FineVLA: Fine-Grained Instruction Alignment for Steerable Vision-Language-Action Policies

Xintong Hu[*x] **Xuhong Huang**[*x] **Jinyu Zhang**[x] **Yutong Yao**[x] **Yuchong Sun**[q]
Qiuyue Wang[q] **Mingsheng Li**[q] **Sicheng Xie**[q] **Yitao Liu**[x] **Junhao Chen**[x]
Yixuan Chen[x] **Yingming Zheng**[x] **Shuai Bai**[q] **Tao Yu**[†x]

[x] XLANG Lab, The University of Hong Kong [q] Qwen Team, Alibaba Inc.
Hugging Face

Abstract

Vision-Language-Action (VLA) models are increasingly expected to not only complete robot tasks, but also follow human instructions about *how* those tasks should be executed. However, existing robot datasets usually pair trajectories with coarse goal-level language, leaving execution-critical details such as active arm, target object, approach direction, contact region, motion path, and final configuration unspecified. This limits steerable policy learning and robotic video understanding. We introduce **FineVLA**, an open framework for action-aligned fine-grained VLA supervision. The framework includes: (1) a data construction tool that unifies 972,247 trajectories across 85K tasks from 10 open-source robot datasets and builds **FineVLA-Data**, a human-verified dataset of 47,159 fine-grained trajectories; (2) a held-out benchmark with 500 videos, 10,816 atomic facts, and 1,030 VQA questions; (3) a robotics-specialized VLM annotator for scalable fine-grained annotation; and (4) a steerable **VLA policy** trained with controlled mixtures of fine-grained and raw goal-level instructions. Our policy experiments yield three findings. First, fine-grained supervision does not sacrifice goal-level task success: FG-only improves over Raw-only by +1.4 to +8.1 success-rate points across architecture and data-scale settings. Second, fine-grained and raw instructions are complementary: performance follows a consistent inverted-U trend, peaking around FG : Raw = 1 : 2 to 1 : 1. The strongest mixed setting reaches **86.8%/82.5%** in RoboTwin simulation and **62.7**/100 in real-world dual-arm manipulation, compared with 49.9 for Raw-only. Third, fine-grained supervision directly improves steerable control by increasing compliance with language-specified execution factors such as object pose, color, active arm, approach direction, and rotation direction. These results show that fine-grained language should augment, rather than replace, goal-level instructions: raw language specifies *what* to achieve, while fine-grained language specifies *how* to execute.

1 Introduction

Vision-Language-Action (VLA) models are moving from task-level robot control toward policies that can be *steered* by human instructions. In this work, we use *steerability* to mean the ability to execute the same high-level goal in different ways according to user-specified execution constraints, such as which arm to use, which target object to manipulate, how to approach it, where to make contact,

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

Preprint.

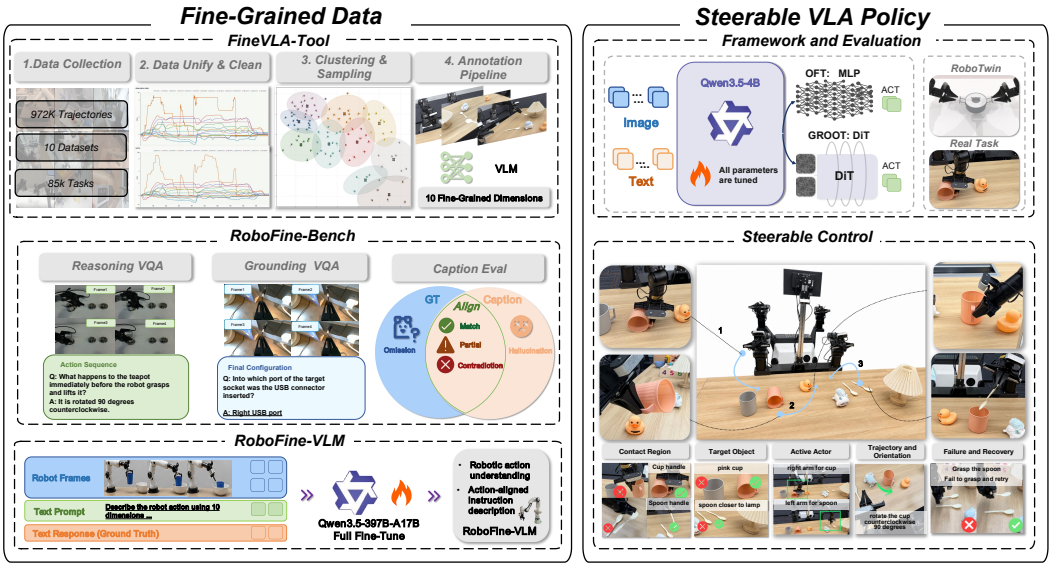


Figure 1: **Overview of FineVLA.** FineVLA builds a closed loop for action-instruction alignment, connecting fine-grained data construction, robotic video understanding, scalable annotation, and steerable VLA policy learning. **Left**: FineVLA-Tool unifies heterogeneous robot trajectories from 10 open-source datasets, removes redundant demonstrations through clustering and sampling, and annotates representative trajectories with action-aligned descriptions across ten fine-grained dimensions. The resulting FineVLA-Data supports both RoboFine-Bench, which evaluates fine-grained robotic video understanding through Grounding VQA, Reasoning VQA, and Caption Evaluation, and RoboFine-VLM, a robotics-specialized VLM trained as a scalable annotator for new trajectories. **Right**: FineVLA-Policy is trained with mixtures of raw goal-level instructions and fine-grained process-level instructions under two action-decoding architectures, and is evaluated in both RoboTwin simulation and real-world dual-arm manipulation. The steerable-control examples illustrate how fine-grained language specifies execution-sensitive factors such as contact region, target object, active actor, trajectory and orientation, and failure recovery.

which motion or rotation direction to follow, and what final configuration to achieve. Recent robot foundation models such as $\pi_{0.7}$, LingBot-VLA, GR00T N1.7, and GEN-1 (1, 2, 3, 4) suggest that future robot policies should not only infer *what* task to complete, but also follow instructions about *how* the task should be performed.

However, building open, steerable VLA systems remains challenging for three reasons. **(i) Heterogeneous data and missing fine-grained annotation infrastructure.** Existing open-source robot datasets use diverse action and state representations that cannot be directly unified (5, 6). Within the same task, demonstrations are heavily redundant, and most trajectories carry only a single goal-level description (e.g., "pick up the cup") while the execution process—actor choice, approach direction, contact region, motion path, and state transitions—remains unspecified. The problem is not only that labels are coarse, but that there is still no open infrastructure for producing action-aligned fine-grained supervision at scale. **(ii) Lack of benchmarks and scalable annotators for fine-grained robotic video understanding.** Although general video-language models and dense captioning methods have advanced video description, their captions often focus on scene appearance rather than action-relevant execution details such as contact regions, approach directions, and motion paths. Existing embodied benchmarks (7, 8, 9) mainly evaluate spatial reasoning or hand-object dynamics, but do not systematically measure whether VLMs capture process-level manipulation details. There is also a lack of open, robotics-specialized annotators for action-aligned fine-grained captions. **(iii) Unknown effectiveness and training recipe.** Even if fine-grained data were available, the community lacks systematic evidence on whether action-aligned instructions improve policy learning, and what mixture of fine-grained and goal-level supervision yields the best steerable control.

To address these challenges, we introduce **FineVLA**, a fully open-source framework for scaling fine-grained VLA data, robotic video understanding, and steerable VLA policies (Figure 1). The framework operationalizes this principle through four components, each targeting one of the gaps

above. **(1) FineVLA-Tool + FineVLA-Data (Gap i).** FineVLA-Tool unifies 972,247 trajectories across 85K tasks from 10 open-source datasets, selects representative samples via dynamic time warping (DTW)-based clustering, and annotates them with process-level descriptions across ten fine-grained dimensions (Table 9). This produces **FineVLA-Data**, a human-verified corpus of 47,159 trajectories whose average instruction length increases 10.4× (from 9.3 to 96.8 words). **(2) RoboFine-Bench (Gap ii).** We curate RoboFine-Bench—500 videos with 10,816 human-reviewed atomic facts and 1,030 VQA questions spanning all ten fine-grained dimensions, with complementary VQA and caption tracks. All benchmark trajectories are held out from both VLM fine-tuning and policy training, ensuring an independent evaluation. **(3) RoboFine-VLM (Gap ii).** We fine-tune Qwen3.5-397B-A17B (10) on FineVLA-Data to obtain RoboFine-VLM, a VLM specialized for robotic action understanding that serves as a scalable annotator for new trajectories. **(4) FineVLA-Policy + training recipe (Gap iii).** We train FineVLA-Policy under two action-decoding architectures (StarVLA-OFT and StarVLA-GR00T) and systematically vary the ratio between fine-grained (FG) and raw goal-level (Raw) instructions— keeping trajectories, actions, and visual observations fixed while changing only the paired language—to isolate the effect of action-aligned supervision.

Our policy experiments yield three key findings. First, fine-grained supervision does not harm goal-level task success; instead, FG-only outperforms Raw-only across the evaluated simulation settings, with the largest gain on AlohaMix-OFT (+6.5/+4.7 points on Easy/Hard). Second, fine-grained and raw instructions are complementary: success follows an inverted-U trend over the FG:Raw ratio and peaks around 1:2–1:1, reaching **86.8%/82.5%** on AlohaMix-OFT Easy/Hard (+15.0/+11.1 over the Raw-only baseline of 71.8%/71.4%). Third, in real-world dual-arm manipulation, the FG:Raw = 1:1 policy achieves the highest average score (**62.7**/100 vs. 49.9 for Raw-only) and reduces instruction violations from 34% to **12%**. Under identical initial configurations and visual observations, varying only the fine-grained instruction produces distinctly different execution behaviors, directly demonstrating steerable control. In addition, RoboFine-VLM achieves the best performance among evaluated VLMs on the held-out RoboFine-Bench, reaching **71.0%** VQA accuracy and **83.6%** captioning score under the hard setting, providing evidence that our annotation schema captures action-relevant manipulation details. We release the complete FineVLA suite—data pipeline, fine-grained annotations, benchmark, model checkpoints, and training code—to provide open foundations for steerable VLA research.

2 FineVLA Data: Construction, Benchmark, and Scalable Annotation

This section describes the data and annotation substrate of FineVLA. We first unify heterogeneous robot demonstrations and construct human-verified fine-grained action-aligned instructions. We then build a held-out benchmark to evaluate process-level robotic video understanding, and finally instantiate RoboFine-VLM as a scalable annotator for extending the same annotation schema to new trajectories.

FineVLA-Tool converts large-scale heterogeneous robot datasets into fine-grained, action-aligned instruction supervision (Figure 2). Its design addresses three practical bottlenecks in open robot data: (1) inconsistent action/state formats across datasets, (2) heavy redundancy among demonstrations of the same task, and (3) sparse, task-level instruction annotations. Starting from 972,247 trajectories across 10 source datasets, FineVLA-Tool produces FineVLA-Data, a human-verified corpus of 47,159 representative trajectories with fine-grained process-level supervision.

2.1 FineVLA-Tool: Canonicalization, Clustering, and Annotation

Data collection and format conversion (Figure 2, Stage 1). We aggregate 972,247 trajectories from 10 open-source datasets (11, 12, 13, 14, 15, 16, 17, 18, 6, 19), convert them to the LeRobot 2.1 format, and filter out invalid or degenerate recordings. The full per-dataset breakdown is in Appendix A.1.1.

Action-state canonicalization and cleaning(Figure 2, Stage 2). Across datasets, action and state values differ in temporal reference (absolute, relative, or delta) and kinematic representation (joint space vs. end-effector space with varied rotation encodings). We canonicalize all trajectories to absolute coordinates with normalized quaternion rotations, then remove trajectories whose action-state DTW distance exceeds a dataset-specific threshold, filtering out corrupted logs or inconsistent control conventions. Details and conversion examples are in Appendix A.1.2.

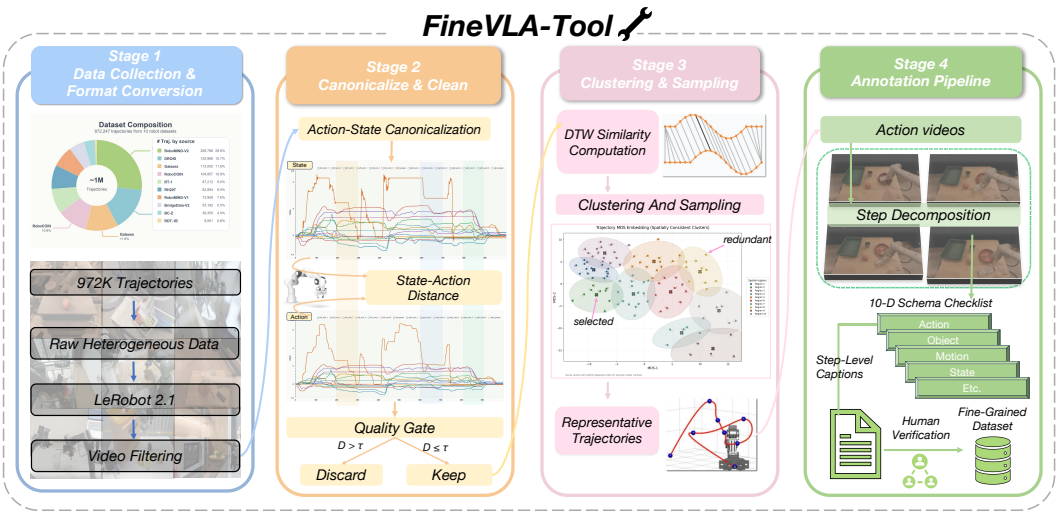


Figure 2: **Pipeline of FineVLA-Tool.** FineVLA-Tool converts large-scale heterogeneous robot demonstrations into action-aligned fine-grained instruction data through four stages. **Stage 1**: raw trajectories from 10 open-source robot datasets are converted into a unified LeRobot-style format and filtered to remove invalid videos. **Stage 2**: action and state representations are canonicalized across embodiments, and an action-state consistency quality gate removes corrupted or inconsistent trajectories. **Stage 3**: dynamic time warping (DTW)-based similarity computation and clustering identify representative trajectories, reducing redundancy while preserving diverse manipulation strategies. **Stage 4**: selected trajectories are decomposed into step-level descriptions and annotated with a ten-dimensional fine-grained schema, followed by human verification. The resulting FineVLA-Data provides human-verified, process-level supervision for training RoboFine-VLM as a scalable annotator and FineVLA-Policy as a steerable VLA policy.

Trajectory clustering and representative sampling(Figure 2, Stage 3). Open robot datasets contain many near-duplicate demonstrations within the same task, often differing only in execution speed or minor spatial offsets. To maximize annotation diversity under a fixed budget, we cluster trajectories within each task using DTW over canonicalized action sequences, followed by hierarchical clustering on the resulting distance matrix. We then select high-quality representatives from each cluster according to cluster size and trajectory quality. This reduces 972,247 raw trajectories to 47,159 representative samples while preserving diversity in manipulation strategies, object interactions, and motion patterns. Details of the DTW formulation, action-space normalization, frame costs, and clustering procedure are provided in Appendix A.1.4.

Fine-grained multi-aspect annotation(Figure 2, Stage 4). Each selected trajectory is annotated with a ten-dimensional schema capturing the control-relevant factors that goal-level instructions omit: *action sequence*, *active actor*, *target object*, *initial configuration*, *final configuration*, *contact and approach*, *trajectory and orientation*, *object interaction*, *failure and recovery*, and *body motion*. Detailed definitions and examples are provided in Appendix A.1.5, Table 9. Annotation proceeds in two phases: we first input sampled video frames from each trajectory into Qwen3.5-Plus (10), which decomposes the manipulation process into temporally ordered steps and fills structured slots for actor, target, contact region, motion path, and state change; human annotators then review the model-generated descriptions against the original video, correcting factual errors and verifying temporal alignment. The result is FineVLA-Data, a human-verified fine-grained instruction dataset for training RoboFine-VLM and downstream controllable VLA policies.

2.2 FineVLA-Data Statistics

Table 1 summarizes the statistics of FineVLA-Data. Fine-grained annotations dramatically increase instruction information density compared to original coarse instructions: the average word count per trajectory rises from 9.3 to 96.8, an approximately 10.4× increase, while covering 47 unique action verbs across all sources. Detailed source dataset statistics are reported in Appendix A.1.1, Table 6.

Table 1: **FineVLA-Data statistics.** Fine-grained annotations dramatically increase instruction information density compared to original coarse instructions across all data sources.

Source	Trajectories	Steps	Avg. Words (Coarse)	Avg. Words (FG)	Density ↑
BridgeData-V2	4,958	21,554	10.1	61.7	6.1×
BC-Z	1,513	5,313	5.2	51.2	9.8×
RT-1	5,232	22,023	6.8	61.4	9.1×
Galaxea	2,834	18,484	4.7	219.9	47.1×
RoboMIND-V1	4,605	20,341	8.6	72.8	8.5×
RoboMIND-V2	7,119	39,166	6.6	98.8	14.9×
RoboCOIN	8,513	43,926	16.1	122.6	7.6×
RH20T	1,387	5,560	7.9	92.1	11.7×
RDT	1,275	8,437	16.9	114.0	6.7×
DROID	9,723	35,802	8.0	90.9	11.3×
Total	**47,159**	**220,606**	9.3	96.8	10.4×

2.3 RoboFine-Bench: Fine-Grained Robotic Video Understanding Benchmark

We introduce RoboFine-Bench to evaluate whether VLMs capture execution-level details of robot manipulation. The benchmark contains 500 videos from 10 robot datasets, covering 32 embodiments, diverse camera views, and a wide range of manipulation tasks. Each trajectory is paired with human-reviewed step-level annotations decomposed into 10,816 atomic facts across ten action-relevant dimensions (Table 9), with an average of 4.3 steps and 21.6 facts per sample. All 500 benchmark trajectories are strictly disjoint from both the RoboFine-VLM SFT training set and all policy-training splits—no trajectory appears in both the 47,159 training samples and the benchmark. Figure 3 illustrates the benchmark statistics and structure.

RoboFine-Bench contains two complementary tracks. The **VQA track** (Figure 3, right bottom) evaluates discriminative understanding through 1,030 questions distributed across the same ten fine-grained dimensions used in annotation, which are aggregated into three reporting axes: **Entity and Scene Grounding**, **Action and Motion Understanding**, and **Interaction and State Reasoning** (Table 16 in Appendix A.3.2). Each model receives video frames and all questions for one sample in a single prompt, and answers are scored by deterministic matching against ground-truth labels. The **Caption track**(Figure 3, right top) evaluates generative understanding by asking models to produce action-aligned step-level fine-grained descriptions of the manipulation process. Generated captions are then judged by an LLM against pre-extracted ground-truth atomic facts, yielding per-fact alignment labels (match, partial, contradiction, omission, hallucination) that are aggregated into Consistency, Coverage, and Anti-Hallucination metrics. Two settings are evaluated: *easy*, where the original task instruction is provided, and *hard*, where the model must infer the process from visual observations alone. Full prompt templates and the evaluation protocol are provided in Appendix A.3.4.

2.4 RoboFine-VLM: Scalable Fine-Grained Annotator

While FineVLA-Data is human verified, scaling the same annotation schema to future robot trajectories still requires a robotics-specialized annotator. General-purpose VLMs often miss execution-level details such as contact regions, approach directions, grasp types, motion paths, and object state transitions, leading to substantial human correction cost.

We therefore fine-tune Qwen3.5-397B-A17B (10) on the human-verified FineVLA-Data to obtain RoboFine-VLM. Given a robot manipulation video, RoboFine-VLM generates temporally action-aligned step-level fine-grained descriptions covering the ten fine-grained dimensions (Table 9). The model serves as a scalable annotator for future data expansion; all policy experiments in this paper use FineVLA-Tool-generated and human-verified annotations, rather than RoboFine-VLM-generated labels. Importantly, RoboFine-VLM is not used to generate the supervision for our policy experiments; it is trained to support future scalable annotation and open-source reproducibility. Full training details are provided in Appendix A.2.2; its annotation quality is evaluated on the held-out RoboFine-Bench in Section 4.2.

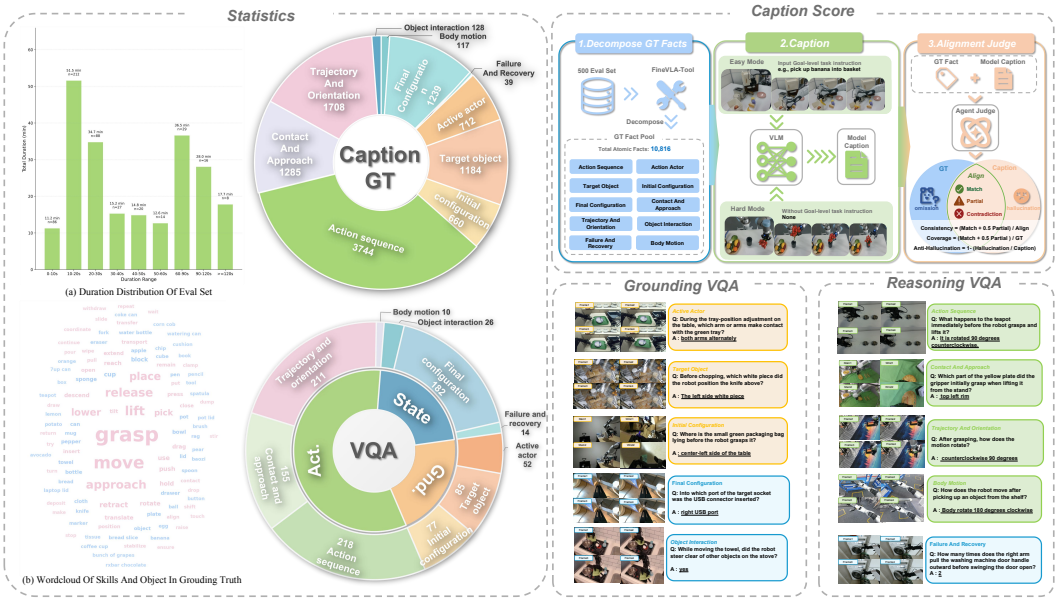


Figure 3: **Overview of RoboFine-Bench.** RoboFine-Bench evaluates fine-grained robotic video understanding through complementary VQA and captioning tracks. **Left**: benchmark statistics, including the video-duration distribution, the word cloud of manipulation skills and objects, and the distribution of ground-truth atomic facts across the ten FineVLA dimensions for captioning and VQA. **Top right**: the captioning track decomposes human-reviewed annotations into a pool of 10,816 atomic facts, asks VLMs to generate ordered step-level action descriptions under easy and hard settings, and uses an LLM judge to align model captions with ground-truth facts, producing Consistency, Coverage, and Anti-Hallucination scores. **Bottom right**: representative Grounding VQA and Reasoning VQA examples probe object/scene grounding, action/motion understanding, and interaction/state reasoning. RoboFine-Bench contains 500 held-out robot videos and 1,030 VQA questions across diverse embodiments, camera views, and manipulation scenarios.

3 Training Fine-Grained VLA Policies

With human-verified fine-grained instructions fixed, we now study how they should be used to train steerable VLA policies. Our goal is not to propose a new policy architecture, but to isolate the effect of action-aligned language supervision. We therefore keep actions, and visual observations fixed, and vary only the instruction paired with each trajectory.

3.1 FineVLA-Policy Architecture

We instantiate FineVLA-Policy under multiple action-decoding frameworks to isolate the effect of instruction supervision from architectural choices. Rather than proposing a new architecture, we adopt two existing frameworks implemented in the StarVLA codebase (20), both built on a shared Qwen3.5-4B vision-language backbone.

StarVLA-OFT attaches a lightweight MLP regression head that reads the hidden states of predefined action tokens and predicts continuous action chunks in parallel with an L1 objective, following OpenVLA-OFT. **StarVLA-GR00T** adopts a dual-system design where the VL backbone serves as System 2 for slow reasoning and a DiT-based flow-matching module serves as System 1 for continuous action generation, consistent with GR00T N1.5. Both variants produce multi-step action chunks and share the same visual observations and language inputs; only the action decoding differs. Using two architectures lets us verify that the benefits of fine-grained supervision are architecture-independent.

3.2 Training Data Mixtures

To isolate the effect of language supervision, we construct two parallel training datasets from the same source trajectories. The **FG dataset** contains the representative trajectories selected by FineVLA-Tool, each paired with its fine-grained process-level instruction (1,287 trajectories for RDT; 5,872 for

AlohaMix). The **Raw dataset** contains *all* source trajectories, each paired with its original goal-level instruction (6,061 for RDT; 84,067 for AlohaMix). AlohaMix is an ALOHA-compatible dual-arm mixture aggregated from RDT, RoboCOIN, RoboMIND-V1.0, and RoboMIND-V2.0, containing 86,662 episodes across 598 tasks (Table 21 in Appendix A.4.2). We restrict the mixture to a single embodiment class to avoid cross-embodiment confounds. Trajectories that appear in both datasets share identical action labels and visual observations; only the paired language instruction differs.

The FG:Raw ratio controls the probability of drawing from each dataset at every training step, and therefore determines the relative number of training iterations that use fine-grained versus raw instructions. For example, FG:Raw = 2:1 means the FG dataset is sampled with twice the weight of the Raw dataset, so approximately two-thirds of training steps use a fine-grained instruction and one-third use a raw instruction. Under Raw-only, training draws exclusively from the Raw dataset; under FG-only, exclusively from the FG dataset.

We compare seven configurations: Raw-only, FG:Raw = 1:4, 1:2, 1:1, 2:1, 4:1, and FG-only. We study three (dataset, framework) combinations—RDT-OFT, RDT-GR00T, and AlohaMix-OFT—to control for architecture and data-scale effects. This design isolates the effect of action-aligned language supervision from changes in data scale, embodiment, or action distribution.

4 Experiments

We evaluate FineVLA along three axes: (1) whether RoboFine-VLM captures fine-grained robotic action details (RoboFine-Bench, Section 4.2), (2) whether fine-grained supervision improves policy learning in simulation (RoboTwin, Section 4.3), and (3) whether the resulting policies exhibit steerable control on real-world dual-arm manipulation (Section 4.4).

4.1 Experimental Setup

Evaluation benchmark. We evaluate FineVLA on three complementary evaluation protocols that measure robotic video understanding, simulated policy learning, and physical steerable control.

(1) RoboFine-Bench. RoboFine-Bench (Section 2.3) evaluates whether RoboFine-VLM captures fine-grained robotic action details. We compare RoboFine-VLM with five strong general-purpose VLMs on both VQA and captioning tracks. The VQA track reports overall and dimension-wise accuracy across the ten FineVLA dimensions, while the captioning track scores ordered step-level action descriptions using Consistency, Coverage, and Anti-Hallucination metrics.

(2) RoboTwin Simulation Evaluation. RoboTwin (21) evaluates simulated bimanual manipulation. We test the seven FG:Raw instruction ratios defined in Section 3.2 across three controlled policy settings: RDT-OFT, RDT-GR00T, and AlohaMix-OFT. Policies are evaluated on the official Easy and Hard splits, with 20 episodes per task.

(3) Real-world Steerability Evaluation. We design this self-contained real-world benchmark on a Cobot Magic dual-arm platform to measure language-conditioned controllability. Unlike broad robustness benchmarks that vary scenes, objects, or lighting, our suite isolates instruction following: for each instruction-sensitive task family, paired variants use the same object set and nearly identical initial scene layout while changing only one language-specified control factor. The suite includes two general manipulation tasks, five in-distribution instruction-sensitive task families (each comprising a paired variant) covering object color, object pose, approach direction, rotation direction, and active arm, and one out-of-distribution active-arm–target binding probe. Each paired variant is evaluated over 10 trials and scored with a partial-completion rubric normalized to a 0–100 scale. Additional hardware and inference details are reported in Appendix A.6.1.

Training setup. We train three policy configurations—**RDT-OFT**, **RDT-GR00T**, and **AlohaMix-OFT**—to decouple architecture and data-mixture effects. RDT-OFT and RDT-GR00T use the same RDT pretraining data with different action decoders, while RDT-OFT and AlohaMix-OFT use the same OFT decoder with different pretraining mixtures. We pretrain each configuration for 100k steps on 64 A100 GPUs with per-device batch size 8 and global batch size 512.

For RoboTwin evaluation, we fine-tune the pretrained checkpoints on the union of the Clean and Random training sets, containing 27,500 trajectories and 6,075,103 transitions, for 100k steps on

Table 2: **VQA benchmark results on RoboFine-Bench (%).** We report overall VQA accuracy together with all ten fine-grained capability dimensions. **AA**: Active Actor; **TO**: Target Object; **IC**: Initial Configuration; **AS**: Action Sequence; **C&A**: Contact & Approach; **T&O**: Trajectory & Orientation; **BM**: Body Motion; **OI**: Object Interaction; **FC**: Final Configuration; **F&R**: Failure & Recovery. Best value per column is **bold**.

Model	Overall↑	Gnd.↑			Act.↑				State↑		
		AA	TO	IC	AS	C&A	T&O	BM	OI	FC	F&R
Qwen3-VL-Plus	50.4	68.9	51.8	55.0	62.1	43.0	43.7	63.6	50.0	46.0	50.0
Qwen3.5-Plus	52.6	70.5	47.1	62.5	55.0	45.5	47.4	72.7	26.9	58.4	42.9
Doubao-Seed-2.0-Pro	54.9	60.7	55.3	61.3	61.4	50.0	45.1	72.7	42.3	61.6	50.0
Gemini-3.1-Pro	62.1	83.6	**67.1**	68.8	72.9	52.6	52.1	63.6	23.1	**67.6**	50.0
GPT-5.4	61.0	85.1	60.0	58.8	66.4	61.5	50.7	63.6	50.0	65.4	28.6
RoboFine-VLM (Ours)	**71.0**	**85.2**	63.5	**72.5**	**73.6**	**67.3**	**56.7**	**81.8**	**57.7**	66.5	**85.7**

Table 3: **Caption benchmark results on RoboFine-Bench (%).** We report caption quality under two settings: **easy**, where the original task instruction is provided, and **hard**, where the model must infer the manipulation process from vision alone. **Cons.**: Consistency; **Cov.**: Coverage; **A-Hal.**: Anti-Hallucination. Best value per column is **bold**.

Model	Easy				Hard			
	Overall↑	Cons.↑	Cov.↑	A-Hal.↑	Overall↑	Cons.↑	Cov.↑	A-Hal.↑
Qwen3-VL-Plus	76.8	75.6	60.4	94.4	65.1	68.7	57.0	69.6
Qwen3.5-Plus	77.9	76.0	61.7	**96.0**	72.5	70.9	56.8	89.7
Doubao-Seed-2.0-Pro	80.2	79.6	72.1	88.9	68.2	72.2	65.6	66.8
Gemini-3.1-Pro	81.3	80.8	69.8	93.2	77.2	77.0	61.3	93.4
GPT-5.4	83.1	80.8	75.1	93.4	78.1	74.2	68.9	91.1
RoboFine-VLM (Ours)	**85.2**	**83.9**	**76.7**	95.1	**83.6**	**81.9**	**75.3**	**93.7**

8 A100 GPUs with global batch size 128. The FG:Raw instruction mixture is applied during this fine-tuning stage; pretraining uses the original instruction format of each source dataset.

For real-world evaluation, we further fine-tune the corresponding simulation checkpoint for each instruction-mixture setting on 50 demonstrations per task from 12 tabletop tasks, for 600 demonstrations in total, collected on the Cobot Magic dual-arm platform. Full optimizer, hardware, batch-size, and training-step configurations are reported in Appendix A.4, Table 20.

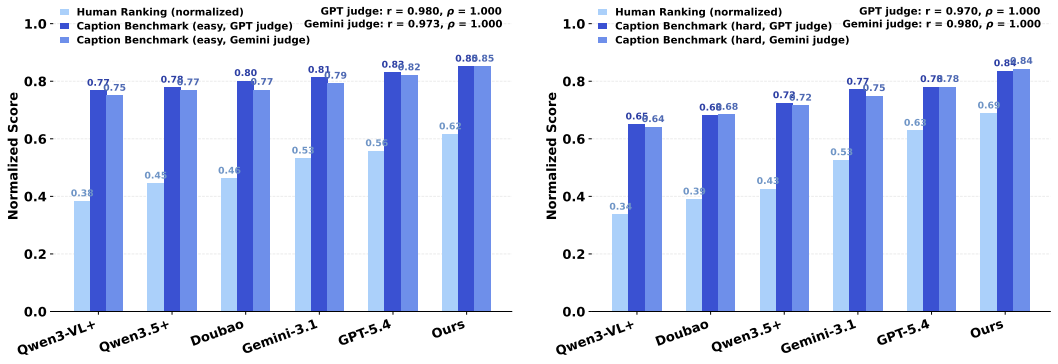
4.2 RoboFine-Bench Results

Tables 2 and 3 compare RoboFine-VLM with five strong general-purpose VLMs on RoboFine-Bench. A more detailed analysis of the benchmark results is provided in Appendix A.3.8.

VQA results. RoboFine-VLM achieves **71.0%** overall accuracy, outperforming the strongest general-purpose baseline, Gemini-3.1-Pro, by **8.9** absolute points. The largest gain appears on Action and Motion Understanding (**68.4%** vs. **58.4%**), indicating that fine-grained supervision improves execution-level reasoning beyond scene recognition. Compared with its base model Qwen3.5-Plus (i.e., Qwen3.5-397B-A17B), SFT on FineVLA-Data improves overall accuracy from **52.6%** to **71.0%**, with consistent gains across grounding, action, and state reasoning.

Caption results. RoboFine-VLM also leads the caption track. In the easy setting, where the task instruction is provided, it obtains the best Overall, Consistency, and Coverage scores. In the hard setting, where the model must infer the manipulation process from video alone, RoboFine-VLM ranks first on all four metrics and improves Overall from the strongest baseline score of **78.1%** to **83.6%**. This setting is especially important because it measures whether the model captures the execution process rather than relying on task-level language priors. Token and latency statistics are reported in Appendix A.3.7.

Benchmark validity. The caption ranking is robust to the choice of alignment judge: replacing GPT-5.4-Pro with Gemini-3.1-Pro yields the same model ranking in both easy and hard settings, with only small changes in absolute scores (Appendix A.3.5). The automatic scores also align strongly



(a) Comparison on easy mode.

(b) Comparison on hard mode.

Figure 4: **Correlation between benchmark caption scores and human ranking.** We recruit 10 human raters to rank the six models on the 500 benchmark videos, and average the resulting subjective scores. Human ranks are normalized from the 1–6 range to $[0, 1]$, while benchmark caption Overall scores are normalized from 0–100 to $[0, 1]$.

Table 4: **RoboTwin simulation success rate (%).** We compare three training settings (RDT-OFT, RDT-GR00T, and AlohaMix-OFT) under seven FG:Raw instruction ratios. Easy/Hard follow the official RoboTwin splits. Best value per column is **bold**.

FG:Raw	RDT-OFT		RDT-GR00T		AlohaMix-OFT	
	Easy↑	Hard↑	Easy↑	Hard↑	Easy↑	Hard↑
Raw-only	61.5	60.0	55.1	53.4	71.8	71.4
FG : Raw = 1 : 4	68.2	66.5	58.2	55.7	75.3	74.3
FG : Raw = 1 : 2	**74.1**	72.1	61.7	60.9	82.8	78.6
FG : Raw = 1 : 1	73.9	**72.4**	**69.4**	**68.2**	**86.8**	**82.5**
FG : Raw = 2 : 1	70.4	68.3	65.9	63.1	80.9	79.3
FG : Raw = 4 : 1	68.6	67.5	64.9	63.2	79.5	78.5
FG-only	62.9	62.0	62.1	61.5	78.3	76.1

with human preference. As shown in Figure 4, the correlation between benchmark Overall scores and the rankings from 10 human raters is high in both settings (easy: Pearson **0.980**, Spearman ρ **1.000**; hard: Pearson **0.970**, Spearman ρ **1.000**).

These results provide evidence that RoboFine-VLM can produce dense, action-aligned robotic descriptions, and that the proposed annotation schema captures execution-level manipulation details. Importantly, the fine-grained supervision used for policy training is produced by FineVLA-Tool with human verification, rather than by RoboFine-VLM. Thus, RoboFine-VLM is evaluated here as a scalable annotator for future data expansion.

4.3 RoboTwin Simulation Results

We evaluate on RoboTwin (21), a simulation benchmark for bimanual manipulation, and report success rate on its official **Easy** and **Hard** splits. Each policy is evaluated over 20 episodes per task and averaged to produce the per-split score. Table 4 reports results across three (dataset, framework) combinations: RDT-OFT, RDT-GR00T, and AlohaMix-OFT. Note that AlohaMix is approximately 13× larger than RDT in episode count, enabling a controlled study of data-scale effects.

Table 4 shows two main results. First, fine-grained supervision does not harm goal-level task success: FG-only improves over Raw-only across all evaluated settings, with gains of +1.4/+2.0 on RDT-OFT (Easy/Hard), +7.0/+8.1 on RDT-GR00T, and +6.5/+4.7 on AlohaMix-OFT. Second, fine-grained and raw instructions are complementary: as the FG proportion increases from 0% to 100%, success rate follows a consistent inverted-U trend across all three settings, peaking around FG : Raw = 1 : 2 to 1 : 1. The best setting, FG : Raw = 1 : 1, reaches **86.8%/82.5%** on AlohaMix-OFT Easy/Hard, a gain of +15.0/+11.1 over the Raw-only baseline (71.8%/71.4%). Both conclusions hold across all three

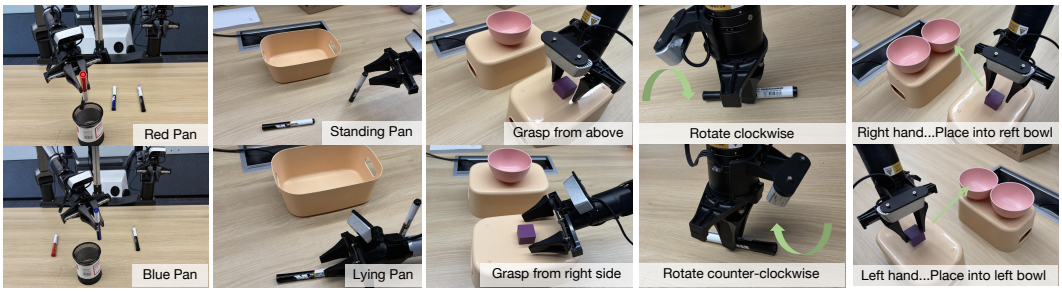


Figure 5: **Paired real-world evaluation.** Each column shows one control factor under the same visual scene with two language variants. From left to right: Color (red/blue), Pose (standing/lying), Approach (above/side), Rotation (clockwise/counterclockwise), Arm (right/left).

Table 5: **Real-world scores on a 100-point scale.** All models use StarVLA-OFT pretrained on AlohaMix (100k fine-tuning steps). Each trial is scored by manually checking ordered subgoals; a completed subgoal receives proportional credit, and the final score is normalized to 100. *Avg (ID)* averages the seven in-distribution tasks; *Avg (All)* includes the OOD probe (†). **Bold** indicates the best score per column. Task descriptions and control factors are listed in Table 24.

Supervision	**In-Distribution Tasks**							**OOD**	**Average**	
	Clean Table	**Stack Block**	**Color**	**Pose**	**Approach**	**Rotate**	**Arm**	**L→R**†	**(ID)**	**(All)**
Raw-only	72	35	22	24	60	76	60	0	49.9	43.6
FG : Raw = 1 : 4	76	36	28	32	65	79	61	0	53.9	47.1
FG : Raw = 1 : 2	79	39	36	**48**	76	**87**	63	5	61.1	54.1
FG : Raw = 1 : 1	**84**	**40**	**40**	47	**78**	86	**64**	**10**	**62.7**	**56.1**
FG : Raw = 2 : 1	80	38	34	42	72	83	62	5	58.7	52.0
FG : Raw = 4 : 1	74	37	31	43	72	83	62	5	57.4	50.9
FG-only	70	35	25	41	70	80	60	0	54.4	47.6

† $L \rightarrow R$: use left hand to place into right bowl; unseen actor-target combination (OOD compositional probe). Control factors are detailed in Appendix A.6.1.

(dataset, framework) combinations, regardless of action-decoding architecture (OFT vs. GR00T) and pretraining data scale (RDT vs. AlohaMix). We analyze this trend and its mechanism in Section 5.2.

4.4 Real-World Steerability Results

We evaluate physical steerable control on our self-designed Real-world Steerability Suite (Figure 5). This benchmark is designed to isolate language-conditioned controllability: paired task variants share nearly the same visual scene but require different behaviors according to the instruction, such as choosing a different object color, object pose, approach direction, rotation direction, or active arm.

Table 5 shows two main results. First, fine-grained supervision improves steerable control—a conclusion that simulation benchmarks alone cannot provide. FG : Raw = 1 : 1 improves every instruction-sensitive factor over Raw-only: Color (22 → **40**), Pose (24 → **47**), Approach (60 → **78**), Rotate (76 → **86**), and Arm (60 → **64**). The largest gains appear on factors invisible to goal-level language—Pose (+23), Color and Approach (+18 each)—precisely the execution choices that raw instructions leave unspecified. Second, consistent with the simulation findings, fine-grained and raw instructions are complementary in the real world. The in-domain score follows a clear inverted-U trend across all seven settings (49.9 → 53.9 → 61.1 → **62.7** → 58.7 → 57.4 → 54.4), peaking at FG : Raw = 1 : 1 (**62.7**/100), which outperforms both Raw-only (49.9) and FG-only (54.4). On the two general manipulation tasks, Clean Table (72 → **84**) and Stack Block (35 → **40**), mixed supervision also matches or exceeds Raw-only, indicating that process-level language does not interfere with routine execution. The OOD actor-target binding remains challenging, but the mixed model improves from 0 to 10/100, suggesting partial factor-level generalization. We analyze factor-level controllability and failure modes in Section 5.4.

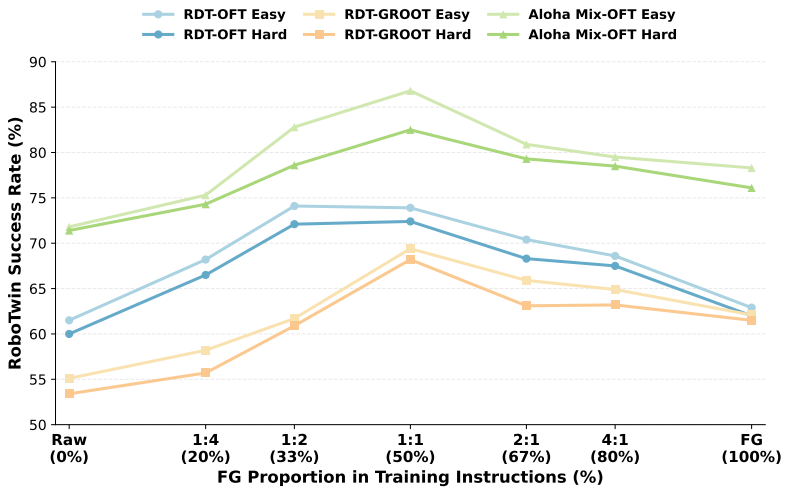


Figure 6: **RoboTwin mixing-ratio curves.** Performance peaks around FG : Raw = 1 : 2 to 1 : 1 across all settings, yielding a consistent inverted-U trend.

5 Analysis

This section analyzes *why* fine-grained supervision improves performance, how it should be mixed with raw goal-level instructions, and which control factors benefit most from action-aligned language.

5.1 Fine-Grained Supervision Does Not Sacrifice Goal-Level Success

A natural concern is that fine-grained instructions may over-specify execution details and distract the policy from completing the high-level goal. Our results suggest the opposite. In RoboTwin (Table 4), FG-only improves over Raw-only across all three (dataset, framework) combinations, with gains ranging from +1.4/+2.0 (RDT-OFT, Easy/Hard) to +7.0/+8.1 (RDT-GR00T) and +6.5/+4.7 (AlohaMix-OFT). In the real-world evaluation (Table 5), mixed supervision also matches or exceeds Raw-only on the two general manipulation tasks (Clean Table, Stack Block), where no fine-grained control factor is explicitly tested. The pattern holds regardless of decoder architecture (OFT vs. GR00T), pretraining data scale (RDT vs. AlohaMix), and environment (simulation vs. physical), indicating that process-level language provides additional action constraints without sacrificing goal-level task completion.

5.2 Raw and Fine-Grained Supervision Are Complementary

Although FG-only outperforms Raw-only, the *best* performance is achieved by mixing both supervision types. As the FG proportion increases from 0% to 100%, success rate traces a clear inverted-U curve in all three RoboTwin settings, peaking around FG : Raw = 1 : 2 to 1 : 1 (Figure 6). The same trend transfers to the real world (Table 5): FG : Raw = 1 : 1 achieves the highest in-domain score (62.7/100), outperforming both Raw-only (49.9) and FG-only (54.4).

We attribute this inverted-U to the complementary roles of the two instruction types. Raw instructions preserve compact goal semantics—*what* task should be completed—while fine-grained instructions expose execution constraints—*how* the task should be performed. Under *Raw-only*, execution-level choices (arm, approach, rotation) are left to implicit co-occurrence statistics. Under *FG-only*, the policy has explicit process-level guidance, but losing goal-level abstractions may weaken generalization to instruction phrasings not seen during training. In addition, FG descriptions are longer and more distributionally different from natural user commands, so training exclusively on FG language may reduce exposure to compact goal-level task phrasing. Under *Mixed* supervision, the policy simultaneously learns task semantics from raw instructions and execution constraints from fine-grained descriptions, retaining the strengths of both. The inverted-U trend suggests that fine-grained language should augment, not replace, raw task instructions.

5.3 Architecture and Data-Scale Effects

FG supervision narrows the architecture gap. Comparing StarVLA-OFT and StarVLA-GR00T on the same dataset (RDT), OFT is clearly stronger under Raw-only supervision (gap of 6.4/6.6 on Easy/Hard), but the gap shrinks as FG ratio increases and nearly vanishes under FG-only (0.8/0.5). This suggests that dense language supervision alleviates a supervision bottleneck, reducing the policy's dependence on decoder architecture choice.

FG supervision benefits more from larger data scale. Comparing RDT-OFT and AlohaMix-OFT, the gain from FG supervision is larger on the bigger AlohaMix dataset. The FG-only improvement over Raw-only grows from +1.4/+2.0 (RDT) to +6.5/+4.7 (AlohaMix). As trajectory diversity grows, dense action-aligned language has more distinct execution patterns to bind to. Together with the architecture result above, this suggests that fine-grained supervision is not merely an incidental improvement for current-scale training: it represents a scalable supervision axis beyond a single architecture or dataset scale.

Detailed per-setting numbers supporting both observations are reported in Appendix A.5, Table 23.

5.4 Fine-Grained Language Enables Factor-Level Steerable Control

Overall task success can hide instruction violations: a policy may complete the goal-level task while using the wrong arm, approaching from the wrong direction, or rotating in the wrong direction. We therefore examine the five single-factor columns in Table 5, where each column isolates exactly one language-specified control attribute while holding the visual scene fixed.

Table 5 shows that FG : Raw $= 1:1$ improves every instruction-sensitive factor over Raw-only. The largest gains appear on attributes invisible to goal-level language: Pose ($24 \rightarrow 47$, +23), Color ($22 \rightarrow 40$, +18), and Approach ($60 \rightarrow 78$, +18). Rotate improves from 76 to 86 (+10), and Arm from 60 to 64 (+4). The gain magnitude correlates with how much each factor is underspecified by raw instructions: object pose, color, and approach direction receive no guidance in goal-level language, while rotation direction and arm selection are occasionally implied by task context. These results show that fine-grained supervision improves not only overall task completion, but also execution compliance on the specific control attribute specified by the instruction.

The OOD actor-target probe reveals a different pattern. The FG : Raw $= 1:2$ and $1:1$ settings achieve the highest OOD scores (5 and 10 respectively), compared with 0 for Raw-only, suggesting that mixed supervision strengthens individual factor grounding. However, this does not translate into full task completion because the policy still fails to bind the selected arm to the unseen target receptacle. Thus, FineVLA improves factor-level controllability, but full compositional generalization remains unsolved.

5.5 Limitations

The remaining real-world failures fall into two categories. The first is *grounding failure*, where the policy selects the wrong object, arm, or target despite the language specifying the correct factor. The second is *execution failure*, where the correct factor is selected but the physical manipulation fails, such as unstable grasping, incomplete rotation, or inaccurate placement. The OOD actor-target probe further shows a compositional limitation: increasing FG supervision improves active-arm grounding, but does not reliably solve novel actor-target binding.

Our framework also has several limitations. RoboFine-VLM reduces annotation cost but does not fully remove human verification. Real-world validation is still limited to a tabletop dual-arm platform and a small set of targeted steerability tasks. Finally, following fine-grained execution instructions in physical environments raises safety concerns; future systems should combine fine-grained language following with feasibility and safety checks.

6 Related Work

VLA policy learning and sparse trajectory language. Recent VLA policies such as RT-2 (22), OpenVLA (23), π_0 (24), and Octo (25) leverage pretrained vision-language models and large demonstration datasets such as Open X-Embodiment (5), DROID (19), and BridgeData V2 (11). While these

efforts substantially improve generalist policy learning, their paired language supervision remains sparse: each trajectory is annotated with a goal-level task name that specifies the desired outcome but omits execution details such as which arm to use, how to approach the object, or what motion path to follow.

Fine-grained supervision for manipulation. Several works enrich supervision beyond trajectory-level labels. Galaxea (14), RoboCOIN (17), and RoboInter (26) introduce subtask or hierarchical annotations; STEER (27) and PartInstruct (28) study low-level or part-level instruction following. These annotations are typically organized around stages, primitives, or object parts rather than full process-level descriptions. FineVLA instead provides process-level, action-aligned supervision across a ten-dimensional schema that unifies actor choice, contact patterns, motion trajectories, state transitions, and recovery behavior—and uses this supervision consistently for data construction, VLM training, benchmark evaluation, and policy learning.

Robotic video understanding and scalable annotation. General video-language models such as Qwen3-VL (29) and Qwen3.5-Omni (30) provide strong foundations for video captioning, while embodied benchmarks such as RoboVQA (7), RoboBench (8), and HanDyVQA (9) evaluate spatial reasoning, affordances, and hand-object dynamics. Dense captioning methods such as Wolf (31), DIAL (32), and RoboAnnotatorX (33) further improve annotation scalability. However, general captions do not necessarily align with robot action. FineVLA closes this gap by connecting robotic video understanding directly to VLA policy learning: RoboFine-VLM generates action-aligned descriptions, RoboFine-Bench evaluates execution-level understanding, and FineVLA-Policy tests whether such supervision improves instruction-sensitive control.

Steerable robot foundation models. Recent robot foundation models increasingly emphasize instruction-steerable behavior, where policies should follow not only task goals but also execution-level constraints (1, 2, 3, 4). However, the data construction and evaluation infrastructure behind many such systems remains limited or closed. FineVLA complements these efforts by providing an open action-aligned annotation pipeline, a held-out benchmark, a scalable annotator, and a controlled policy-training study.

7 Conclusion

We presented **FineVLA**, a framework that reframes steerable VLA learning as an action-instruction alignment problem: language supervision should specify not only *what* task to complete, but also the execution-level choices that determine *how* the robot completes it.

Starting from 972,247 trajectories across 10 open-source datasets, FineVLA-Tool produces 47,159 human-verified trajectories with process-level annotations spanning ten fine-grained dimensions. RoboFine-VLM, fine-tuned on this data, serves as a scalable annotator that achieves 71.0% VQA accuracy and 83.6% captioning score on the held-out RoboFine-Bench. FineVLA-Policy, trained under controlled FG:Raw instruction mixtures, reaches **86.8%/82.5%** on AlohaMix-OFT Easy/Hard in RoboTwin simulation, and **62.7**/100 in real-world dual-arm manipulation (vs. 49.9 for Raw-only), with the largest per-factor gains on execution-sensitive attributes such as pose (+23), color (+18), and approach direction (+18).

Two key findings emerge. First, fine-grained supervision does not sacrifice goal-level task success; it consistently improves over raw-only baselines across architectures, data scales, and environments. Second, fine-grained and raw instructions are complementary: the inverted-U trend across all settings shows that the best steerable control comes from mixing both—raw instructions specify *what* to achieve, while fine-grained descriptions specify *how* to execute it. Third, fine-grained supervision directly improves factor-level steerable control: in real-world evaluation, the largest gains appear on execution-sensitive attributes such as color, pose, and approach direction, where goal-level instructions provide no guidance.

We release FineVLA-Tool, RoboFine-VLM, RoboFine-Bench, and FineVLA-Policy checkpoints and training code to support reproducible research on steerable VLA policies. Remaining challenges include compositional generalization to unseen instruction combinations, validation across broader embodiments and task domains, and integrating feasibility and safety checks for fine-grained language following in physical deployment.

References

[1] Physical Intelligence, Bo Ai, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Greg Balke, Kevin Black, George Bokinsky, Shihao Cao, Thomas Charbonnier, Vedant Choudhary, Foster Collins, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Maitrayee Dhaka, Jared DiCarlo, Danny Driess, Michael Equi, Adnan Esmail, Yunhao Fang, Chelsea Finn, Catherine Glossop, Thomas Godden, Ivan Goryachev, Lachlan Groom, Haroun Habeeb, Hunter Hancock, Karol Hausman, Gashon Hussein, Victor Hwang, Brian Ichter, Connor Jacobsen, Szymon Jakubczak, Rowan Jen, Tim Jones, Gregg Kammerer, Ben Katz, Liyiming Ke, Mairbek Khadikov, Chandra Kuchi, Marinda Lamb, Devin LeBlanc, Brendon LeCount, Sergey Levine, Xinyu Li, Adrian Li-Bell, Vladislav Lialin, Zhonglin Liang, Wallace Lim, Yao Lu, Enyu Luo, Vishnu Mano, Nandan Marwaha, Aikys Mongush, Liam Murphy, Suraj Nair, Tyler Patterson, Karl Pertsch, Allen Z. Ren, Gavin Schelske, Charvi Sharma, Baifeng Shi, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, Will Stoeckle, Jiaming Tang, Jimmy Tanner, Shalom Tekeste, Marcel Torne, Kyle Vedder, Quan Vuong, Anna Walling, Haohuan Wang, Jason Wang, XuDong Wang, Chris Whalen, Samuel Whitmore, Blake Williams, Charles Xu, Sukwon Yoo, Lili Yu, Wuming Zhang, Zhuoyang Zhang, and Ury Zhilinsky. $\pi_{0.7}$: a steerable generalist robotic foundation model with emergent capabilities, 2026. URL https://arxiv.org/abs/2604.15483.

[2] Wei Wu, Fan Lu, Yunnan Wang, Shuai Yang, Shi Liu, Fangjing Wang, Qian Zhu, He Sun, Yong Wang, Shuailei Ma, Yiyu Ren, Kejia Zhang, Hui Yu, Jingmei Zhao, Shuai Zhou, Zhenqi Qiu, Houlong Xiong, Ziyu Wang, Zechen Wang, Ran Cheng, Yong-Lu Li, Yongtao Huang, Xing Zhu, Yujun Shen, and Kecheng Zheng. A pragmatic vla foundation model, 2026. URL https://arxiv.org/abs/2601.18692.

[3] NVIDIA. Nvidia isaac gr00t. https://github.com/NVIDIA/Isaac-GR00T, 2026. GitHub repository, accessed April 13, 2026.

[4] Generalist AI. Gen-1. https://generalistai.com/blog/apr-02-2026-GEN-1, 2026. Blog post, accessed April 2, 2026.

[5] Open X-Embodiment Collaboration. Open X-Embodiment: Robotic learning datasets and RT-X models, 2023. URL https://arxiv.org/abs/2310.08864.

[6] Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation, 2025. URL https://arxiv.org/abs/2410.07864.

[7] Pierre Sermanet, Tianli Ding, Jeffrey Zhao, Fei Xia, Debidatta Dwibedi, Keerthana Gopalakrishnan, Christine Chan, Gabriel Dulac-Arnold, Sharath Maddineni, Nikhil J. Joshi, Pete Florence, Wei Han, Robert Baruch, Yao Lu, Suvir Mirchandani, Peng Xu, Pannag Sanketi, Karol Hausman, Izhak Shafran, Brian Ichter, and Yuan Cao. RoboVQA: Multimodal long-horizon reasoning for robotics, 2023. URL https://arxiv.org/abs/2311.00899.

[8] Yulin Luo, Chun-Kai Fan, Menghang Dong, Jiayu Shi, Mengdi Zhao, Bo-Wen Zhang, Cheng Chi, Jiaming Liu, Gaole Dai, Rongyu Zhang, Ruichuan An, Kun Wu, Zhengping Che, Shaoxuan Xie, Guocai Yao, Zhongxia Zhao, Pengwei Wang, Guang Liu, Zhongyuan Wang, Tiejun Huang, and Shanghang Zhang. Robobench: A comprehensive evaluation benchmark for multimodal large language models as embodied brain, 2025. URL https://arxiv.org/abs/2510.17801.

[9] Masatoshi Tateno, Gido Kato, Hirokatsu Kataoka, Yoichi Sato, and Takuma Yagi. HanDyVQA: A video QA benchmark for fine-grained hand-object interaction dynamics, 2025. URL https://arxiv.org/abs/2512.00885.

[10] Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

[11] Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, et al. BridgeData V2: A dataset for robot learning at scale. In *Proceedings of the Conference on Robot Learning*, 2023. URL https://arxiv.org/abs/2308.12952.

[12] Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning, 2022. URL https://arxiv.org/abs/2202.02005.

[13] Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, et al. RT-1: Robotics transformer for real-world control at scale, 2022. URL https://arxiv.org/abs/2212.06817.

[14] Tao Jiang, Tianyuan Yuan, Yicheng Liu, Chenhao Lu, et al. Galaxea open-world dataset and G0 dual-system VLA model, 2025. URL https://arxiv.org/abs/2509.00576.

[15] Kun Wu, Chengkai Hou, Jiaming Liu, Zhengping Che, Xiaozhu Ju, Zhuqin Yang, Meng Li, Yinuo Zhao, Zhiyuan Xu, Guang Yang, Shichao Fan, Xinhua Wang, Fei Liao, Zhen Zhao, Guangyu Li, Zhao Jin, Lecheng Wang, Jilei Mao, Ning Liu, Pei Ren, Qiang Zhang, Yaoxu Lyu, Mengzhen Liu, He Jingyang, Yulin Luo, Zeyu Gao, Chenxuan Li, Chenyang Gu, Yankai Fu, Di Wu, Xingyu Wang, Sixiang Chen, Zhenyu Wang, Pengju An, Siyuan Qian, Shanghang Zhang, and Jian Tang. RoboMIND: Benchmark on multi-embodiment intelligence normative data for robot manipulation. In *Robotics: Science and Systems XXI*, RSS2025. Robotics: Science and Systems Foundation, June 2025. doi: 10.15607/rss.2025.xxi.152. URL http://dx.doi.org/10.15607/RSS.2025.XXI.152.

[16] Chengkai Hou, Kun Wu, Jiaming Liu, Zhengping Che, et al. RoboMIND 2.0: A multimodal, bimanual mobile manipulation dataset for generalizable embodied intelligence, 2025. URL https://arxiv.org/abs/2512.24653.

[17] Shihan Wu, Xuecheng Liu, Shaoxuan Xie, Pengwei Wang, et al. RoboCOIN: An open-sourced bimanual robotic data collection for integrated manipulation, 2025. URL https://arxiv.org/abs/2511.17441.

[18] Hao-Shu Fang, Hongjie Fang, Zhenyu Tang, Jirong Liu, Chenxi Wang, Junbo Wang, Haoyi Zhu, and Cewu Lu. Rh20t: A comprehensive robotic dataset for learning diverse skills in one-shot, 2023. URL https://arxiv.org/abs/2307.00595.

[19] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. DROID: A large-scale in-the-wild robot manipulation dataset, 2024. URL https://arxiv.org/abs/2403.12945.

[20] StarVLA Community. Starvla: A lego-like codebase for vision-language-action model developing, 2026. URL https://arxiv.org/abs/2604.05014.

[21] Yao Mu, Tianxing Chen, Shijia Peng, Zanxin Chen, Zeyu Gao, Yude Zou, Lunkai Lin, Zhiqiang Xie, and Ping Luo. RoboTwin: Dual-arm robot benchmark with generative digital twins, 2024. URL https://arxiv.org/abs/2409.02920.

[22] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control, 2023. URL https://arxiv.org/abs/2307.15818.

[23] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. OpenVLA: An open-source vision-language-action model, 2024. URL https://arxiv.org/abs/2406.09246.

[24] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control, 2024. URL https://arxiv.org/abs/2410.24164.

[25] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy, 2024. URL https://arxiv.org/abs/2405.12213.

[26] Hao Li, Ziqin Wang, Zi-han Ding, Shuai Yang, et al. RoboInter: A holistic intermediate representation suite towards robotic manipulation, 2026. URL https://arxiv.org/abs/2602.09973.

[27] Laura Smith, Alex Irpan, Montserrat Gonzalez Arenas, Sean Kirmani, Dmitry Kalashnikov, Dhruv Shah, and Ted Xiao. STEER: Flexible robotic manipulation via dense language grounding, 2024. URL https://arxiv.org/abs/2411.03409.

[28] Yifan Yin, Zhengtao Han, Shivam Aarya, Jianxin Wang, Shuhang Xu, Jiawei Peng, Angtian Wang, Alan Yuille, and Tianmin Shu. PartInstruct: Part-level instruction following for fine-grained robot manipulation, 2025. URL https://arxiv.org/abs/2505.21652.

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

[30] Qwen Team. Qwen3.5-Omni technical report, 2026. URL https://arxiv.org/abs/2604.15804.

[31] Boyi Li, Ligeng Zhu, Ran Tian, Shuhan Tan, Yuxiao Chen, Yao Lu, Yin Cui, Sushant Veer, Max Ehrlich, Jonah Philion, Xinshuo Weng, Fuzhao Xue, Linxi Fan, Yuke Zhu, Jan Kautz, Andrew Tao, Ming-Yu Liu, Sanja Fidler, Boris Ivanovic, Trevor Darrell, Jitendra Malik, Song Han, and Marco Pavone. Wolf: Dense video captioning with a world summarization framework, 2025. URL https://arxiv.org/abs/2407.18908.

[32] Ted Xiao, Harris Chan, Pierre Sermanet, Ayzaan Wahid, Anthony Brohan, Karol Hausman, Sergey Levine, and Jonathan Tompson. Robotic skill acquisition via instruction augmentation with vision-language models. In *Robotics: Science and Systems*, 2023. URL https://arxiv.org/abs/2211.11736.

[33] Longxin Kou, Fei Ni, Yan Zheng, Peilong Han, Jinyi Liu, Haiqin Cui, Rui Liu, and Jianye Hao. RoboAnnotatorX: A comprehensive and universal annotation framework for accurate understanding of long-horizon robot demonstration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10353–10363, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Kou_RoboAnnotatorX_A_Comprehensive_and_Universal_Annotation_Framework_for_Accurate_Understanding_ICCV_2025_paper.html.