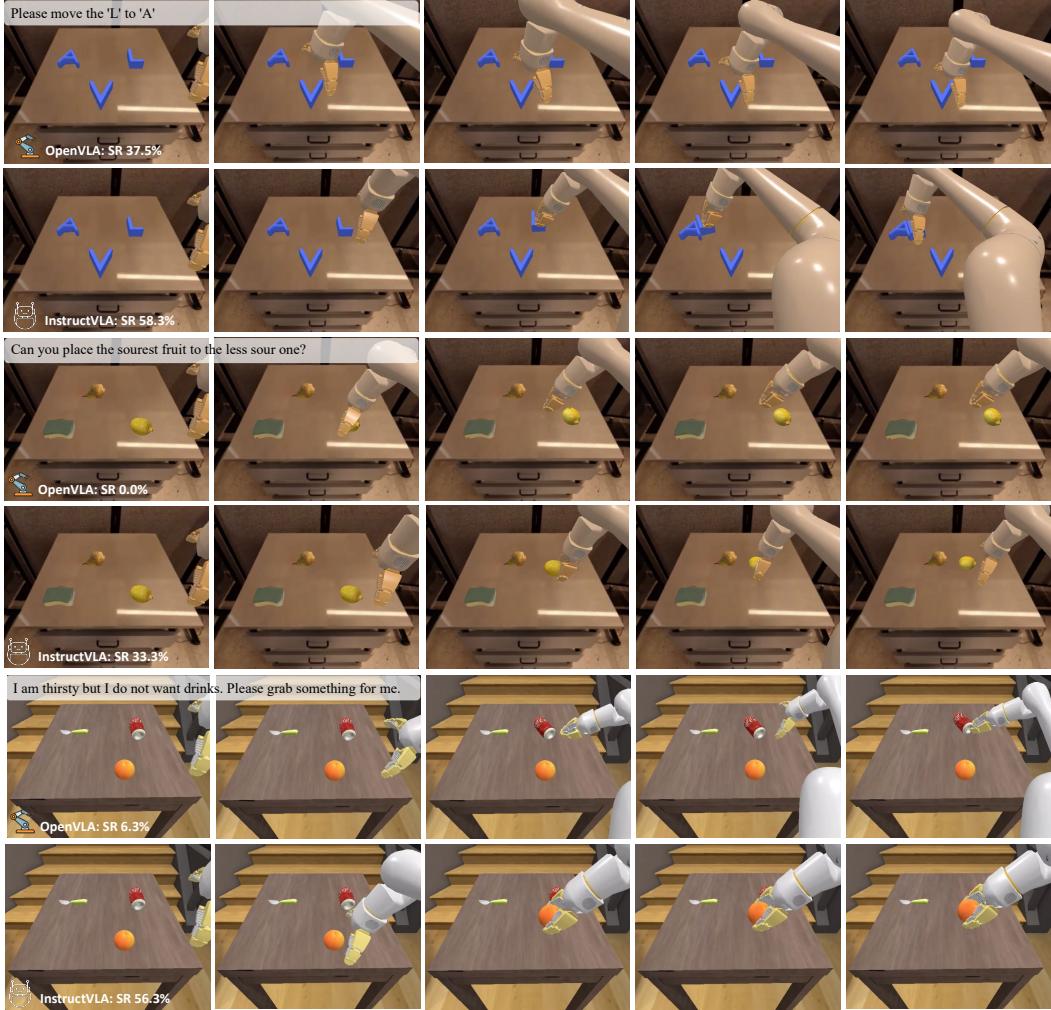


Figure 13: **Reasoning cases in SimplerEnv-Instruct.** Three cases of the VL fine-tuned OpenVLA and InstructVLA-Generalist. “SR” denotes success rate.

We present three representative reasoning cases in Figure 13. In the first example, OpenVLA fails to associate the letters “V” and “L” with their corresponding shapes in the image, resulting in consistent failure to grasp in all similar scenarios. In the second case, OpenVLA does not correctly associate the concept of “sour” with the corresponding fruit. As a result, its action is influenced by both the pear and lemon, leading to a grasp attempt between them that strikes the table. In the final example, OpenVLA fails to interpret the negation in the instruction and incorrectly grasps Coke instead of orange.

B.2 Failure Cases

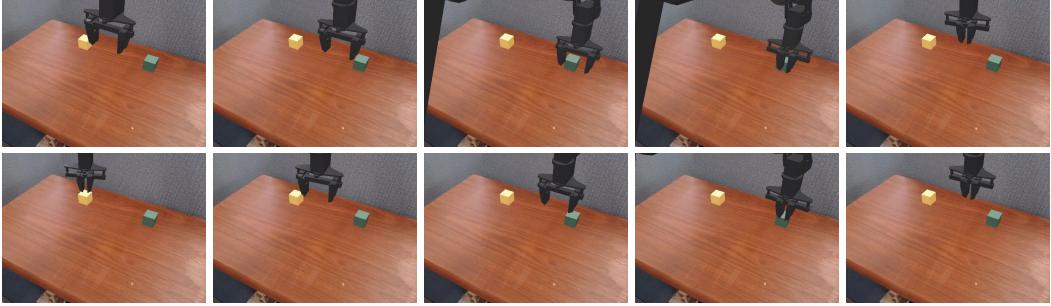


Figure 14: **Failure case 1 of InstructVLA.** The model receives only a third-person view image as visual input, making it difficult to estimate depth or the gripper’s relative position to the object. Consequently, it fails to grasp the object accurately, despite the gripper appearing aligned with the target in the image.

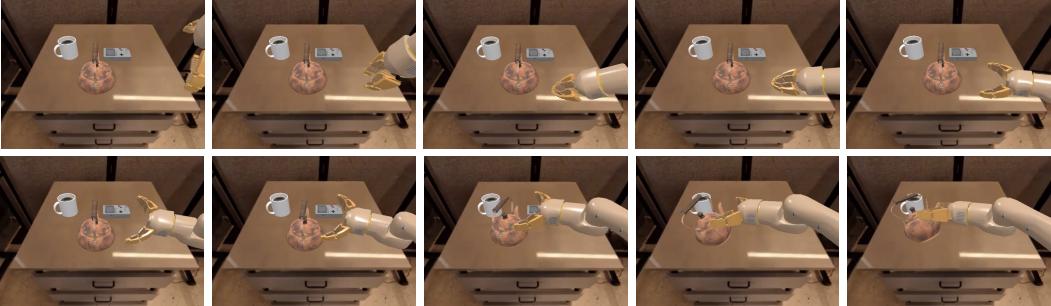


Figure 15: **Failure case 2 of InstructVLA.** The model fails to accurately estimate depth due to the real-to-sim gap, specifically the absence of arm reflection on the table, which causes the robot to become stuck in an out-of-distribution position.

We illustrate two representative failure cases of InstructVLA in Figures 14 and 15. While some failures may result from the real-to-sim gap, incorporating additional sensory inputs such as depth information and robot state may enhance performance. We leave this exploration for future work. Additionally, we observe that the model achieves higher success rates in language responses than in action execution, suggesting that multimodal understanding is more readily transferable than manipulation skills. This highlights a fundamental challenge in the development of embodied models.

B.3 GPT4o as the Auxiliary System 2

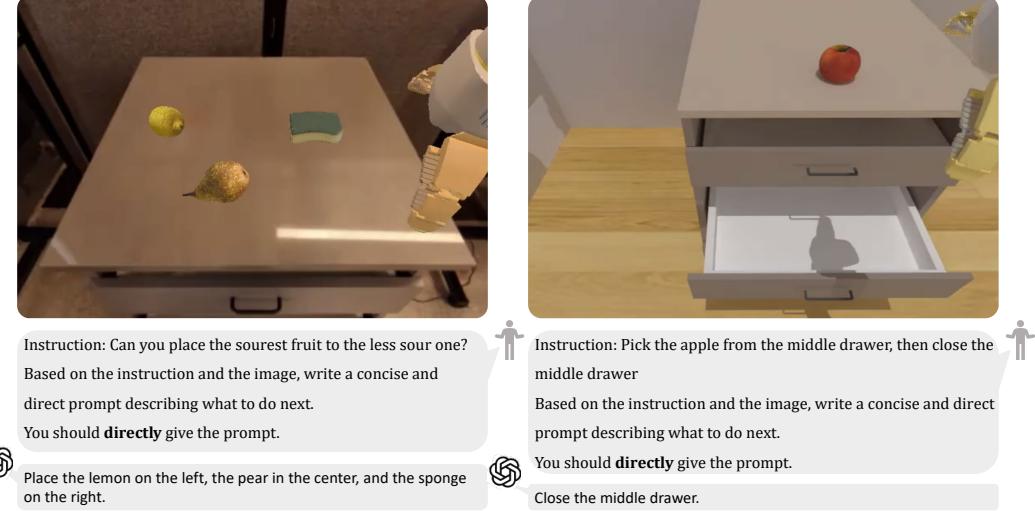
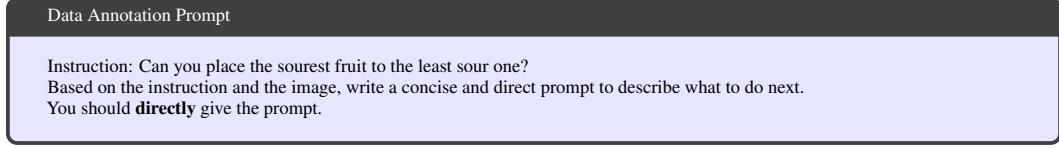


Figure 16: **GPT-4o as the auxiliary system 2.** We prompt GPT-4o with the first image from the environment along with the instruction, asking it to rewrite the prompt in a simple and clear format.

A strong baseline for InstructVLA integrates an expert model capable of executing atomic instructions with GPT-4o as an instruction parser to decompose complex, free-form commands for decision-making [29, 89]. The prompt used is listed in Prompt 1, and it was evaluated and refined on 20 test cases from the Instruction Aggregation to ensure reliable performance. Results on additional test cases are presented in Figure 16. GPT-4o successfully identified the atomic instruction in the second case but failed in the first.

During evaluation, GPT-4o is invoked only in the initial step to ensure an unobstructed view of the scene and to generate a free-form instruction. We do not provide a closed set of task-relevant instructions for selection, as the training set (Figure 17) lacks sufficient diversity in instructions and objects, and therefore does not adequately cover the evaluation settings. Across 80 evaluation cases, GPT-4o frequently fails in physical grounding, maintaining coherence, and accurately interpreting the scene.



C Data Annotation Details and Analysis

The data analysis and GPT4o prompt are listed as follows (Figure 17 and Prompt 2).

C.1 Task Diversity Analysis

We categorize tasks into two broad classes: **Command Rewriting / Context Creation** and **Question Answering**. Each class includes several common task types:

Command Rewriting / Context Creation

- **Complex Object Referencing:** Uses attributes, pronouns, or relational terms to reference an object.
Example: “Place the red item next to the box.”
- **Novel Action Referencing:** Rephrases a previously known action using a different verb or motion.
Example: “Shut the drawer” (instead of “Close the drawer”).
- **Negative Task Specification:** Specifies the correct action by negating incorrect alternatives.
Example: “I’m thirsty, but I don’t want sparkling water—bring me something else.”
- **Subtask Identification:** Isolates a step from a multi-step instruction with a clear sequential order.
Example: From “Take the spoon out of the top drawer,” execute only the first step.
- **Situated Task Identification:** Infers the required action based on contextual cues or situational conditions.
Example: “I want to clean the table. What should I use?”
- **Direct Instruction:** Provides an explicit and unambiguous command.
Example: “Organize the drinks by putting the green can next to the Coke can.”
- **Tool-Use Understanding:** Refers to an object by its utility or function rather than its name.
Example: “Hand me something to cut with” (instead of “Use the knife”).

Question Answering

- **Quantitative Identification:** Requires determining the number or quantity of items.
Example: “How many apples are on the table?”
- **Spatial Identification:** Involves spatial relationships between objects or with the user.
Example: “Is the cup on the left or the right of the plate?”
- **Visual Identification:** Focuses on appearance-based attributes such as color or shape.
Example: “Which one is the metallic silver object?”
- **Commonsense Answering:** Requires everyday reasoning or world knowledge.
Example: “Which of these would you use to cut paper?”
- **State Identification:** Determines the current condition or status of an object.
Example: “Is the drawer currently open or closed?”

The data examples for VIA-IT are provided in Figures 18 and 19.

C.2 Prompting

The Prompt 2, along with three images captured at the beginning, middle, and end of each episode, is packaged and sent to GPT-4o. Episodes from the Bridge dataset [33] that lack valid instructions are excluded from annotation.

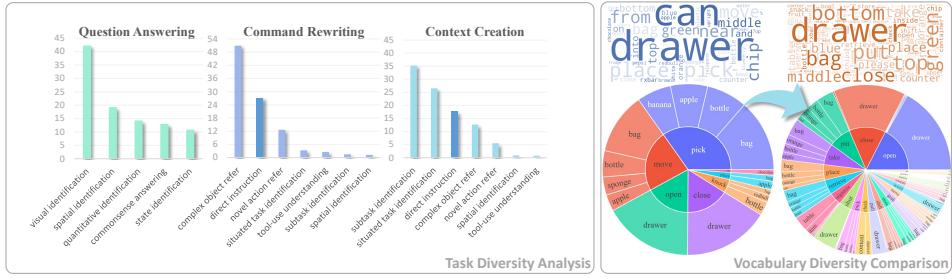


Figure 17: **Data analysis.** Left: We manually identify common task categories and calculate the distribution. The proportion of direct prompts is reduced in favor of more diverse, free-form expressions. Right: Word cloud and verb-noun analyses compare the original Fractal instructions with the VLA-IT corpus.

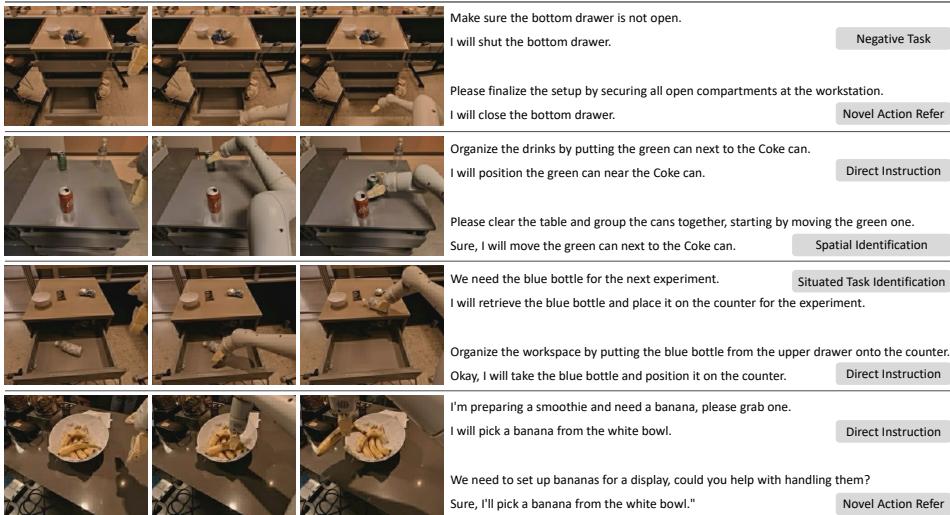


Figure 18: More VLA instructions on Fractal dataset.

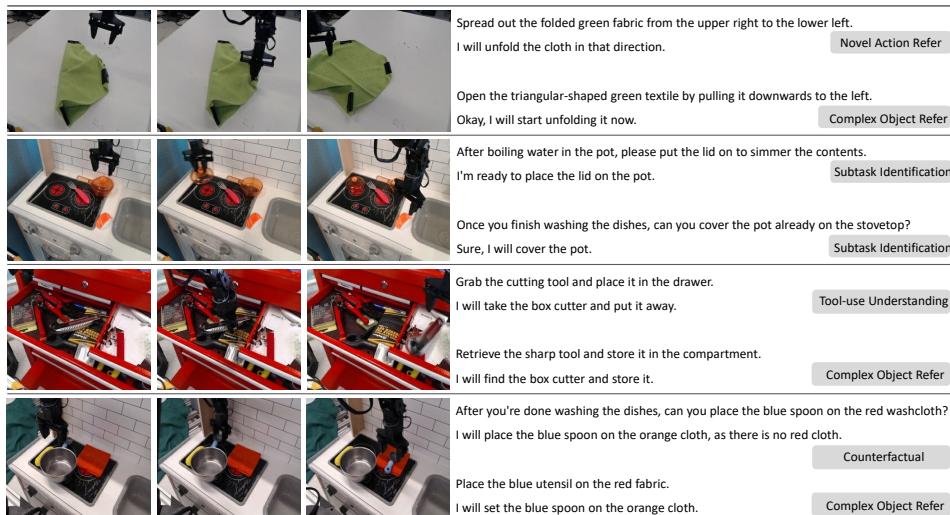


Figure 19: More VLA instructions on Bridge dataset.

Data Annotation Prompt

Imagine a robot assistant operating in a laboratory or household environment. The robot is expected to follow diverse commands based on realistic tasks and human interactions. Your task is to:

1. Write a caption to describe the visual scene shown in the **first image**. You should **NOT** include the robot itself here.
2. Based on the given robot task description and the images, generate new user instructions and corresponding robot responses with QA pairs.

The new user instructions should align with the actions performed by the robot in the images and with the environment shown in the images. You are required to produce three categories of instructions:

1. **Command Rewriting (CR)**: Rephrase the task description using diverse language styles and vocabulary. You may refer to objects by their utility, color, shape, or other attributes, but ensure the attribute you use is unique to each object.
2. **Context Creation (CC)**: Generate detailed scenarios where the robot needs to perform the given instruction. The situation should involve realistic surroundings or tasks where this instruction would be necessary. You may also simulate a long-horizon task based on the context provided by the image. Your generated question should **NOT** include the answer itself.
3. **Scene-related Commonsense QA (QA)**: Generate some other QA pairs that are related to the scene. The answer should be concise and consistent among the three images.

For each instruction, provide a concise robot response that clearly (use simple words) communicates the next action the robot will take. **Do not chain multiple actions together using phrases like "and then."** If necessary, the response may include a brief explanation of the reasoning. Avoid repeating the instruction in the response.

Response Format: You MUST respond in JSON format. You should include "Caption", "CR", "CC", and "QA" in your response. You should create 1-3 entries for each of CR, CC, and QA.

Example 1: For the instruction "Close middle drawer":

(Corresponding three images omitted)

Caption: "A table with a Coke and chips on top, with its middle drawer open."

```
{
  "Caption": "A table with a Coke and chips on top, with its middle drawer open.",
  "CR": [ { "question": "Push the middle drawer closed.", "answer": "Ok, I will close it." },
    { "question": "Ensure the center drawer is closed.", "answer": "I will close the drawer." },
  "CC": [ { "question": "I want you to take out the Coke from the middle drawer and closing it.", "answer": "The Coke is on the table, and the middle drawer is empty. So, I should close the middle drawer." },
    { "question": "Please push the middle drawer shut so we can clear the workspace.", "answer": "Okay, I will close the middle drawer." } ],
  "QA": [ { "question": "What is in the middle drawer?", "answer": "The middle drawer is empty." },
    { "question": "How many Coke cans are on the table?", "answer": "One." } ]
}
```

Example 2: For the instruction "move the apple near the Coke":

(Corresponding three images omitted)

Caption: "A table with Coke, apple, and soap on it."

```
{
  "Caption": "A table with Coke, apple, and soap on it.",
  "CR": [ { "question": "Move the healthy food near the Coke.", "answer": "The healthy food refers to the apple, and I will move the apple to the Coke." },
    { "question": "Move the apple to the cylindrical-shaped object.", "answer": "Of course!" } ],
  "CC": [ { "question": "Gather all objects near the Coke, except the soap.", "answer": "I will move the apple to the Coke." } ],
  "QA": [ { "question": "I'm thirsty, what can I have?", "answer": "The Coke is on the table." },
    { "question": "What is the healthy food on the table?", "answer": "The apple." } ]
}
```

Your task description is "<placeholder>".

Now give your response in JSON format.

C.3 Ground Truth Instruction for Data annotation

During data generation, we observe that GPT-4o often struggles to accurately interpret robot behavior using only the three provided images, performing noticeably worse than humans. To quantify this, we randomly sample 100 examples and prompt GPT-4o to generate our four types of annotations using a similar prompt (excluding the ground truth instruction from a human expert). We then manually evaluate the correctness of the results: a sample is scored as 1 if no obvious errors are found, 0.5 if minor errors are present, and 0 if completely incorrect.

The results are summarized in Tables 9 and 10, with two representative cases illustrated in Figures 21 and 22. In the first case, GPT-4o hallucinates the robotic arm as a bread roll, leading to an incorrect caption and instruction. In the second, it reverses the temporal order of actions, resulting in an inaccurate annotation.

We attribute this performance gap to GPT-4o’s lack of temporal grounding and the low visual quality of images in manipulation datasets. In contrast, human-provided instructions inherently encode temporal links across the image sequence by grounding the task in context, identifying target objects, and specifying corresponding robot actions. This finding underscores that, despite their impressive capabilities, even state-of-the-art VLMs lack embodied experience and temporal grounding, limiting their ability to infer fine-grained actions in robot manipulation tasks.

Table 9: **Data annotation success rate.** GPT-4o shows a significant performance drop without ground truth instructions during data annotation.

Method	Success Rate
With GT Instruction	95.4%
Without GT Instruction	45.0%

Table 10: **Distribution of common error types.** Error analysis of GPT-4o annotations generated without access to ground truth instructions, with long-tail errors omitted.

Error Type	Percentage
Ignore Vision Context	32.5%
Reverse Temporal Order	10.2%
Minor Object Hallucination	5.7%

C.4 Language Motion Examples

Language motion [54] describes end-effector movements using natural language, enhancing the VLM’s understanding of robotic manipulation. To generate such annotations, we leverage proprioceptive data that captures the end-effector’s position and orientation relative to the robot base. While the Bridge dataset [33] adopts annotations from ECoT [14], we additionally annotate the Fractal dataset [34] using a similar approach. The examples on the Fractal dataset are presented in Figure 20.



Figure 20: Language motion examples

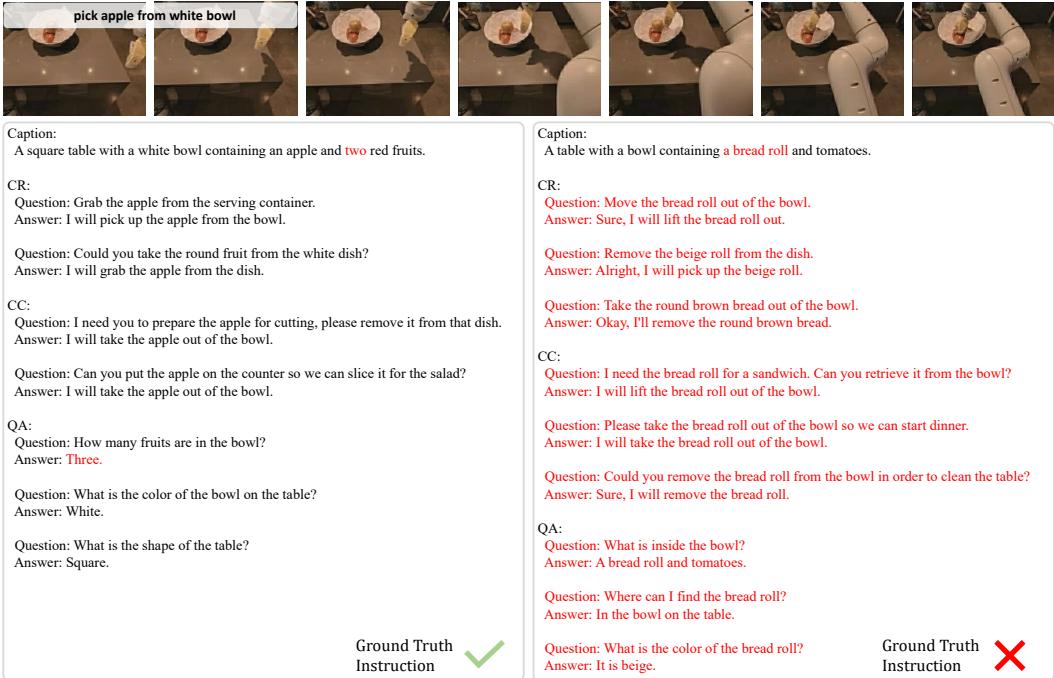


Figure 21: **Comparison of GPT annotations with and without ground truth instruction.** Errors are highlighted in red.

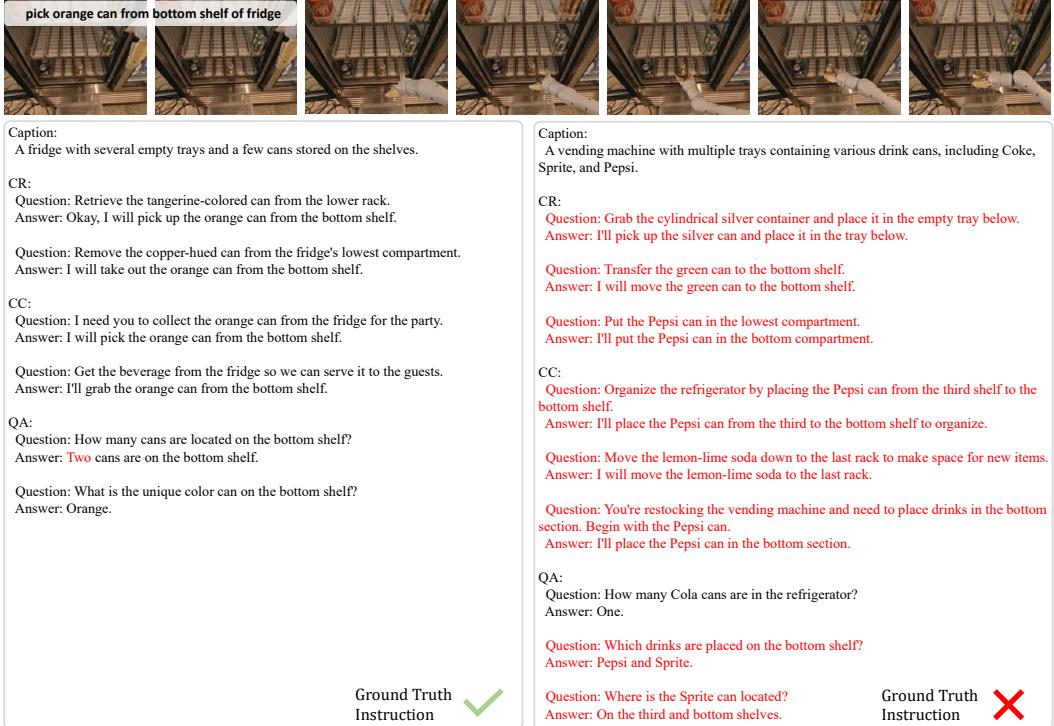


Figure 22: **Comparison of GPT annotations with and without ground truth instruction.** Errors are highlighted in red. In this case, GPT-4o incorrectly infers the temporal sequence of actions without access to the instruction.

D Benchmark Visualization

As shown in Table 11, although SimplerEnv-Instruct is primarily designed for instruction generalization, we incorporate diverse out-of-distribution objects, environments, and distractors to prevent VLA models from exploiting the benchmark by disregarding the instructions.

Table 11: Task distribution

Attr.	with OOD Obj.	with OOD Env.	with Distract Obj.	Only Language OOD
Percentage(%)	50.0	62.5	35.0	5.0

We select 10 task scenes with InstructVLA rollout actions and responses, as shown in Figures 23 and 24, to illustrate its performance on both Instruction Aggregation and Situated Reasoning tasks.

Rollout Video	Instruction	Model Response
	Please retrieve the green teardrop-shaped fruit from the surface. Veuillez récupérer le fruit en forme de goutte verte de la surface.	I will pick up the pear.
	"Pick up the yellow fruit on the table saisis la banane	I will pick up the banana.
	Pick up the small round purple fruit	I will pick up the grape.
	Grab the green and white fizzy drink can from the workspace. toma la lata verde y blanca	I will pick up the 7up can.
	Please move the 'L' to 'V'	I will move the 'L' next to the 'V'.
	pick the plastic bottle and then place it near the orange.	I will pick up the bottle and move it to the orange.
	Can you place the sourest fruit to the less sour one?	I will put the lemon next to the pear.
	pick the mug and then place it near the play station.	I will pick the mug and move it closer to the play station.
	Pull the center drawer open	I will open the middle drawer now.
	Pull the top drawer out place the apple into the drawer you just opened	I will open the top drawer now. I will open the top drawer and place the apple inside.

Figure 23: **Examples of Instruction Aggregation in SimplerEnv-Instruct.** We list ten examples with corresponding instructions and responses. Notably, InstructVLA shows the strong **zero-shot** ability to interpret multilingual instructions, recognize novel objects, and leverage OCR capabilities.

Acknowledgements of 3D assets. We gratefully acknowledge the creators of the following 3D assets used in *SimplerEnv-Instruct*. All assets are licensed under the Creative Commons Attribution license:

- *Fruit Bowl Collection*
<https://sketchfab.com/3d-models/fruit-bowl-collection-d89f6311cb0c4c37b0bf9cdd1e5abcb3>
- *Coffee Mug*
<https://sketchfab.com/3d-models/coffee-mug-school-project-5f5cce1514c440887c072fae8e0d699>
- *Copper Tea Pot*
<https://sketchfab.com/3d-models/copper-tea-pot-27f2ac58f7614f2796630bdc6f18ee2f>
- *Game Boy Classic*
<https://sketchfab.com/3d-models/game-boy-classic-0ae80019e6f046168923286d7e628f6f>

Rollout Video	Instruction	Model Response
	Please move the water bottle to wet the sponge.	I will pick up the water bottle to help you.
	Please check if there are any items in the bottom drawer	The bottom drawer is currently closed, so I will open it to check for items. The bottom drawer is already open, so I will check its contents.
	I want a health vegetable. Grab it for me.	I will pick up the eggplant.
	Can you grab the fruit before preparing the salad?	I will pick up the orange first.
	I want to clean the table. Pick a suitable tool for me.	I will select the sponge for cleaning.
	Open the top drawer and place coke can into it.	I will pick up the bottle and move it to the orange.
	Pick the mug from the top drawer, then close the top drawer	The coffee mug is already on the table, so I will now close the top drawer.
	put the tool that can be used to feed a baby on the plate	I will place the spoon on the plate.
	After I finish my drink, can you place the fruit near it for a snack?	Sure, I will place the apple near the Pepsi can.
	I am thirsty but I do not want drinks. Please grab something for me.	I will pick up the orange from the table.

Figure 24: **Examples of Situated Reasoning in SimplerEnv-Instruct.** The second example’s responses is recorded before and after the drawer is open.

All other assets are created using Blender or modified from SimplerEnv [37].

E Model Design and Training Details

E.1 Instruction Format

To train captioning, question answering, and instruction-following capabilities, we integrate all tasks into a unified dialogue format. For captioning and question answering, we adopt the template shown in Prompt 3, where the captioning instruction is sampled from Prompt 4. For free-form instructions, we append the postfix “First answer my question.” to elicit a direct response from the model, as illustrated in Prompt 5.

Dialogue Format

```
[  
  {  
    "role": "system", "content": DEFAULT_SYSTEM_MESSAGE  
  },  
  {  
    "role": "user",  
    "content": "[Question]",  
    "image": image  
  },  
  {  
    "role": "assistant",  
    "content": "[Answer]"  
  },  
  {  
    "role": "user",  
    "content": "What action should the robot take to [Instruction]?"  
  },  
  {  
    "role": "assistant",  
    "content": "[Latent Action Queries]"  
  }  
]
```

Caption Prompts

- Describe what's on the table. Don't mention the robot arm.
- What objects are in the scene? Ignore the robot arm.
- Tell me what you see on the table, not the robot.
- Describe the items and their positions, but skip the robot.
- Look at the table and describe it. Don't include the arm.
- Only talk about the objects, not the machine.
- Give a short description of the scene, without the robot.
- Describe the setup on the table. Leave out the robotic arm.
- Focus on the objects and environment. Ignore the robot.
- Describe the environment and tabletop contents, excluding any robotic hardware.

Instruction Format

```
[  
  {  
    "role": "system", "content": DEFAULT_SYSTEM_MESSAGE  
  },  
  {  
    "role": "user",  
    "content": "What action should the robot take to [Instruction]? First answer my question.",  
    "image": image  
  },  
  {  
    "role": "assistant",  
    "content": "[Response] [Latent Action Queries]"  
  }  
]
```

Table 12: **Model parameters.** ‘‘Adaptor’’ and ‘‘Scale Head’’ are used for MoE adaptation. Specifically, two LoRA adaptors are used to learn latent action generation and assistant response during VLA-IT.

Component	Parameter	Value
Adaptor	Rank	128
	Alpha	256
	Dropout	0.05
	Target	Attn. Q/K/V/O MLP Up/Down
Scale Head	Depth	4
	Size	128
Action Backbone	Depth	12
	Head	12
	Hidden Size	768
	RoPE Theta	1000
Proprioception Encoder(Optional)	Hidden Size Activation	8 → 768 → 768 SiLU
Action Encoder with Time Embedding	Hidden Size Activation	7+768 → 1536 → 768 SiLU

Table 13: **Flow matching parameters.** The time steps is sampled from $p(\tau) = \beta(\frac{s-\tau}{s}; 1.5, 1)$ [47]

Component	Parameter	Value
Flow Sampling	s	0.999
	Inference Steps	10
Sinusoidal Time Embed	Max Period	100

E.2 Learning Objective and Inference Procedure

We adopt flow matching [47, 50] to learn the action chunk $\mathbf{A} \in \mathbb{R}^{H \times 7}$ [90] over a horizon H . The training objective is defined as the flow matching loss:

$$\mathcal{L}_{FM} = \mathbb{E} \left[\|V\theta(\mathbf{A}^\tau, q_t) - (\epsilon - \mathbf{A})\|^2 \right], \quad (1)$$

where $\tau \in [0, 1)$ denotes the flow step, and $V\theta(\mathbf{A}^\tau, q_t)$ is the network output conditioned on q_t , which encodes information from DINOv2 [1] and a latent action C . The interpolated noisy action is given by $\mathbf{A}^\tau = \tau\mathbf{A} + (1 - \tau)\epsilon$, with $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

During inference, we generate the action chunk using forward Euler integration:

$$\mathbf{A}^{\tau+1/N} = \mathbf{A}^\tau + \frac{1}{N} V\theta(\mathbf{A}^\tau, q_t), \quad (2)$$

starting from $\mathbf{A}^0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, with $N = 10$ denoising steps.

E.3 Model Parameters

Additional model parameters are provided in Table 12, with flow-matching sampling settings detailed in Table 13. All projectors—including those aligning latent actions and DINO-ViT visual features to the action expert’s dimension—use a simple two-layer MLP with SiLU activation. The action head, also a shallow MLP with SiLU, maps the action expert’s hidden states to $\mathbb{R}^{N \times 7}$, where $N = 16$ is the prediction horizon and 7 denotes the action dimension, including the gripper.

E.4 Inference Speed

We evaluate the inference speed of InstructVLA on a single A100 GPU with BF16 precision, as shown in Table 14. To support language feedback during evaluation (i.e., CoT inference), in the “Thinking” setting, we enable VLM auto-regressive generation every 20 action expert steps. The “Action Only” setting bypasses language generation and directly decodes latent actions via a single VLM forward pass. In the “Latent Action Caching”, latent actions are generated every two expert steps; this introduces minimal performance impact. All settings are tested without action chunking. Note that although the model predicts 16-step action sequences, only one step is executed.

Table 14: **Inference speed.** Inference speed is evaluated under three settings **without using action chunking**. Each evaluation includes a 50-step warm-up followed by 200 steps for stable measurement.

	With Language	Action Only	Latent Action Caching
Inference Frequency(Hz)	2.07	3.50	4.96

E.5 Experiments Compute Resources

The action pretraining phase takes approximately 27 hours on 64 A100 GPUs, each node equipped with 1 TB of CPU memory. The VLA-IT phase requires around 12 hours under the same GPU configuration. Simulator-based evaluations are conducted using 8 A100 GPUs. For real-world experiments, training is performed over 4 hours on 32 A100 GPUs, and deployment is carried out on a single A100 GPU.

F Multimodal Examples

Figure 25 illustrates InstructVLA’s multimodal and embodied commonsense reasoning across diverse scenarios. The model demonstrates accurate visual inference (e.g., recognizing a dog via reflection, identifying synthetic images), basic scene text recognition, and reliable grounding of objects and colors. In manipulation tasks, it interprets high-level goals, predicts appropriate next actions, and verifies task completion. These capabilities showcase its integration of perception, language, and manipulation, enabling effective performance in complex daily-life scenarios.

	
<p>How many dogs in this image? Explain why.</p> <p>There is one dog in the image. This conclusion is based on the single reflection visible in the mirror, which shows only one dog's face. The dog's reflection is clear and distinct, indicating that there is only one dog present in the scene.</p>	<p>What is on the top of the shelf</p> <p>There is a yellow block and a red block.</p>
	
<p>Is this a real image? Explain why</p> <p>This image appears to be a digital creation rather than a real photograph. The reasons include the unrealistic proportions of the tree, the smooth and uniform texture of the tree's bark, and the perfect symmetry of the branches. Additionally, the background is a solid, unvarying blue, which is not typical of natural landscapes.</p>	<p>The current goal is: open drawer, take out snacks, place on table, close drawer. what is the immediate next step based on the image?</p> <p>The immediate next step is to pick the brown chip bag from the top drawer and place it on the counter.</p>
	
<p>Can you help me recognize the words on the advertising board where a bus on it?</p> <p>The advertising boards have words like "midnight", "7", "12"</p>	<p>drop the orange sachet in the drawer. is it satisfied?</p> <p>Yes, the drawer is now open.</p>
	
<p>What is unusual about this image</p> <p>The unusual aspect of this image is that a man is ironing clothes on a makeshift ironing board attached to the back of a yellow taxi. This is not a common sight, as ironing is typically done indoors or in a designated space, and it is not practical to iron clothes while driving.</p>	<p>I'm preparing my lunch. What did I just do, and what should I do next based on the current environment?</p> <p>You've already washed the vegetables and are in the process of preparing your lunch. The next step would be to heat the cooked vegetables in the microwave, then assemble your lunch with any additional ingredients you have.</p>

Figure 25: **Zero-shot multimodal question answering.** Four commonsense and four embodied examples are selected.

G Real-world Experiments Setup and Analysis

We collect data exclusively for few-shot settings as shown in Figure 26. In the first setting, which focuses on grasping objects in a clustered arrangement, the robot is instructed to classify objects within a 20×40 cm region on the table—placing all cubic objects into a plate and all others into a box. This setting includes 70 complete episodes, totaling 677 pick-and-place actions. In the second setting, which emphasizes spatial actions, the robot is instructed to randomly grasp three objects from the top of a rack and place them into a plate. We collect 60 complete episodes for this setting, comprising 180 pick-and-place actions. The experimental setups are depicted in Figure 30.

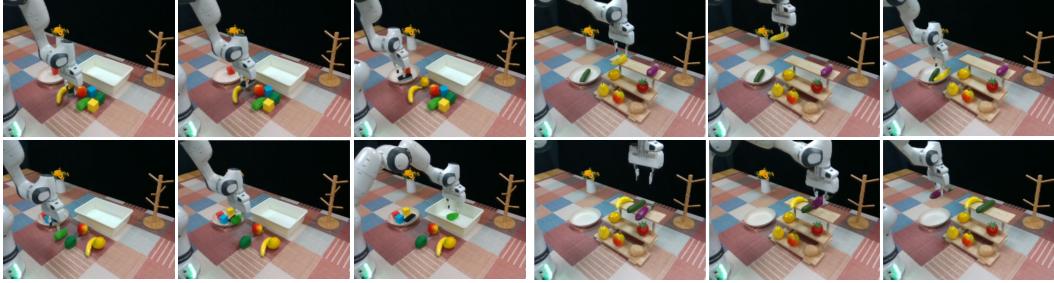


Figure 26: **Real-world dataset examples.** Four examples from the few-shot training set, illustrating cluster classification tasks (left) and rack pick-and-place tasks (right).

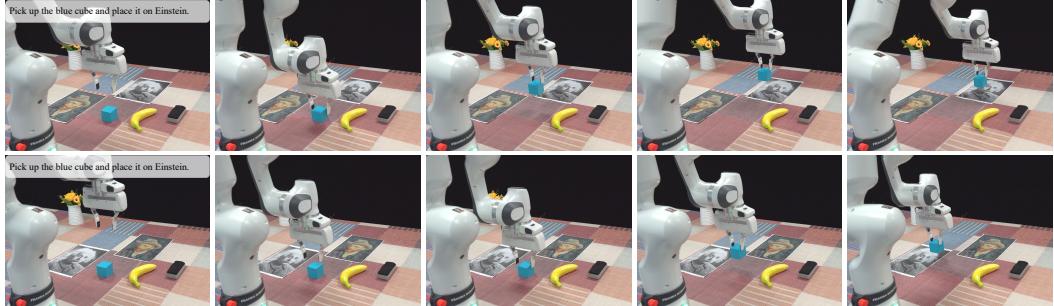


Figure 27: **Zero-shot grounding.** In a clustered pick-and-place setting, InstructVLA accurately places the blue cube by semantically grounding the reference to the celebrity.



Figure 28: **Light distraction.** Stable visual features from DINO and SigLIP enable the model to operate robustly under extreme out-of-distribution lighting conditions.

To assess semantic grounding in novel contexts, we replace the plate and box in the cluster classification setting with images of celebrities. As illustrated in Figure 27, the model accurately interprets instructions and places the blue cube correctly by leveraging object and celebrity recognition.

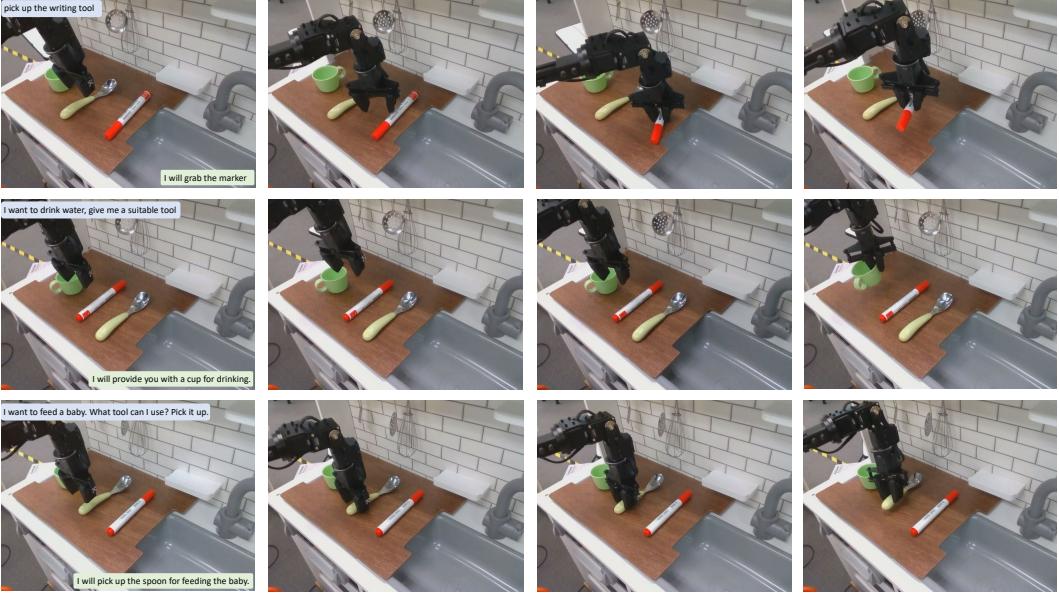


Figure 29: **Zero-shot evaluation.** We perform zero-shot evaluation in the Bridge kitchen environment with augmented background and novel objects. The instruction and model response are visualized in the first image.

Figure 28 shows that InstructVLA remains robust under extreme lighting conditions, supported by stable visual features from DINO and SigLIP. Finally, we evaluate zero-shot generalization in the Bridge kitchen environment with augmented backgrounds and unfamiliar objects. As shown in Figure 29, the model successfully follows novel instructions and completes the tasks.



Figure 30: **Real-world settings.** A third-person view is captured using an Intel D435i camera for the Franka (few-shot) and WidowX (zero-shot) settings.

H Broader Impacts and Future Work

H.1 Broader Impacts

InstructVLA contributes to the advancement of general-purpose embodied agents by integrating vision-language understanding with action generation. Its ability to follow free-form instructions and generalize to novel tasks supports applications in assistive robotics and human-robot collaboration. Nonetheless, as with other large pretrained models, careful attention must be given to potential limitations such as dataset bias and safety in real-world deployment. Ensuring responsible use and reliable performance across diverse environments is essential.

H.2 Future Work

We plan to incorporate additional sensory modalities, such as depth and tactile feedback, to enhance safety and reliability in physical interactions. Leveraging recent advances in digital twins and simulation technologies, we aim to reduce reliance on real-world data by utilizing large-scale synthetic datasets. Finally, we will extend the evaluation and deployment of InstructVLA to a broader range of environments to further assess its generalization capabilities.