WHAT TO TEACH BEFORE ACTING: REVISITING EMBODIED VQA FOR VLA MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-Language-Action (VLA) models build their policies on pretrained Vision-Language Models (VLMs), and embodied visual question answering (VQA) is widely used to prepare these VLMs for robot control. This paper revisits a practical yet seldom systematically studied question: which embodied VQA supervision actually transfers to downstream VLA policies? We introduce an automatic episode-to-VQA pipeline that turns simulated and real-world robot episodes into VQA, separating what a question teaches, which episodes it comes from, and how its answer is presented. Using this pipeline, we adapt a VLM with 16 generalist, capability-specific, and task-specific VQA recipes and train every adapted backbone under the same VLA protocol. Through closed-loop evaluation under diverse perturbations, we find that VLM benchmark averages are poor predictors of policy success. For instance, cross-view consistency VQA yields the strongest policy despite a lower VLM average than single-view spatial VQA. This challenges the common practice of selecting embodied supervision by VLM benchmarks. We further find that the gains are conditional. Cross-view supervision improves robustness to camera and especially robot-state changes, and much of the net improvement comes from preserving the policy's existing successes. Contrary to intuition, changing only the answer interface from multiple choice to direct generation raises the VLM average but lowers policy success. Finally, VQA built from real-world episodes also improves the policy, and mixing it with simulated VQA outperforms the simulation-only recipe at the same budget. These results indicate that embodied supervision should be chosen by the control conditions it improves and the successes it preserves.

1 INTRODUCTION

Vision-language-action (VLA) models build robot policies on pretrained vision-language models (VLMs), using their visual and semantic knowledge as a prior for control (Brohan et al., 2023, Kim et al., 2024, Black et al., 2024, Physical Intelligence et al., 2025). To strengthen this prior, a growing body of work adapts VLMs with embodied visual question answering (VQA) that teaches spatial relations, localization, interaction regions, and task progression (Chen et al., 2024, Sermanet et al., 2024, Yuan et al., 2024, Song et al., 2025, Chen et al., 2025, BAAI RoboBrain Team et al., 2025, Zhou et al., 2025a, NVIDIA et al., 2025a, Li et al., 2026). However, recent studies find that improving a VLM on such tasks does not guarantee better control (Zhang et al., 2026a, Lin et al., 2026). These studies treat each auxiliary dataset as a fixed whole, leaving open which property of the supervision matters: what a question teaches, which episodes it comes from, or how its answer is presented. This raises a data-construction question: *what should a VLM learn before acting, and when does that supervision improve control?*

Answering this question is difficult for two reasons. First, supervision choices are entangled in existing datasets. When placing a cup on a tray, localizing the cup, identifying its handle, and indicating its destination all require a point answer, yet they teach different targets. Conversely, the same target can be asked as a multiple-choice question or as a free-form answer. Second, a policy acts under changes in viewpoint, appearance, and layout, and a recipe can improve some of these conditions while losing successes in others. A useful study therefore needs supervision whose target, source, and interface are explicit, and an evaluation that resolves conditions and tracks which successes are gained or lost.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

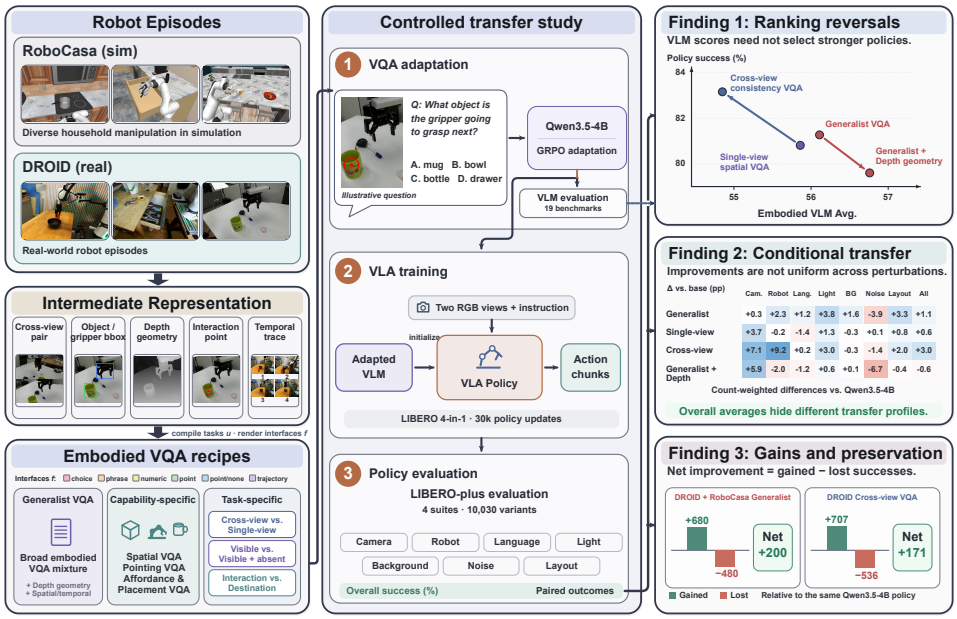

Figure 1: **Study overview and observed transfer patterns.** Simulated and real-world episodes are compiled into embodied VQA recipes (left), which adapt Qwen3.5-4B before common VLA training and LIBERO-plus evaluation (center). The findings (right) use RoboCasa recipes, except Finding 3, which uses DROID recipes (Sec. 5.5).

To provide such supervision, we build an automatic episode-to-VQA pipeline (Fig. 1). Source-specific extractors turn simulated RoboCasa (Nasiriany et al., 2026) and real-world DROID (Khazatsky et al., 2024) episodes into a shared evidence graph of objects, gripper states, views, and manipulation phases. Task compilers execute each task on this graph to obtain a target, and interfaces render the question together with a verifier that later serves as the GRPO (Shao et al., 2024) reward. The same compilers run unchanged on both sources, so simulated and real episodes yield VQA through one construction path.

Based on this pipeline, we run a controlled study. We adapt Qwen3.5-4B (Qwen Team, 2026a) with 16 VQA recipes through GRPO, train a policy on each adapted backbone under the same VLA protocol, and evaluate every policy on LIBERO-plus (Fei et al., 2025) under diverse perturbations, matching outcomes variant by variant. The study asks three questions about embodied supervision:

- **What to teach.** Does a broad mix of questions help more than supervision focused on one capability, and do finer choices within a capability, such as how many views a question uses or what a point marks, change the policy?
- **Where the episodes come from.** Can VQA built from real-world recordings support the policy as well as VQA built from simulation?
- **How the question is asked.** Does the answer format matter when the question targets the same thing?

Our findings reveal three aspects of transfer. First, VLM averages do not reliably rank policies: Cross-view consistency VQA yields the strongest policy despite a lower VLM average than Single-view spatial VQA, and direct generation raises the VLM average of cross-view supervision while lowering policy success. Second, gains are conditional. Camera and robot-state perturbations account for most baseline failures, and Cross-view consistency VQA improves both in every suite, whereas Generalist VQA leads the language, lighting, background, and layout conditions. Third, much of the net improvement comes from preserving baseline successes. Recipes that solve a similar number of failures differ mainly in how many successes they lose. Across sources, VQA built from real DROID episodes also improves the policy, and mixing it with RoboCasa questions outperforms the RoboCasa-only recipe at the same budget. Taken together, these results indicate that

embodied supervision should be selected by the control conditions it improves and the successful behavior it retains.

Our contributions are threefold. **(1)** We build an automatic and scalable episode-to-VQA pipeline that separates reusable evidence, task definitions, and answer interfaces, and runs on both simulated and real-world episodes. **(2)** We conduct a controlled study of embodied VQA along what is taught, where the episodes come from, and how questions are asked, under a common VLA training and evaluation protocol. **(3)** We provide a variant-level analysis showing that VLM averages do not reliably rank policies and that net gains depend on preserved successes.

2 Related Work

VLMs for robot control. VLA models adapt vision-language pretraining to action prediction through discrete action tokens, as in RT-2 (Brohan et al., 2023) and OpenVLA (Kim et al., 2024), or flow matching, as in π_0 (Black et al., 2024), with efficient tokenizers such as FAST (Pertsch et al., 2025) and diffusion-transformer action heads such as GR00T N1 (NVIDIA et al., 2025b). RoboFlamingo (Li et al., 2023) shows that pretrained VLMs are effective robot imitators, and RoboVLMs (Li et al., 2024) compares VLM backbones and policy designs. Beyond action data, $\pi_{0.5}$ (Physical Intelligence et al., 2025) and ChatVLA (Zhou et al., 2025b) co-train policies with vision-language data, knowledge insulation (Driess et al., 2025) protects the VLM from action-head gradients, and ECoT (Zawalski et al., 2024) reasons through intermediate embodied steps before acting. VLM4VLA (Zhang et al., 2026a) finds that VLM benchmark strength and auxiliary embodied tasks do not guarantee better control, and Rethinking VLM Representation (Lin et al., 2026) stresses preserving useful pretrained representations during adaptation. VLAct (Yang et al., 2026a) and PRTS (Zhang et al., 2026b) shape these representations with action supervision and goal-reachability objectives. We instead vary upstream embodied VQA under fixed action learning and relate each recipe to gained and preserved policy successes.

Embodied data and intermediate representations. SpatialVLM (Chen et al., 2024) lifts internet images into 3D to generate spatial VQA, SAT (Ray et al., 2025) renders dynamic spatial questions in simulation, RoboPoint (Yuan et al., 2024) synthesizes spatial affordance data, RoboSpatial (Song et al., 2025) makes reference frames explicit, RoboRefer (Zhou et al., 2025a) trains spatial referring with reasoning, and Cross-View Relations (Jeong et al., 2026) targets multi-view reasoning. Closer to robot episodes, RoboVQA (Sermanet et al., 2024) annotates long-horizon robot videos, Robo2VLM (Chen et al., 2025) generates questions from robot signals and manipulation phases in Open X-Embodiment (Open X-Embodiment Collaboration et al., 2023), and RoboInter (Li et al., 2026) provides intermediate representations with spatial and temporal VQA. Embodied VLMs such as RoboBrain (Ji et al., 2025, BAAI RoboBrain Team et al., 2025) and Cosmos-Reason1 (NVIDIA et al., 2025a) combine such data at scale. StellaVLA (Xu et al., 2026) instead retrieves structured demonstrations as policy context. Our construction separates evidence, target, and answer interface (Sec. 3) so that these supervision choices can be compared as initializations of one action model.

3 Episode-to-VQA Construction

The construction separates what an episode records, what a question supervises, and how its answer is presented (Fig. 1, left). Following the frame-level view of robot data in Robo2VLM (Chen et al., 2025), an episode from source s is $e_s = (\ell, \{O_t, r_t, a_t\}_{t=1}^{T}, \kappa)$ with instruction ℓ, synchronized views $O_t = \{I_t^v\}_{v \in \mathcal{V}}$, robot signals r_t, actions a_t, and calibration and provenance κ. Construction applies three functions:

$$\mathcal{G}_e = F_s(e_s), \qquad (x,\, c,\, z) = C_u(\mathcal{G}_e), \qquad (q,\, \rho) = R_f(u,\, c,\, z), \tag{1}$$

where the input satisfies $x \subseteq \{I_t^v : t \le t_u\}$ and C_u is defined only when $\mathcal{G}_e$ supports the target z. Each training item is (x, q, ρ). As in executing a functional program on a scene graph (Johnson et al., 2017), a task u is executed on the evidence graph $\mathcal{G}_e$ to obtain a format-independent target z and question content c. The interface f then determines how the question q is posed and how the verifier ρ scores an answer.

Source-specific evidence graphs. Each source needs its own extractor F_s. In RoboCasa (Nasiriany et al., 2026), the simulator already records object and gripper geometry, task state, and synchro-

Table 1: **VQA supervision recipes.** Indented rows are splits or variants. Interfaces f: choice, phrase, numeric, point, point or "No object," trajectory. One example question is shown per recipe (all templates in Appendix C.3). Episodes are from RoboCasa unless DROID is named, and * marks supplement data.

Recipe	Example question	f	#VQA
Reference			
Generalist VQA	*Equal mix of nine question types*		18,023
DROID + RoboCasa Generalist	*Same mix, mostly from DROID episodes*		18,023
+ Depth Geometry*	Which point is closer to the camera?		+2,000
+ Spatial/Temporal Data*	Trace the arm trajectory for the task.		+5,112
Spatial			
Spatial VQA*	*Mix of the two spatial splits below*		4,600
Cross-view consistency VQA	Across two views, which subgoal is underway?		2,000
Single-view spatial VQA*	Where is the gripper relative to the target?		2,000
Cross-view direct generation	Same questions, answered without options		2,000
DROID Cross-view VQA	Same questions on DROID camera pairs		2,000
Pointing			
Pointing VQA*	*Pointing, counting, and presence mix*		5,400
Visible object pointing VQA	Point to the visible {object}.		1,600
Visible + absent object pointing VQA	Point to each visible {object}, or "No object."		1,600
Affordance & Placement			
Affordance & Placement VQA*	*Mix of the two point splits below*		4,200
Interaction-point grounding VQA	Point to where the gripper interacts to {task}.		1,620
Destination-point grounding VQA	Point to where the object should be placed.		1,620

nized views, so the extractor reads them directly. DROID (Khazatsky et al., 2024) provides only raw recordings, so the extractor recovers the same information with perception. A VLM identifies the task object, SAM 3 (Carion et al., 2025) tracks the object and gripper, the gripper signal marks manipulation phases, and the synchronized cameras feed a separate cross-view branch. Both extractors write to the same schema, in which every entity carries its identity, time, coordinate frame, and evidence status. The graph can therefore keep an object's center, the point where the gripper touched it, and the place where it was set down as separate entries, even though a question about any of them is answered with a single point. These graphs are used only to construct VQA and never enter the policy input. Because simulator records give exact targets, we build the controlled comparisons from RoboCasa. DROID then tests whether the same supervision can be built from estimated evidence (Sec. 5.5). The task compilers run on DROID graphs unchanged and produce most Generalist question types. Questions that need several tracked objects or long-horizon plans still come from RoboCasa.

Tasks and interfaces. A task u decides what a question asks about: which views and frames the model may see, which entity or event is the target, and in which coordinate frame the answer is given. The interface f decides how the question is posed and answered. The two vary independently. "Point to the visible pear," "Open the dishwasher door," and "Place the cheese in the stand mixer bowl" all ask for a single point, yet they teach where an object is, where to interact with it, and where to put it. In the other direction, the same target can take different answer forms: a choice among options, a short phrase, an image coordinate, an ordered trajectory, or a coordinate that may instead be "No object" when the target is absent. Keeping u and f separate lets us change what is taught while holding the answer form fixed, or change the answer form while holding the target fixed. Table 1 lists the tasks and interfaces used by each recipe.

Verification and reuse. Every question comes with a verifier ρ that scores a model's answer, and this score is used directly as the GRPO reward (Sec. 4.2). The verifier first checks that the answer is well formed, with valid syntax, coordinates inside the image, and the expected number of points, and then compares it with the target. We keep only questions that the base model solves some of the time, so every question gives a useful training signal. Finally, because evidence graphs are cached,

a single extraction yields many tasks and interfaces without labeling questions by hand. Details of each stage appear in Appendices A and B.

4 Study Design

4.1 Comparisons

Table 1 lists what each recipe contains. This section explains how the recipes are paired to answer the three questions from the introduction. Every recipe starts from Qwen3.5-4B (Qwen Team, 2026a), which also serves as the reference policy.

What to teach. We first compare Generalist VQA with the three capability families to see whether focusing on one capability beats a broad mix. We then split each family into two recipes with the same size, difficulty, and GRPO settings. Cross-view consistency VQA and Single-view spatial VQA differ in whether a question needs two synchronized views or a single image. The two pointing recipes differ in whether the model is also taught to answer "No object." The two affordance recipes both answer with a point but differ in what the point marks, either where the gripper interacts or where the object goes. Results appear in Secs. 5.1–5.4.

Where the episodes come from. We rebuild Generalist VQA and Cross-view consistency VQA from DROID episodes at the same size and compare each with its RoboCasa version (Sec. 5.5).

How the question is asked. We remove the answer options from Cross-view consistency VQA and ask for the answer directly, leaving the questions otherwise unchanged (Sec. 5.6).

Depth Geometry and Spatial/Temporal Data add extra depth or trajectory questions to Generalist VQA. Depth Geometry also serves as a contrast in Sec. 5, and both are detailed in Appendix J. Data counts for every recipe are in Appendix C.

4.2 GRPO Adaptation and Verifier Rewards

Each recipe adapts the same pretrained Qwen3.5-4B checkpoint with full-parameter GRPO (Shao et al., 2024) under one shared configuration. Sampled responses are scored by the item's verifier ρ (Eq. 1), which rewards correct and well-formed answers and gives bounded partial credit to continuous predictions (Appendix D.1).

4.3 Common Policy Training

Every adapted VLM is turned into a policy in the same way. We use the QwenGR00T policy from StarVLA (StarVLA Community, 2026), which follows GR00T (NVIDIA et al., 2025b) and attaches a DiT-B action head to the VLM. The VLM, the interface between them, and the action head are trained jointly for 30,000 steps on LIBERO 4-in-1 (Liu et al., 2023), which contains 1,693 episodes and 273,465 frames across 40 tasks. At each step the policy sees agent-view and wrist-view images at 224×224 together with the instruction, and predicts a chunk of eight 7D actions. It receives no proprioception at evaluation. We train every policy with two random seeds and report the mean. Architecture and optimizer details appear in Appendix D.

4.4 Evaluation Protocol

Policy evaluation. Each policy is evaluated on all 10,030 LIBERO-plus variants, which cover the Goal, Spatial, Object, and LIBERO-10 suites under seven categories of perturbation. We report the success rate averaged over the two seeds, broken down by suite and by perturbation category.

VLM evaluation. We evaluate each adapted VLM with thinking disabled so that it answers directly. Our summary score, Embodied VLM Avg., is the unweighted mean of 19 EmbodiedEvalKit (Yuan et al., 2026b) benchmarks covering spatial understanding, pointing and affordance, trajectory prediction, and embodied reasoning and planning. The two trajectory benchmarks are scored only on valid outputs. Each benchmark and its source are listed in Appendix E.2, and full per-benchmark results for the adapted VLMs, including CV-Bench and RoboVQA, appear in Appendix F.

Table 2: **Results on LIBERO-plus.** Published expert VLA results are shown for comparison. Our Qwen3.5-4B policies share the same training and evaluation settings.

	LIBERO-plus evaluation							
Model	**Language**	**Camera**	**Robot**	**Light**	**Background**	**Noise**	**Layout**	**Overall**
Expert VLA models								
π_0 (Black et al., 2024)	58.8	13.8	6.0	85.0	81.4	79.0	68.9	53.6
OpenVLA-OFT (Kim et al., 2025)	79.5	56.4	31.9	88.7	93.3	75.8	74.2	69.6
π_0-FAST (Pertsch et al., 2025)	61.0	65.1	21.6	73.2	73.2	74.4	68.8	61.6
StarVLA (StarVLA Community, 2026)	88.5	52.5	49.8	95.7	95.7	73.0	76.9	74.1
ABot-M0 (Yang et al., 2026b)	86.4	60.4	<u>67.9</u>	96.2	91.6	86.4	<u>82.6</u>	80.5
Cosmos-Policy (Kim et al., 2026)	81.7	**75.8**	63.3	96.5	88.9	**92.7**	82.2	<u>82.2</u>
Reference initializations								
Qwen3.5-4B	92.3	60.7	59.6	94.4	96.2	88.0	79.4	80.2
+ Generalist VQA	**93.4**	61.0	61.9	**98.2**	**97.8**	84.1	**82.8**	81.3
Capability-specific VQA								
+ Spatial VQA	92.7	62.0	63.7	94.7	<u>97.2</u>	81.0	82.2	80.6
+ Pointing VQA	92.8	65.9	60.3	95.9	96.0	86.8	80.5	81.4
+ Affordance & Placement VQA	92.9	65.9	65.9	96.9	95.8	82.1	81.2	81.7
Spatial VQA splits								
+ Cross-view consistency VQA	92.5	<u>67.8</u>	**68.8**	<u>97.4</u>	95.9	86.5	81.4	**83.2**
+ Single-view spatial VQA	90.9	64.4	59.4	95.7	95.9	<u>88.1</u>	80.2	80.8
Pointing VQA splits								
+ Visible object pointing VQA	<u>93.1</u>	65.5	62.1	95.5	97.0	83.8	79.4	81.1
+ Visible + absent object pointing VQA	90.2	58.7	64.7	96.8	95.3	85.5	80.9	80.4
Affordance & Placement VQA splits								
+ Interaction-point grounding VQA	91.4	64.2	60.1	95.8	96.4	84.1	80.7	80.5
+ Destination-point grounding VQA	91.4	60.6	62.3	95.8	95.0	84.9	79.2	80.0

5 Experiments and Analysis

We first examine what to teach using the RoboCasa recipes, covering overall success, VLM–policy agreement, conditional transfer, and preserved successes (Secs. 5.1–5.4). We then turn to where the episodes come from (Sec. 5.5) and how the question is asked (Sec. 5.6).

5.1 Effect of VQA Supervision on Policy Performance

Supervision choice matters beyond capability labels. Table 2 reports every policy, with published expert VLAs for reference. Nearly every VQA recipe in Table 2 improves on Qwen3.5-4B. Generalist VQA gives a moderate gain of 1.07%, and the three capability families land close to it. Cross-view consistency VQA stands out with the highest overall success, 83.16%, a gain of 2.96%, and improves all four suites. It also exceeds every published expert VLA in Table 2, including Cosmos-Policy at 82.2%. All of our policies outperform the expert VLAs under language perturbations, while Cosmos-Policy remains strongest under camera and noise perturbations. Specialization is not uniformly better, however. Cross-view beats both Generalist VQA and its own family pool, whereas neither the pointing nor the affordance splits beat theirs. Recipe size does not explain the ranking either. The small task splits include both the best policy, Cross-view consistency VQA, and the only recipe in Table 2 that falls below Qwen3.5-4B, Destination-point grounding VQA. Sections 5.3 and 5.4 trace this gap to where each recipe gains and how many existing successes it retains. Suite-level results appear in Appendix H.

Overall gains address the largest baseline deficits. Camera and Robot perturbations make up 31% of the variants but 63% of the failures of Qwen3.5-4B. Cross-view improves on both and reaches the highest Robot score of all policies, 68.8%, so most of its overall gain comes from these two categories. On the other five categories it performs close to Generalist VQA. Its advantage therefore comes from fixing the conditions where the baseline fails most.

5.2 Relationship Between VLM Evaluation and Policy Performance

VLM averages do not reliably pick the stronger policy. In Fig. 2, moving from Single-view spatial VQA to Cross-view consistency VQA (blue arrow) lowers the VLM average from 55.86 to 54.85 but raises policy success from 80.81% to 83.16%. Adding Depth Geometry to Generalist VQA (red arrow) does the opposite, raising the VLM average from 56.11 to 56.76 while policy success drops from 81.27% to 79.60%. Choosing supervision by VLM average would therefore favor Single-view over Cross-view and add Depth Geometry to Generalist VQA, and both choices would weaken the policy. Both reversals hold under the full 21-benchmark average, and the Cross-view one survives removing any single benchmark (Appendix G). The same pattern appears when the answer interface (Sec. 5.6) or the episode source (Sec. 5.5) changes. The mismatch is not a simple inverse relation, however. For the pointing and affordance pairs (grey arrows), the recipe with the higher VLM average also gives the stronger policy. Benchmark-level links appear in Appendix I.

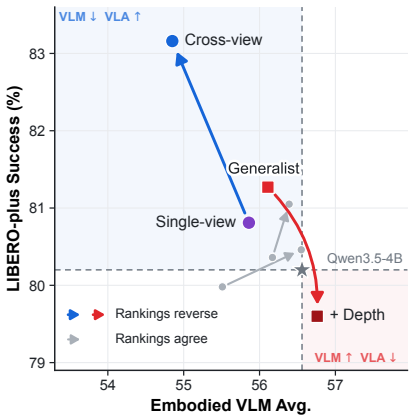

Figure 2: **Comparison of embodied VLM capability and VLA performance.**

5.3 Analysis of Transfer Across Tasks and Perturbations

Aggregate success rates can mask where a recipe is effective. Figure 3 therefore decomposes each within-family comparison by task suite and perturbation category, with positive (blue) values favoring the first recipe named in each panel.

Cross-view supervision benefits robot-state robustness most. Cross-view consistency VQA outperforms Single-view spatial VQA under robot-state perturbations in all four suites (Fig. 3a), with a margin of 9.42% overall and up to 13.7% on Goal. Its advantage under camera perturbations is smaller, at 3.38%. The benefit of multi-view supervision is therefore concentrated on changes to the robot's initial state.

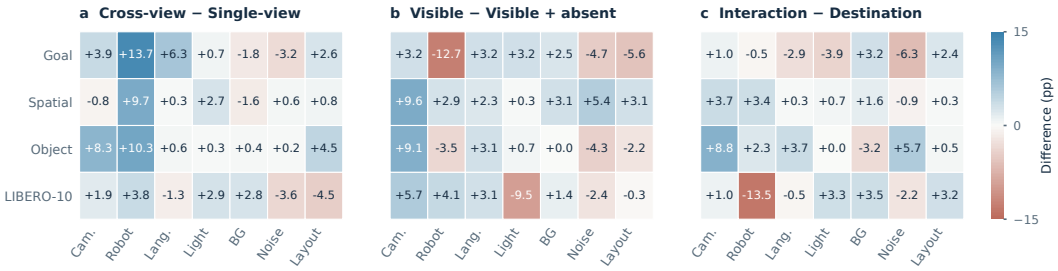

Figure 3: **Task-conditional transfer.** Each cell is the difference in success rate (%) between the two recipes in the panel title.

Comparable overall success can conceal opposing effects. The two pointing recipes differ by less than 1% overall. Including absence questions reduces camera robustness consistently across suites, by 6.82% in aggregate, while improving Goal under robot-state perturbations by 12.7% (Fig. 3b). Interaction-point and Destination-point grounding VQA differ by only 0.48% overall (Fig. 3c). Yet the former leads by 8.8% on the Object suite under camera perturbations, and the latter leads by 13.5% on LIBERO-10 under robot-state perturbations. A shared point interface thus does not imply shared transfer when the supervised targets differ. Full suite breakdowns appear in Appendix H.1.

Adaptation shifts robustness in consistent directions. Across the ten RoboCasa recipes in Table 2, all improve Light over Qwen3.5-4B and eight improve Camera, whereas nine lose successes under Noise. The distribution of gains depends on the supervision. Generalist VQA leads on Language, Light, Background, and Layout but leaves Camera nearly unchanged, while the Pointing and

Affordance & Placement families each raise Camera by about 5%. Adding Depth Geometry or Spatial/Temporal Data to Generalist VQA raises success on LIBERO-10 under camera perturbations by 10.0% and 8.6%, respectively, relative to Generalist VQA trained under the same reasoning setting, but lowers overall success (Appendices J and K).

5.4 Analysis of Gained and Preserved Successes

Because every policy is evaluated on the same variants, each outcome can be paired with that of Qwen3.5-4B. A *gained success* is a variant that the reference fails and the policy solves, and a *lost success* is the reverse. Figure 4 plots both counts for each recipe.



Figure 4: **Successes gained and lost** relative to Qwen3.5-4B, with compared recipes in adjacent rows.

Recipes differ more in what they lose than in what they gain. Apart from Cross-view, every recipe in Fig. 4 gains between 588 and 625 successes, yet losses range from 517 to 677. Within the pointing and affordance pairs, the gained bars are nearly identical, and fewer losses account for 68 of the 69-success and 44 of the 48-success gaps. Cross-view consistency VQA is the exception, gaining 159 more successes than Single-view spatial VQA and losing 77 fewer.

Depth gains can be offset by lost successes. Adding Depth Geometry to Generalist VQA leaves the gained bar almost unchanged but lengthens the lost bar. Relative to Generalist VQA, it gains 509 successes but loses 676, mostly outside camera perturbations, for a net decline of 167. Whether additional supervision helps can therefore depend as much on the successes it retains as on those it adds. Detailed alignments appear in Appendices K and L.

5.5 Supervision from Simulated and Real Episodes

We next ask whether VQA built from real-world recordings transfers as well as VQA built from simulation. Figure 5 compares each RoboCasa recipe with a counterpart built from DROID episodes at the same question budget. Both sources improve the policy over Qwen3.5-4B.

Real-episode supervision transfers, with different trade-offs. The DROID + RoboCasa Generalist reaches 82.19% overall, above both the RoboCasa-only Generalist (81.27%) and Qwen3.5-4B (80.20%). Its gains over the backbone come mainly from the Goal and Object suites and from camera and layout perturbations, and it also preserves robustness to sensor noise, reaching 88.07% against 84.07% for the RoboCasa-only recipe. DROID Cross-view reaches 81.90% overall and the highest Camera score of all our policies, 69.42%, but is weaker under noise (83.51%). The paired

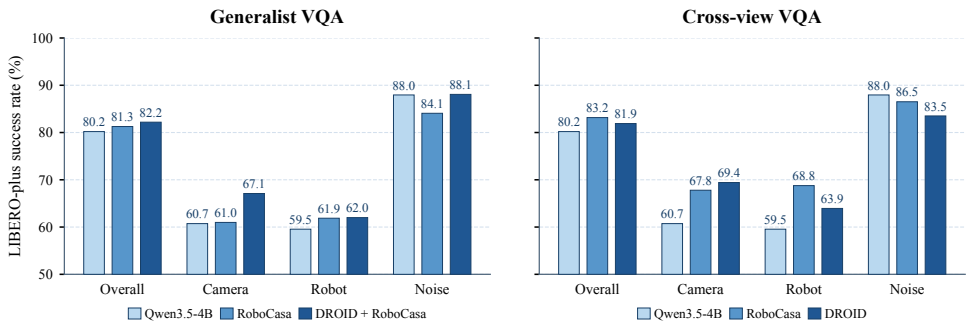

Figure 5: **Performance of VQA from simulated vs. real-world episodes.** Generalist VQA is built from RoboCasa alone or from a DROID + RoboCasa mixture, and Cross-view VQA from RoboCasa or DROID alone. All seven perturbation categories are reported in Table 25 (Appendix M).

outcomes follow the pattern of Sec. 5.4. The DROID + RoboCasa Generalist gains 680 successes and loses 480, whereas DROID Cross-view gains 707 but loses 536, so the Generalist's larger net gain comes from losing fewer successes (Fig. 1, Finding 3).

VLM averages again reverse the policy ordering. The DROID + RoboCasa Generalist has a lower VLM average than the RoboCasa-only Generalist (55.51 versus 56.11) but yields the stronger policy, whereas DROID Cross-view has a higher VLM average than RoboCasa Cross-view (55.45 versus 54.85) but yields the weaker one.

5.6 Cross-view VQA Supervision Interfaces

Removing the answer options from Cross-view consistency VQA, which also changes the answer target and its verifier, raises the VLM average from 54.85 to 55.53 but lowers policy success from 83.16% to 80.87% (Table 3). The drop comes mainly from robot-state perturbations, where direct generation trails MCQ by 8.19%, accounting for over half of the overall gap. Conversion details and full profiles appear in Appendix L.

Table 3: **Cross-view VQA supervision interfaces.** MCQ presents options for selection, whereas direct generation requires an answer phrase without options.

Qwen3.5-4B	VLM Avg.↑	Qwen3.5-4B	LIBERO-plus↑
Baseline	56.56	Baseline	80.20
Cross-view **MCQ**	54.85 (-1.71)	Cross-view **MCQ**	83.16 ($+2.96$)
Cross-view **direct gen.**	55.53 (-1.03)	Cross-view **direct gen.**	80.87 ($+0.67$)

6 Discussion

These results point to three practical lessons for preparing VLMs with embodied VQA. VLM benchmark averages are an unreliable guide to which supervision will help control, as they reversed the policy ranking in every source and interface comparison we examined. Supervision is better chosen with the target operating conditions in mind, since different recipes strengthen different ones, with cross-view questions favoring robot-state changes and Generalist VQA favoring language and appearance changes. A recipe should also be judged by the successes it preserves as well as those it adds, because preservation is what separated recipes that solved similar numbers of failures. As the same construction applies to both simulated and real-world episodes, these lessons can be tested at larger scale on real-robot corpora and with other backbones and policy frameworks. Appendix N discusses data selection and diagnostic profiles in more detail.

Limitations. The construction can pose only questions that its source records support. Real recordings do not yet provide the evidence needed for multi-instance and long-horizon questions, and perception-backed annotations are bounded by the accuracy of current perception models, a gap that should narrow as those models improve. VQA also teaches only what can be phrased as a question with a verifiable answer. Behaviors that are difficult to verify from images, such as contact forces or precise timing, will require complementary forms of supervision.

7 Conclusion

We examined which embodied VQA supervision transfers to VLA policies by varying what a question teaches, which episodes it is built from, and how it is asked. Along all three dimensions, VLM averages failed to reliably rank the resulting policies. Transfer instead depended on the operating condition, as different recipes strengthened different perturbation types, and on how many existing successes a recipe preserved while it solved new failures. VQA built from real-world episodes also improved the policy, showing that the same construction extends beyond simulation. Embodied supervision is therefore best selected by the conditions it improves and the successes it retains, not by VLM scores alone. We hope the framework and these findings encourage future work to treat VLM adaptation as part of policy design and to judge it by the behavior of the resulting policy.

AI USE STATEMENT

Generative AI tools, including ChatGPT and Codex, assisted with literature organization, manuscript drafting and revision, exploratory interpretation of the supplied experiment reports, analysis and plotting code, and figure preparation. Figure 1 is a vector schematic of the construction framework and empirical study. Numerical analyses use the recorded experiment results. The authors reviewed all AI-assisted content and take full responsibility for the paper, including its text, claims, and results.

REFERENCES

BAAI RoboBrain Team, M. Cao, H. Tan, et al. RoboBrain 2.0 technical report. arXiv:2507.02029, 2025. URL https://arxiv.org/abs/2507.02029.

K. Black et al. π_0: A vision-language-action flow model for general robot control. arXiv:2410.24164, 2024. URL https://arxiv.org/abs/2410.24164.

A. Brohan et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. arXiv:2307.15818, 2023. URL https://arxiv.org/abs/2307.15818.

N. Carion et al. SAM 3: Segment Anything with Concepts. arXiv:2511.16719, 2025. URL https://arxiv.org/abs/2511.16719.

B. Chen, Z. Xu, S. Kirmani, et al. SpatialVLM: Endowing vision-language models with spatial reasoning capabilities. CVPR, 2024. URL https://arxiv.org/abs/2401.12168.

K. Chen, S. Xie, Z. Ma, P. R. Sanketi, and K. Goldberg. Robo2VLM: Visual question answering from large-scale in-the-wild robot manipulation datasets. arXiv:2505.15517, 2025. URL https://arxiv.org/abs/2505.15517.

L. Cheng, J. Duan, Y. R. Wang, et al. PointArena: Probing multimodal grounding through language-guided pointing. arXiv:2505.09990, 2025. URL https://arxiv.org/abs/2505.09990.

M. Deitke, C. Clark, S. Lee, et al. Molmo and PixMo: Open weights and open data for state-of-the-art vision-language models. arXiv:2409.17146, 2024. URL https://arxiv.org/abs/2409.17146.

D. Driess, J. T. Springenberg, B. Ichter, et al. Knowledge insulating vision-language-action models: Train fast, run fast, generalize better. arXiv:2505.23705, 2025. URL https://arxiv.org/abs/2505.23705.

M. Du, B. Wu, Z. Li, X. Huang, and Z. Wei. EmbSpatial-Bench: Benchmarking spatial understanding for embodied tasks with large vision-language models. ACL, 2024. URL https://arxiv.org/abs/2406.05756.

S. Fei et al. LIBERO-Plus: In-depth robustness analysis of vision-language-action models. arXiv:2510.13626, 2025. URL https://arxiv.org/abs/2510.13626.

X. Fu, Y. Hu, B. Li, et al. BLINK: Multimodal large language models can see but not perceive. ECCV, 2024. URL https://arxiv.org/abs/2404.12390.

Gemini Robotics Team, S. Abeyruwan, J. Ainslie, et al. Gemini Robotics: Bringing AI into the physical world. arXiv:2503.20020, 2025. URL https://arxiv.org/abs/2503.20020.

X. Hao, Y. Tang, L. Zhang, et al. RoboAfford++: A generative AI-enhanced dataset for multimodal affordance learning in robotic manipulation and navigation. arXiv:2511.12436, 2025. URL https://arxiv.org/abs/2511.12436.

S. Jeong et al. Learning multi-view spatial reasoning from cross-view relations. arXiv:2603.27967, 2026. URL https://arxiv.org/abs/2603.27967.

Y. Ji, H. Tan, J. Shi, et al. RoboBrain: A unified brain model for robotic manipulation from abstract to concrete. arXiv:2502.21257, 2025. URL https://arxiv.org/abs/2502.21257.

J. Johnson, B. Hariharan, L. van der Maaten, L. Fei-Fei, C. L. Zitnick, and R. Girshick. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. CVPR, 2017. URL https://arxiv.org/abs/1612.06890.

A. Khazatsky et al. DROID: A large-scale in-the-wild robot manipulation dataset. Robotics: Science and Systems, 2024. URL https://arxiv.org/abs/2403.12945.

M. J. Kim, C. Finn, and P. Liang. Fine-tuning vision-language-action models: Optimizing speed and success. arXiv:2502.19645, 2025. URL https://arxiv.org/abs/2502.19645.

M. J. Kim, Y. Gao, T.-Y. Lin, et al. Cosmos Policy: Fine-tuning video models for visuomotor control and planning. arXiv:2601.16163, 2026. URL https://arxiv.org/abs/2601.16163.

M. J. Kim et al. OpenVLA: An open-source vision-language-action model. arXiv:2406.09246, 2024. URL https://arxiv.org/abs/2406.09246.

J. Lee, J. Duan, H. Fang, et al. MolmoAct: Action reasoning models that can reason in space. arXiv:2508.07917, 2025. URL https://arxiv.org/abs/2508.07917.

H. Li et al. RoboInter: A holistic intermediate representation suite towards robotic manipulation. ICLR, 2026. URL https://arxiv.org/abs/2602.09973.

X. Li, M. Liu, H. Zhang, et al. Vision-language foundation models as effective robot imitators. arXiv:2311.01378, 2023. URL https://arxiv.org/abs/2311.01378.

X. Li, P. Li, L. Qian, et al. What matters in building vision-language-action models for generalist robots. arXiv:2412.14058, 2024. URL https://arxiv.org/abs/2412.14058.

W. Lin et al. Rethinking VLM representation for VLA initialization. arXiv:2605.25802, 2026. URL https://arxiv.org/abs/2605.25802.

B. Liu et al. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. arXiv:2306.03310, 2023. URL https://arxiv.org/abs/2306.03310.

R. Liu, Y. Chen, Y. Zhang, et al. What if? Emulative simulation with world models for situated reasoning. arXiv:2603.06445, 2026. URL https://arxiv.org/abs/2603.06445.

Y. Liu, B. Zhang, Y. Zang, et al. Spatial-SSRL: Enhancing spatial understanding via self-supervised reinforcement learning. arXiv:2510.27606, 2025. URL https://arxiv.org/abs/2510.27606.

Y. Lu, Y. Fan, B. Deng, F. Liu, Y. Li, and S. Wang. VL-Grasp: A 6-DoF interactive grasp policy for language-oriented objects in cluttered indoor scenes. arXiv:2308.00640, 2023. URL https://arxiv.org/abs/2308.00640.

S. Nasiriany, S. Nasiriany, A. Maddukuri, and Y. Zhu. RoboCasa365: A large-scale simulation framework for training and benchmarking generalist robots. ICLR, 2026. URL https://arxiv.org/abs/2603.04356.

NVIDIA, A. Azzolini, J. Bai, et al. Cosmos-Reason1: From physical common sense to embodied reasoning. arXiv:2503.15558, 2025a. URL https://arxiv.org/abs/2503.15558.

NVIDIA, J. Bjorck, F. Castañeda, et al. GR00T N1: An open foundation model for generalist humanoid robots. arXiv:2503.14734, 2025b. URL https://arxiv.org/abs/2503.14734.

Open X-Embodiment Collaboration, A. O'Neill, A. Rehman, et al. Open X-Embodiment: Robotic learning datasets and RT-X models. arXiv:2310.08864, 2023. URL https://arxiv.org/abs/2310.08864.

K. Pertsch, K. Stachowicz, B. Ichter, et al. FAST: Efficient action tokenization for vision-language-action models. arXiv:2501.09747, 2025. URL https://arxiv.org/abs/2501.09747.

Physical Intelligence, K. Black, N. Brown, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. arXiv:2504.16054, 2025. URL https://arxiv.org/abs/2504.16054.

L. Qiu, Y. Chen, Y. Ge, Y. Ge, Y. Shan, and X. Liu. EgoPlan-Bench2: A benchmark for multimodal large language model planning in real-world scenarios. arXiv:2412.04447, 2024. URL https://arxiv.org/abs/2412.04447.

Qwen Team. Qwen3.5-4B, 2026a. URL https://huggingface.co/Qwen/Qwen3.5-4B.

Qwen Team. Qwen3.8-27B, 2026b. URL https://huggingface.co/Qwen/Qwen3.8-27B.

A. Ray, J. Duan, E. Brown, et al. SAT: Dynamic spatial aptitude training for multimodal language models. COLM, 2025. URL https://arxiv.org/abs/2412.07755.

P. Sermanet, T. Ding, J. Zhao, et al. RoboVQA: Multimodal long-horizon reasoning for robotics. ICRA, 2024. URL https://arxiv.org/abs/2311.00899.

Z. Shao et al. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. arXiv:2402.03300, 2024. URL https://arxiv.org/abs/2402.03300.

C. H. Song, V. Blukis, J. Tremblay, S. Tyree, Y. Su, and S. Birchfield. RoboSpatial: Teaching spatial understanding to 2D and 3D vision-language models for robotics. CVPR, 2025. URL https://arxiv.org/abs/2411.16537.

StarVLA Community. StarVLA: A Lego-like codebase for vision-language-action model developing. arXiv:2604.05014, 2026. URL https://arxiv.org/abs/2604.05014.

S. Tong, E. Brown, P. Wu, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal LLMs. NeurIPS, 2024. URL https://arxiv.org/abs/2406.16860.

S. Xu, Y. Wang, Z. Wang, et al. StellaVLA: In-Context Structured Demonstration for Generalizable Vision-Language-Action Models. arXiv:2608.11671, 2026. URL https://arxiv.org/abs/2608.11671.

H. Xue, Y. Ge, Y. Zeng, et al. Point-It-Out: Benchmarking embodied reasoning for vision language models in multi-stage visual grounding. arXiv:2509.25794, 2025. URL https://arxiv.org/abs/2509.25794.

J. Yang, S. Yang, A. W. Gupta, R. Han, L. Fei-Fei, and S. Xie. Thinking in space: How multimodal large language models see, remember, and recall spaces. CVPR, 2025. URL https://arxiv.org/abs/2412.14171.

S. Yang, C. Wang, Y. Chen, et al. Beyond Data Scaling: Representation-Centric Continued Pre-training for Vision-Language-Action Models. arXiv:2608.27550, 2026a. URL https://arxiv.org/abs/2608.27550.

Y. Yang, S. Zeng, T. Lin, et al. ABot-M0: VLA foundation model for robotic manipulation with action manifold learning. arXiv:2602.11236, 2026b. URL https://arxiv.org/abs/2602.11236.

H. Yuan, Z. Liang, A. Chen, et al. Qwen-RobotManip technical report: Alignment unlocks scale for robotic manipulation foundation models. arXiv:2606.17846, 2026a. URL https://arxiv.org/abs/2606.17846. Table 4.

W. Yuan et al. RoboPoint: A vision-language model for spatial affordance prediction for robotics. CoRL, 2024. URL https://arxiv.org/abs/2406.10721.

Y. Yuan, H. Cui, Y. Chen, et al. From seeing to doing: Bridging reasoning and decision for robotic manipulation. arXiv:2505.08548, 2025a. URL https://arxiv.org/abs/2505.08548.

Y. Yuan, H. Cui, Y. Huang, et al. Embodied-R1: Reinforced embodied reasoning for general robotic manipulation. arXiv:2508.13998, 2025b. URL https://arxiv.org/abs/2508.13998.

Y. Yuan, Y. Huang, X. Yao, et al. Embodied-R1.5: Evolving physical intelligence via embodied foundation models. arXiv:2606.11324, 2026b. URL https://arxiv.org/abs/2606.11324.

M. Zawalski, W. Chen, K. Pertsch, O. Mees, C. Finn, and S. Levine. Robotic control via embodied chain-of-thought reasoning. arXiv:2407.08693, 2024. URL https://arxiv.org/abs/2407.08693.

J. Zhang et al. VLM4VLA: Revisiting vision-language-models in vision-language-action models. arXiv:2601.03309, 2026a. URL https://arxiv.org/abs/2601.03309.

Y. Zhang, J. Zhao, C. Fan, et al. PRTS: A Primitive Reasoning and Tasking System via Contrastive Representations. arXiv:2604.27472, 2026b. URL https://arxiv.org/abs/2604.27472.

E. Zhou, J. An, C. Chi, et al. RoboRefer: Towards spatial referring with reasoning in vision-language models for robotics. NeurIPS, 2025a. URL https://arxiv.org/abs/2506.04308.

Z. Zhou, Y. Zhu, M. Zhu, et al. ChatVLA: Unified multimodal understanding and robot control with vision-language-action model. arXiv:2502.14420, 2025b. URL https://arxiv.org/abs/2502.14420.

ETHICS STATEMENT

We acknowledge the ICLR Code of Ethics. This study uses existing robot demonstrations and simulated manipulation benchmarks. DROID videos are real-world recordings, so reuse or release of frames and derived annotations should respect the source dataset's terms and receive appropriate privacy review. Perception-derived object and gripper boxes, phase boundaries, and contact proxies are labeled as estimates rather than simulator-provided evidence.

REPRODUCIBILITY STATEMENT

Sections 3 and 4, including the evaluation protocol in Sec. 4.4, specify the VQA construction, supervision recipes, and policy adaptation. Appendices A, B, C, D, and E document episode-to-IR extraction, VQA construction and verification, data counts, model and training settings, and evaluation settings. Code and data will be released at `https://anonymous.4open.science/r/vqa4vla-anonymous`.

A FROM ROBOT EPISODES TO EMBODIED VQA

Our construction scheme separates reusable episode evidence from supervision choices. Simulator-backed extraction processes RoboCasa annotations; perception-backed extraction processes DROID recordings. The original main-study recipes use RoboCasa-derived records and separately converted supplements (Appendix C). Two further policies use DROID-derived VQA, either alone for Cross-view or together with RoboCasa questions for Generalist (Sec. 5.5).

A.1 SOURCE-SPECIFIC EXTRACTION INTO REUSABLE REPRESENTATIONS

Section 3 defines the episode e_s and the construction in Eq. 1; this appendix details each stage. The evidence graph $\mathcal{G}_e$ retains entity identity, time, coordinate frame, and evidence status: object centers, mask-interior points, gripper centers, and demonstrated interaction locations remain distinct.

Simulator-backed extraction. The RoboCasa (Nasiriany et al., 2026) adapter reads episode annotations, synchronized views, instructions, and available object, gripper, and task-state information. Scripted transforms compute image-plane targets and relations, retaining whether labels are simulator-recorded or rule-derived. This route supplies the episode-derived component of the empirical pools without per-question manual labeling.

Perception-backed extraction. For DROID (Khazatsky et al., 2024), we use Qwen3.8-27B (Qwen Team, 2026b) to identify the task object and select object/gripper instances, with SAM 3 (Carion et al., 2025) propagating their image-plane tracks. Gripper signals supply coarse sampling anchors and separate refined phase estimates. Synchronized camera streams support a distinct cross-view annotation branch. All perception labels keep their estimated status, and Appendix B details the full data flow.

A.2 TASK-CONDITIONED VQA GENERATION

A task specifies available views and frames, the target entity or event, its time and coordinate frame, and valid answer cardinality and support. It is emitted only when the IR provides the required evidence and the target is inferable from the input or explicitly posed as a prediction. Object-location, interaction-location, and destination questions share a point schema but teach distinct targets. Recorded RoboCasa questions illustrate why the target matters. "Point to the visible pear" asks for a normalized object-location point computed from a box. "Open the dishwasher door" asks for a demonstrated interaction location. "Place the cheese in the stand mixer bowl" asks for a destination. Their stored answers use the same point schema, but the supervised information differs. A single-view question about the pan's position relative to the gripper uses spatial relations. A cross-view question about a microwave task instead presents synchronized views and asks for the active subgoal. Complete prompts appear in Appendix C.4.

The question–answer interface determines how a target is presented. JSON and plain-text coordinates can encode the same point; replacing it with a box changes spatial information. Adding a

no-object answer expands valid support, while removing candidate options changes the question, answer, and verifier. These interface changes therefore also change what is supervised. Generalist VQA mixes target classes, capability-specific recipes pool related questions, and task splits select narrower supervision.

A.3 Verification, Reuse, and Dataset Instantiation

Verification separates syntax, bounds, cardinality, and agreement with the supported target. Simulator records, rule-derived labels, robot measurements, and perception estimates retain distinct evidence status; Questions use only evidence visible in their inputs, so an occluded object recorded by the simulator does not yield a visible-object grounding question. Missing observations are not negative labels. Future frames may define prediction targets but are excluded from inputs, and derived variants remain with their source episode when partitioned. Cached IRs support multiple targets and interfaces without repeated extraction, and episodes can be processed in batches. Recipe pools can share examples, and Appendices C and B give the full dataset accounting.

B DROID Episode-to-VQA Construction

This appendix details the perception-backed construction for DROID (Khazatsky et al., 2024) real-robot recordings. Because DROID provides no simulator object states, object boxes and temporal phases are estimated by perception models and signal rules, and each label keeps this estimated status. The DROID-derived questions supply the two additional policy comparisons in Sec. 5.5. They are separate from the RoboCasa-derived rows in Table 2.

B.1 Source Records and Data Flow

The source unit is a recording with an instruction, frame-indexed robot-state records in Parquet, and time-aligned RGB MP4 streams from three cameras. The Parquet records include `episode_index`, `frame_index`, and `observation.state.gripper_position`; the language instruction is read from the episode metadata, with an explicit fallback among available instruction fields. Camera videos are identified by their source filenames. The sampler writes one manifest per episode containing source-video paths, sampled-video paths, and the original frame indices retained in each view. The manifest, not the position of a frame in the shortened MP4, defines its location in the source recording. Parquet frame indices and video timestamps are aligned before a robot-state event is attached to an RGB observation.

The route consists of four branches that meet at VQA compilation:

1. A coarse gripper-signal pass supplies frame-sampling anchors. It reads the Parquet signal, normalizes and smooths it, and marks the signal minimum as a contact anchor and the final frame as release/success anchors. These anchors are used only to select frames.
2. The sampler retains the first and last frames and up to four evenly spaced indices within each coarse interval, removes repeated indices, and writes shortened videos for all three camera streams. A separate view-mapping step chooses one left-camera stream for the large-scale object and gripper trackers. It compares a reference image with a candidate video frame at reduced resolution and records the selected video. The other two views remain available for independently constructed cross-view questions.
3. A refined gripper-signal pass provides a distinct set of temporal boundaries for a 3,000-episode subset. Its configuration uses the 60th percentile, a one-frame state run, and five-frame smoothing. Its output records pre-contact, predicted-contact, hold-end, and release-end frame indices.
4. Object/gripper IR, refined temporal IR, and an existing synchronized cross-view candidate pool feed task-specific question builders. The final builder selects family quotas, renders answer interfaces, and runs structural checks before writing training JSON.

In total, the run processes 11,726 candidate recordings, of which 9,996 yield a three-view sampled-video manifest. The coarse phase pass covers 95,632 recordings from the broader DROID collection.

Each sampled video keeps four to eight frames per view, usually four or five, since coinciding phase boundaries are merged.

B.2 Extracting the DROID Intermediate Representations

Target-object identity and image-plane track. On the selected left-view sampled video, Qwen3.8-27B (Qwen Team, 2026b) reads the instruction and first RGB frame to propose a task-relevant object name, a referring expression, and a 2D point. This is the annotation model, distinct from the Qwen3.5-4B backbone used in the VLM and policy experiments. SAM 3 (Carion et al., 2025) produces text-conditioned mask and box candidates. The selector first takes the three candidates nearest the VLM point and then chooses the highest-scoring SAM candidate among them. The chosen instance initializes SAM3 video propagation. The per-episode detection record stores the instruction field, VLM response, candidates, selected instance, image size, model/configuration metadata, and status. A separate tracking record stores sampled-frame IDs and propagated `bbox_xyxy` values. A first-frame screen keeps only points supported by a visible mask.

Gripper image-plane track. The gripper mapper runs on that same selected video but has a separate identity and SAM3 track. SAM3 first proposes regions for the robot gripper; a Qwen3.8-27B candidate selector uses an anchor frame (the recorded run uses the sampled video's last-frame anchor) to select the gripper candidate. SAM3 propagates it through the shortened clip. The resulting JSON keeps the anchor frame, selected box, candidate rankings, per-frame boxes, model settings, and status. Its box covers the visible gripper assembly. Missing or invalid tracks are never converted into negative object-presence labels. The run produces 9,996 gripper tracks and 9,918 object tracks.

Derived 2D geometric IR. For a box $B = (x_1, y_1, x_2, y_2)$ in an image of width W and height H, the builder represents its center as $(\mathrm{round}(500(x_1 + x_2)/W), \mathrm{round}(500(y_1 + y_2)/H))$ on the $[0, 1000]^2$ grid. For paired object and gripper boxes in the same sampled frame, it defines $r = |B_o \cap B_g| / \max(1, \min(|B_o|, |B_g|))$. Contact-decision candidates use $r \geq 0.05$ for positive and $r \leq 0.01$ for negative examples; intermediate cases are not forced into this binary set. Interaction-point families select a high-overlap frame and use the center of $B_o \cap B_g$. These deterministic transforms are fully reproducible from the saved boxes.

Temporal phases and outcome labels. The coarse phase JSON has labels *approach*, *contact*, *transfer*, *release*, and *success*. Its contact anchor is the global minimum of the normalized, smoothed gripper signal; release and success are both set to the last frame. The resulting contact, release, and success intervals can be single-frame intervals. The separately refined P60/one-frame output supplies the boundary indices used by several final question builders. Each label records the signal rule that produced it. In the refined run, pre-contact is the nearest local minimum of the smoothed normalized signal before predicted contact; hold-end is the last contiguous frame above the percentile cutoff starting at contact; release-end is the first local minimum after hold-end. All boundaries are computed automatically from the gripper signal, without manual annotation.

Cross-view branch. The three synchronized RGB streams support cross-view questions. The object and gripper tracks above are computed on the selected left stream, and the 2,000 cross-view questions come from a separate annotation branch over synchronized camera pairs.

The reusable IR for an episode therefore carries its instruction and source recording, original and sampled frame IDs, view identity, available object and gripper masks/boxes, temporal boundary estimates, and the source/model version of each annotation. Estimated-evidence status is preserved when an answer is compiled from it.

B.3 Compiling IR into Question–Answer Items

A task contract specifies available frames and views, target entity or event, target time, coordinate frame, answer cardinality, and answer support. The builder computes a target from the corresponding IR, instantiates the family's question template, and renders the supported choice, normalized point, point-or-no-object, or trajectory interface. Object centers, demonstrated interaction locations, and destinations share the point syntax but remain distinct targets. All normalized 2D points in this route use the declared $[0, 1000]$ image coordinate system. When a future frame determines a prediction label, it is excluded from the question's input images. Table 4 summarizes the IR and answer construction for each question target.

Table 4: **DROID IR-to-VQA mapping.** The IR used by each question target and how its answer is constructed.

Question target	Input IR and answer construction
Visible object pointing/reference	Screened first-frame object mask and point, with the object phrase from the instruction-guided VLM
Contact decision and interaction point	Same-frame object and gripper tracks, with an overlap threshold for yes/no or the intersection-box center for a point
Destination point	Target-object box center in the final sampled frame
Single-view configuration	Object and gripper box centers with their relative image direction or overlap, as a choice or a point
Secure grasp, completion, planning	Refined phase boundaries, selected input frames, and the instruction, as a choice
Action consequence	First and last full-video frames and the episode instruction, with the instruction-level change as a choice
Cross-view consistency	Synchronized left/right images and the separate cross-view annotation
Gripper trajectory	Screened gripper-box centers at ordered frames

The hybrid builder uses the original 18,023-example RoboCasa Generalist family histogram as quotas. Twenty-nine families are filled from DROID IR and candidate pools (13,940 questions); ten families without supported DROID evidence remain RoboCasa records (4,083 questions). These include multi-object pointing, several reference-object spatial queries, balanced present/absent localization, and long-horizon planning, which need evidence beyond a single target track. The DROID cross-view branch contributes 2,000 questions across phase consistency (673), multi-view reasoning (667), and task consistency (660). It covers 842 recordings.

The emitted record has top-level `id`, `images`, `messages`, `answer`, `reward_spec`, `reward_model`, and `metadata`. The user message includes one `<image>` marker per image and requests the final value in `<answer></answer>`; the reward specification encodes the accepted choice label, point, no-object response, or ordered trajectory. Metadata retains family, episode, source, build version, annotation provenance, and quality tier. The builder checks family quotas, image existence and marker counts, reward-spec/ground-truth agreement, unique IDs, and exact image–question–answer duplicates. The output contains 18,023 records from 5,121 unique episodes across both sources: 13,940 DROID records from 2,959 episodes and 4,083 RoboCasa records from 2,162 episodes. Reward interfaces are 8,800 choice, 7,200 point, 2,000 point-or-no-object, and 23 trajectory. All 21,462 unique images pass the automated readability check, and the final dataset contains no exact image–question–answer duplicates.

C Training Dataset Inventory, Construction, and Templates

C.1 Data Provenance and Recipe Inventory

Table 5 summarizes the training pools: the 18,023-example RoboCasa core, the 5,112-example supplement, their 23,135-example union, the capability-specific and task-split pools, and the two DROID-derived recipes. Because the union and its subsets share examples, the row counts do not sum to a unique dataset size. Reasoning-enabled and reasoning-disabled adaptations use the same example pools with different prompting and training settings.

Sources. All task families listed below are constructed from RoboCasa episodes unless a supplement dataset is named. The supplement draws on three public datasets: Spatial-SSRL-81k (Liu et al., 2025), WanderDream (Liu et al., 2026), and the MolmoAct Midtraining Mixture (Lee et al., 2025). Appendix A describes the episode extraction, and the source-specific conversions and screening are documented below.

Additional evaluated recipes. The depth increment (2,000 examples; 20,023 after merging) is documented in Appendix K. The direct-generation Cross-view variant (Sec. 5.6) converts the same 2,000 examples to answer phrases after removing the candidate options, with a `text_phrase` verifier; it does not add 2,000 new visual examples. Table 6 shows how the supplement changes the target-class balance.

Table 5: **Training pools.** Composition refers to target classes or task families. Parent pools and subsets overlap.

Recipe / pool	Examples	Composition
Generalist VQA	18,023	Nine target classes with 2,000 each; 23 trajectories
Generalist VQA (DROID + RoboCasa)	18,023	DROID 13,940; RoboCasa 4,083; original family quotas
DROID Cross-view VQA	2,000	Phase consistency 673; multi-view reasoning 667; task consistency 660
Spatial/Temporal increment	5,112	Spatial-SSRL 1,000; WanderDream 2,112; MolmoAct 2,000
Generalist + Spatial/Temporal	23,135	Core plus the complete increment
Spatial VQA	4,600	Cross-view consistency 2,000; spatial geometry 2,600
Pointing VQA	5,400	Object grounding 2,200; counting/presence 2,000; spatial geometry/relation 1,200
Affordance & Placement VQA	4,200	Placement 2,000; part/function affordance 2,200
Cross-view consistency VQA	2,000	Multi-view reasoning 667; phase and task consistency 1,333
Single-view spatial VQA	2,000	Single-view spatial relation 615; Spatial-SSRL 770; WanderDream 615
Visible object pointing VQA	1,600	800 shared present-object records plus 800 positive point records
Visible + absent object pointing VQA	1,600	800 present-object records and 800 explicit no-object answers
Interaction-point grounding VQA	1,620	Rule-based contact 569; object affordance 401; object-part pointing 348; part affordance 302
Destination-point grounding VQA	1,620	Spatial-referring placement 947; action-goal point 673

Table 6: **Target-class balance before and after supplementation.** Names denote selection categories, which can contain several answer types.

Target class	Core	Added	Merged
Object grounding	2,000	200	2,200
Counting, multi-object, and negative	2,000	0	2,000
Spatial geometry	2,000	1,800	3,800
Cross-view perspective	2,000	0	2,000
Part/function affordance	2,000	200	2,200
Placement destination	2,000	0	2,000
Planning, future, and causal questions	2,000	572	2,572
State and success	2,000	0	2,000
Temporal action dynamics	2,000	340	2,340
Trajectory	23	2,000	2,023
Total	18,023	5,112	23,135

C.2 Construction and Selection of the Empirical Pools

Episode-derived questions. RoboCasa annotation manifests provide synchronized camera observations, object/gripper geometry, episode instructions, and task or phase labels. The recorded family selects the relevant question and evidence: a visible object location, a demonstrated interaction target, an observed destination, a synchronized cross-view relation, or a temporal outcome. Some records convert a bounding box into its center point; the original box supervision and conversion remain in metadata. Box-center targets stay distinct from mask-interior points and contact locations. Text-to-choice records retain the original answer, candidate option texts, correct label, and conversion version. Distractors are semantically similar objects or tasks.

Supplement conversion. Spatial-SSRL records supply multiple-choice spatial tasks, including patch rearrangement questions. WanderDream records retain the source situation, question, answer, temporal phase, and view information; free-form answers are converted to four-option questions. Their metadata records the conversion model and transform. A single image can be a composite of route views, so one image file need not mean one time step. MolmoAct supplies 1,000 Home and 1,000 Tabletop robot-arm trajectories. The adapter converts source coordinates from a 255-scale representation to $[0, 1000]$, samples the requested waypoint count, and serializes the trajectory as ordered point objects. Its screening metadata specifies interpolated trajectory RMSE with threshold 200 in normalized coordinates. This aligns the answer interface with ShareRobot while retaining the MolmoAct image and task domain.

Difficulty filtering and sampling. Difficulty screening uses Qwen3.5-4B with thinking disabled and four responses per candidate. Let c_i count responses accepted by the record's verifier. Selection retains $1 \leq c_i \leq 3$, removing both zero-success and four-success candidates. Target-class and family quotas then determine the selected mixture. Supplement metadata records family/answer-label balancing; the MolmoAct policy also balances scene and waypoint-count strata. Three pairs additionally match the complete distribution of c_i (Table 7).

Table 7: **Difficulty matching within task pairs.** Each arm of a pair has the listed counts.

Pair	$c_i = 1$	$c_i = 2$	$c_i = 3$	Total per arm
Cross-view consistency VQA / Single-view spatial VQA	670	666	664	2,000
Visible object pointing VQA / Visible + absent object pointing VQA	651	569	380	1,600
Interaction-point grounding VQA / Destination-point grounding VQA	876	571	173	1,620

Serialization and reward specification. Each JSON record contains `id`, `images`, `messages`, `answer`, `reward_model`, `reward_spec`, and `metadata`. The prompt is stored as a user message with image placeholders. The reference answer contains the final value; the training prompt requests its serialization inside an `<answer>` block. Choice records retain the allowed labels and correct label. Point records retain the reference coordinate, coordinate scale, schema, and tolerance (40 for points). Presence records add an expected point/no-object kind. Trajectory records retain ordered coordinates, waypoint count, and tolerance. These fields are the verifier inputs, and the reward rules are given in Appendix D.1.2. The merged pool has 11,912 choice, 7,200 point, 2,000 point-or-no-object, and 2,023 trajectory records. The two Pointing splits share a difficulty histogram and differ in answer support.

Packaging and traceability. Portable records rewrite only image paths relative to the package root; original annotation paths remain as provenance. The package manifest records input and portable JSON SHA-256 values. Its image mapping contains 31,144 unique original image paths, 35,708 references in the merged pool, and 5,974,132,018 image bytes, with no missing images.

C.3 Question and Answer Templates

Table 8 gives parameterized forms of the training prompts. Braced variables denote source-dependent substitutions; option texts and numeric answers come from the associated annotation. One representative template is shown per main family. The extraction prompts used by the anno-

tation models are described in Appendix B; the following prompts are the inputs to the adapted VLM.

Table 8: **Representative training question templates.** Every point or trajectory answer uses normalized image coordinates.

Task	Question template	Answer value
Object grounding	Point to the visible {object}.	One point
Visible + absent object pointing VQA	Point to every visible {object}, exactly one point per instance. If none are visible, output "No object".	Point list or text
Spatial geometry	Which target/gripper configuration is shown? {relation options}	Option label
Cross-view consistency VQA	These are synchronized left/right views. Given the composite instruction "{instruction}", which task-relevant subgoal is active?	Option label
Interaction-point grounding VQA	Given the task "{instruction}", point to the demonstrated interaction target.	One point
Destination-point grounding VQA	{instruction}. Identify the final destination region where the object should be placed. Do not mark the source object.	One point
Planning/future	What is expected to happen next to complete "{instruction}"?	Option label
State/success	Based on these frames of one robot execution, was the task "{instruction}" completed successfully?	Yes/no option
Temporal dynamics	What changed from the first image to the second?	Option label
Trajectory	Use 2D points to trace the movement trajectory of the robotic arm (not the manipulated object) as it moves to complete the task.	Ordered points

Shared thinking-mode suffix (verbatim). The core prompts and the MolmoAct supplement use the following suffix. Other supplemental choices use the shorter suffix below.

```
Use the model's thinking mode to reason from the visual evidence. Then put exactly the
     required final value inside <answer></answer>. Put no explanation or extra fields inside
      the answer tags.
```

```
First think briefly, then give the final answer inside <answer></answer> tags. Put only the
     answer itself inside the tags, with no extra words.
```

Output templates. Here the symbols are placeholders, not literal model outputs. A choice uses the labels actually provided in the question; success questions may have two options. The point-list length is determined by the task.

```
Choice:      <answer>A</answer>
Point:       <answer>[{"point_2d":[x,y]}]</answer>
Absent:      <answer>No object</answer>
Trajectory:  <answer>[{"point_2d":[x1,y1]}, ...,
                     {"point_2d":[xN,yN]}]</answer>
```

Example of an actual point record. The following values are taken from a Generalist VQA object-pointing record built from a RoboCasa replay. The stored answer is a box-center target.

```
Question: Point to the visible pear.
Stored answer: [{"point_2d":[653,217]}]
Reward fields:
  type: point
  point: [653.0,217.0]
  tol: 40.0
  strict_schema:
    kind: point_object_array
    point_key: point_2d
    length: 1
    coordinate_dim: 2
    coordinate_min: 0
    coordinate_max: 1000
    allow_extra_keys: false
```

Trajectory horizons. Core trajectory questions ask for eight evenly sampled waypoints over the next ten simulator frames at 20 Hz, excluding the current point. MolmoAct questions instead ask for

four waypoints covering task completion. Both serialize ordered 2D points, and each record keeps its temporal horizon in the metadata.

C.4 Complete Input Prompts and Reference Output Responses

The following fourteen examples instantiate the templates using actual training records. Each *Input prompt* reproduces the complete user message, including image placeholders, options, coordinate conventions, and the reasoning/output instruction. The associated image files are supplied separately in the record's ordered `images` field; `<image>` denotes their insertion into the model input. Line wrapping is typographical. The number of image files is stated for each example; a composite image may contain several temporal observations.

Each *Output response* is the stored reference answer wrapped in `<answer>` tags as requested by the prompt. These outputs are the expected final responses. The dataset stores final answers and verifiers, so GRPO leaves the reasoning free. The prompts include both the task-specific output instruction and the final answer-wrapper instruction, and illustrate the reasoning-enabled format.

Example 1: Object grounding. Pool: Generalist VQA. Image files: 1.
Input prompt (user message):

```
<image>
Point to the visible pear.
Return only normalized 2D point coordinates in the range [0, 1000]. Use
    exactly this JSON format: [{"point_2d":[x,y]}]. Return exactly one
    point. Do not output <object>, <area>, <affordance>, <grasp pose>, a
    bounding box, labels, or explanations.
Use the model's thinking mode to reason from the visual evidence. Then
    put exactly the required final value inside <answer></answer>. Put no
     explanation or extra fields inside the answer tags.
```

Output response (reference final answer):

```
<answer>[{"point_2d":[653,217]}]</answer>
```

Example 2: Present-object localization. Pool: Generalist VQA. Image files: 1.
Input prompt (user message):

```
<image>
Point to every visible cheese, exactly one point per instance. If none
    are visible, output "No object". Otherwise return only JSON in the
    format [{"point_2d":[x,y]}], with integer coordinates normalized to
    [0,1000].
Use the model's thinking mode to reason from the visual evidence. Then
    put exactly the required final value inside <answer></answer>. Put no
     explanation or extra fields inside the answer tags.
```

Output response (reference final answer):

```
<answer>[{"point_2d":[549,705]}]</answer>
```

Example 3: Absent-object response. Pool: Visible + absent object pointing VQA. Image files: 1.
Input prompt (user message):

```
<image>
Point to every visible pineapple, exactly one point per instance. If
    none are visible, output "No object". Otherwise return only JSON in
    the format [{"point_2d":[x,y]}], with integer coordinates normalized
    to [0,1000].
Use the model's thinking mode to reason from the visual evidence. Then
    put exactly the required final value inside <answer></answer>. Put no
     explanation or extra fields inside the answer tags.
```

Output response (reference final answer):

```
<answer>No object</answer>
```

Example 4: Single-view spatial geometry. Pool: Generalist VQA. Image files: 1.
Input prompt (user message):

```
<image>
Which target/gripper configuration is shown?
(A) the pan is to the right of the robot gripper
(B) the pan is horizontally aligned with the robot gripper
(C) the relation cannot be determined
(D) the pan is to the left of the robot gripper
Answer with A, B, C, or D.
Use the model's thinking mode to reason from the visual evidence. Then
    put exactly the required final value inside <answer></answer>. Put no
     explanation or extra fields inside the answer tags.
```

Output response (reference final answer):

```
<answer>D</answer>
```

Example 5: Cross-view task reasoning. Pool: Generalist VQA. Image files: 2.
Input prompt (user message):

```
<image> <image>
These are synchronized left/right views. Given the composite instruction
     "Place the microwavable item in the microwave. Then, close the
    microwave door and press the start button.", which task-relevant
    subgoal is active?
Which option best answers the question?
(A) interact with the glass cup as part of the composite task
(B) interact with the dishwasher as part of the composite task
(C) interact with the rolling pin as part of the composite task
(D) interact with the tupperware as part of the composite task
Answer with A, B, C, or D only.
Use the model's thinking mode to reason from the visual evidence. Then
    put exactly the required final value inside <answer></answer>. Put no
     explanation or extra fields inside the answer tags.
```

Output response (reference final answer):

```
<answer>A</answer>
```

Example 6: Interaction-point grounding VQA. Pool: Generalist VQA. Image files: 1.
Input prompt (user message):

```
<image>
Given the task "Open the dishwasher door.", point to the demonstrated
    interaction target.
Return only normalized 2D point coordinates in the range [0, 1000]. Use
    exactly this JSON format: [{"point_2d":[x,y]}]. Return exactly one
    point. Do not output <object>, <area>, <affordance>, <grasp pose>, a
    bounding box, labels, or explanations.
Use the model's thinking mode to reason from the visual evidence. Then
    put exactly the required final value inside <answer></answer>. Put no
     explanation or extra fields inside the answer tags.
```

Output response (reference final answer):

```
<answer>[{"point_2d":[714,766]}]</answer>
```

Example 7: Destination-point grounding VQA. Pool: Generalist VQA. Image files: 1.
Input prompt (user message):

```
<image>
Place the cheese in the stand mixer bowl
Identify the final destination region where the object should be placed.
     Do not mark the source object.
Return only normalized 2D point coordinates in the range [0, 1000]. Use
    exactly this JSON format: [{"point_2d":[x,y]}]. Return exactly one
    point. Do not output <object>, <area>, <affordance>, <grasp pose>, a
    bounding box, labels, or explanations.
Use the model's thinking mode to reason from the visual evidence. Then
    put exactly the required final value inside <answer></answer>. Put no
     explanation or extra fields inside the answer tags.
```

Output response (reference final answer):

```
<answer>[{"point_2d":[332,568]}]</answer>
```

Example 8: Planning and future prediction. Pool: Generalist VQA. Image files: 3.
Input prompt (user message):

```
<image>
<image>
<image>
What is expected to happen next to complete "Preheat the oven by turning
     the temperature knob."?
Which option best answers the question?
(A) the robot will manipulate the pot as instructed
(B) the robot will manipulate the oven as instructed
(C) the robot will manipulate the tongs as instructed
(D) the robot will manipulate the kiwi as instructed
Answer with A, B, C, or D only.
Use the model's thinking mode to reason from the visual evidence. Then
    put exactly the required final value inside <answer></answer>. Put no
     explanation or extra fields inside the answer tags.
```

Output response (reference final answer):

```
<answer>B</answer>
```

Example 9: Execution success. Pool: Generalist VQA. Image files: 8.
Input prompt (user message):

```
<image>
<image>
<image>
<image>
<image>
<image>
<image>
<image>
Based on these frames of one robot execution, was the task "transfer the
     potato from the counter to the microwave" completed successfully?
A. Yes
B. No
Answer with the option letter only.
Use the model's thinking mode to reason from the visual evidence. Then
    put exactly the required final value inside <answer></answer>. Put no
     explanation or extra fields inside the answer tags.
```

Output response (reference final answer):

```
<answer>B</answer>
```

Example 10: Temporal action dynamics. Pool: Generalist VQA. Image files: 2.
Input prompt (user message):

```
<image>
<image>
What changed from the first image to the second?
Which option best answers the question?
(A) the mango was moved from the cabinet into the counter
(B) the robot completed "Take the cheese, steak, and broccoli from the
    fridge and place them on the plate on the dining counter to create an
     appetizer plate."
(C) the mango was moved from the pan into the plate
(D) the robot completed "Place the fish on the plate and place the
    broccoli on the same plate."
Answer with A, B, C, or D only.
Use the model's thinking mode to reason from the visual evidence. Then
    put exactly the required final value inside <answer></answer>. Put no
     explanation or extra fields inside the answer tags.
```

Output response (reference final answer):

```
<answer>C</answer>
```

Example 11: Short-horizon end-effector trajectory. Pool: Generalist VQA. Image files: 1.
Input prompt (user message):

```
<image>
You are currently a robot performing robotic manipulation tasks. The
    task instruction is: Pick the ladle from the counter and place it in
    the drawer.
Use 2D points to trace the immediate short-horizon movement trajectory
    of the robotic arm end-effector (not the manipulated object) over the
     next 10 simulator frames at 20 Hz.
The current gripper position is [368, 675].
Provide the points in trajectory order as 8 evenly sampled waypoints. Do
     not include the current point.
Return only a valid JSON array in exactly this format: [{"point_2d":[x1,
    y1]},...,{"point_2d":[xN,yN]}].
Coordinates are [x,y] values normalized to [0,1000] on both axes. Do not
     add labels, bounding boxes, z coordinates, or extra JSON fields.
Use the model's thinking mode to reason from the visual evidence. Then
    put exactly the required final value inside <answer></answer>. Put no
     explanation or extra fields inside the answer tags.
```

Output response (reference final answer):

```
<answer>[{"point_2d":[368,686]},{"point_2d":[368.5714,700.1429]},{"
    point_2d":[370.5714,714.2857]},{"point_2d":[371.8571,728.4286]},{"
    point_2d":[375.2857,739.2857]},{"point_2d":[377.8571,750.4286]},{"
    point_2d":[380.4286,760]},{"point_2d":[382,766]}]</answer>
```

Example 12: Spatial-SSRL patch rearrangement. Pool: Spatial/Temporal increment. Image files: 1.
Input prompt (user message):

```
<image>
The image has been divided into 4 shuffled patches labeled 0, 1, 2, and
    3. One of the four patches is masked completely by white pixels.
    Based on visual clues such as continuity, alignment, and context,
    answer the correct arrangement of the patches to restore the original
     image, where the format is 'TopLeft-TopRight-BottomLeft-BottomRight
    '.
(A) 2-3-0-1
(B) 2-1-0-3
(C) 0-1-2-3
(D) 2-0-1-3
Answer with A, B, C, or D.
First think briefly, then give the final answer inside <answer></answer>
     tags. Put only the answer itself inside the tags, with no extra
    words.
```

Output response (reference final answer):

```
<answer>B</answer>
```

Example 13: WanderDream route reasoning. Pool: Spatial/Temporal increment. Image files: 1.
Input prompt (user message):

```
<image>
Target situation: If I want to stand with a metal sink one meter in
    front and a black monitor with blue glow one meter to the right.
Based on the visual evidence along the route shown from left to right,
    How will my distance to the small vase change while moving?
(A) It increases quickly as I leave the near table.
(B) It decreases steadily as I approach the near table.
(C) It increases slowly as I move toward the black monitor.
(D) It remains constant while I adjust my position relative to the sink.
Answer with A, B, C, or D.
First think briefly, then give the final answer inside <answer></answer>
     tags. Put only the answer itself inside the tags, with no extra
    words.
```

Output response (reference final answer):

```
<answer>A</answer>
```

Example 14: MolmoAct task-completion trajectory. Pool: Spatial/Temporal increment. Image files: 1.
Input prompt (user message):

```
<image>
You are currently a robot performing robotic manipulation tasks. The
    task instruction is: flip the mug upright.
Use 2D points to trace the movement trajectory of the robotic arm (not
    the manipulated object) as it moves to complete the task. Provide the
     points in trajectory order, and the number of points must be exactly
     4.
The final value must be a JSON array in exactly this format: [{"point_2d
    ":[x,y]}]. Coordinates are [x,y] values normalized to integers or
    decimals from 0 to 1000. Do not add labels, bounding boxes, z
    coordinates, Markdown, or extra JSON fields.
Use the model's thinking mode to reason from the visual evidence. Then
    put exactly the required final value inside <answer></answer>. Put no
     explanation or extra fields inside the answer tags.
```

Output response (reference final answer):

```
<answer>[{"point_2d":[388.2353,392.1569]},{"point_2d
    ":[441.83,546.4053]},{"point_2d":[517.647,473.2026]},{"point_2d
    ":[600.0,392.1569]}]</answer>
```

D Training Data and Configuration Coverage

D.1 VLM Adaptation with GRPO

D.1.1 Optimization and Sampling

All VQA-adapted models are initialized from the same pretrained Qwen3.5-4B checkpoint and updated with full-parameter GRPO. In the reference 4B configuration, each training update contains 16 prompts and samples eight responses per prompt, yielding 128 responses. Rewards are normalized within the eight responses associated with each prompt to obtain response-level advantages; no separate value or critic model is trained.

We use a token-mean PPO objective with dual clipping. The standard clipping threshold is 0.2 and the dual-clip coefficient is 3.0. A fixed K2 KL loss with coefficient 0.001 regularizes the current policy against the frozen reference policy. This term is applied in the optimization loss rather than added to the task reward. We use AdamW with learning rate 5×10^{-7}, 3% warmup followed by

a constant schedule, weight decay 0.01, and gradient clipping at 1.0. Responses are sampled with temperature 1.0 and top-p 1.0.

Table 9 gives the reference configuration for the Qwen3.5-4B GRPO runs. Most reported adaptations enable reasoning during training, while the reasoning-disabled ablation changes the prompting and decoding mode. All VLM evaluations request direct answers with thinking disabled. Run-specific settings are given with the corresponding experiments.

Table 9: **Reference configuration for Qwen3.5-4B GRPO adaptation.** Run-specific settings are given with each experiment.

Setting	Value
Initialization	Qwen3.5-4B pretrained checkpoint
Parameter update	Full-parameter fine-tuning
Prompts per update	16
Responses per prompt / update	8 / 128
Advantage estimation	Reward normalization within each prompt group
Policy objective	Token-mean PPO with dual clipping
PPO clipping threshold / dual-clip coefficient	0.2 / 3.0
PPO epochs per batch	1
Reference regularization	Fixed K2 KL loss, coefficient 0.001
Reward-side KL penalty	Disabled
Optimizer	AdamW, $\beta_1 = 0.9$, $\beta_2 = 0.999$
Learning rate / schedule	5×10^{-7}; 3% warmup, then constant
Weight decay / gradient clipping	0.01 / 1.0
Sampling	Temperature 1.0, top-p 1.0
Maximum prompt / response length	6,144 / 8,192 tokens
Maximum model context	14,336 tokens
Training precision	bfloat16
Trainer / rollout backend	FSDP2 / asynchronous vLLM
Tensor parallelism	1 per rollout engine
Rollout–trainer synchronization	Every 8 trainer updates
Maximum image side before model preprocessing	768 pixels

D.1.2 Verifier-Grounded Reward

Each training example stores a typed reward specification with its ground-truth target, expected output schema, and task-specific tolerance. The trainer scores the final answer with the verifier associated with that output type. Difficulty screening and GRPO training use the same typed verifiers. The reward combines strict task correctness, bounded partial credit for continuous predictions, an invalid-format penalty, and an anti-collapse penalty for trajectories.

Strict correctness. A response receives a correctness reward of 1.0 only if its extracted final answer passes the task-specific verifier and satisfies the required container and schema. Schema checks include the expected JSON keys, array length, coordinate dimensionality and range, and, where applicable, the number of trajectory points. Reasoning outside the final answer container is not scored directly.

Bounded partial credit. A well-formed but incorrect response can receive partial credit only when its output type admits a continuous quality measure. Geometric answers receive at most 0.35 and numeric answers at most 0.15. Multiple-choice, permutation, and text answers receive no partial credit. Strictly correct answers do not additionally receive this partial credit.

For trajectory questions, a prediction covering less than 25% of the ground-truth path length is treated as collapsed. It receives neither a correctness reward nor positive partial credit and is penal-

ized by up to 0.20 according to the motion deficit. The 4B runs use neither a positive format bonus nor an overlength reward penalty.

D.1.3 TYPED VERIFIERS

Table 10 lists the output types needed for the reported experiments. Point and trajectory coordinates use a 0–1000 image coordinate system.

Table 10: **Typed verifiers used by the reported VQA recipes.**

Answer type	Strict correctness criterion	Partial-credit behavior
Multiple choice	Predicted option label exactly matches the target label	None
Short answer text	Normalized exact match or token F1 of at least 0.7; the Cross-view direct-generation verifier also imposes a 24-word limit	None
Point	Two-dimensional point with Euclidean error at most 40	Distance-based quality, capped by the geometric weight
Point or no object	Accepted absence phrase, or one-to-one point matching with maximum error at most 40	Point-set quality for non-exact geometric predictions
2D trajectory	Required point count and schema, RMSE at most 40, and sufficient path motion	Quality decreases with trajectory RMSE and is capped by the geometric weight
Numeric range	Ordered two-endpoint range whose maximum endpoint error is within the stored tolerance; the Depth data use 0.0025	Quality decreases with endpoint error and is capped by the numeric weight

Cross-view direct generation. The multiple-choice condition uses exact option-label verification. The direct-generation condition keeps the same 2,000 samples, paired images, question stems, and order. It removes the candidate options, replaces the option-label target with its answer phrase, and scores the response with the short-text verifier. The comparison therefore evaluates the complete answer interface, covering the presented options, the target, and its verifier.

D.2 DOWNSTREAM TRAINING

All eleven main-study policies and six additional policies use Qwen3.5-4B and the same LIBERO 4-in-1 recipe with 30,000 updates. Table 11 lists this configuration, which every policy shares, including the dataset, image inputs, action horizon, and checkpoint step.

The training mixture contains 454 Object, 428 Goal, 432 Spatial, and 379 LIBERO-10 episodes, totaling 1,693. The corresponding frame counts are 66,984, 52,042, 52,970, and 101,469, totaling 273,465. The LeRobot v2.1 sources use the four `no_noops` subsets, 20 FPS, Franka embodiment, and equal registry mixture weights. A training sample contains two camera images and an instruction. The reference prompt additionally asks the model to identify key objects and locate their boxes in `[x1,y1,x2,y2]` format. All policies share this prompt.

D.3 VLM ADAPTATION DATA AND CHECKPOINTS

All ten main-study GRPO adaptations start from Qwen3.5-4B.

Capability-specific pools. Spatial VQA uses 2,600 spatial-geometry and 2,000 Cross-view consistency VQA examples, all multiple-choice. Pointing VQA uses 2,200 object-grounding, 2,000 counting/multi-object/negative, 1,000 spatial-relation, and 200 spatial-geometry examples. Its reward types are 3,200 point, 2,000 point-or-no-object, and 200 choice. Of these, 1,172 point targets are box centers. Affordance & Placement VQA uses 2,000 placement-goal and 2,200 part/function-affordance examples, with 4,000 point and 200 choice rewards. Point coordinates use a $[0, 1000]$ normalization. All three recipes enable thinking, ask for a final `<answer>` block, and use the typed-verifier reward of Appendix D.1.2 with two training GPUs and six rollout GPUs.

Table 11: **Downstream policy configuration.** All policies share these settings.

Setting	Value
Framework / action head	StarVLA QwenGR00T / DiT-B
Head depth / width	16 layers / 1,024
Policy updates	30,000
Input / output	Two images + instruction / eight 7D actions
Training GPUs / batch	8 / 4 per GPU, accumulation 4
Effective batch	128
Trainable modules	VLM, interface, action model
Learning rates	Action 10^{-4}, VLM 2.5×10^{-5}, interface 10^{-5}
Schedule	Cosine, 1,500 warmup steps, minimum 10^{-6}
Optimizer parameters	$\beta = (0.9, 0.95)$, $\epsilon = 10^{-8}$, weight decay 10^{-8}
Gradient clipping	1.0
Precision / parallelism	BF16, ZeRO-2, FlashAttention 2, gradient checkpointing
Diffusion repetition	8 during training
Evaluation sampler	DDIM, 10 steps, `do_sample=False`
Action ensemble	Enabled, horizon 3
Proprioception at evaluation	Not passed to model
Random seeds	Two per policy, results averaged

Representative point-pool examples ask for the object nearest a marked gripper, a referred ladle, or an explicit no-object response when a pineapple is absent. Affordance/placement examples ask for the counter destination of a honey bottle, a valid location inside a pan, or a demonstrated brush grasp. Each pool thus covers a wide range of objects, scenes, and targets.

Generalist VQA adaptation runs. Generalist VQA uses 18,023 examples: nine classes with 2,000 each and 23 trajectories. It retains examples answered correctly one to three times in four non-thinking base-model rollouts and trains for 1,127 steps, with 16 prompts per step and eight responses per prompt. Generalist VQA + Spatial/Temporal Data adds 1,000 Spatial-SSRL, 2,112 WanderDream, and 2,000 MolmoAct examples. The resulting 23,135-example pool has 11,912 choice, 7,200 point, 2,000 point-or-no-object, and 2,023 trajectory examples. Its learning rate is 5×10^{-7}, with a 6,144-token prompt limit and an 8,192-token response limit. The Generalist reference enables thinking during training, while VLM evaluation disables it. The reasoning-disabled ablation is detailed in Appendix J.

Spatial VQA splits. Cross-view consistency VQA uses 667 multi-view reasoning and 1,333 phase and task consistency examples. Single-view spatial VQA uses 615 single-view spatial relation, 770 Spatial-SSRL, and 615 WanderDream examples. Both have difficulty counts 670/666/664 for one, two, or three correct responses among four base-model samples. Both train for 125 updates with learning rate 5×10^{-7}, 16 prompts per batch, eight rollouts per prompt, and thinking enabled. VLM evaluation disables thinking.

Pointing VQA splits. Both arms have 800 shared present-object examples. Visible object pointing VQA adds 800 further positive point examples. Visible + absent object pointing VQA adds 800 same-image absent-object questions, yielding 800 point and 800 no-object answers. The two arms match in difficulty distribution and GRPO hyperparameters.

Affordance & Placement VQA splits. Interaction-point grounding VQA uses 569 rule-based contact, 401 object affordance, 348 object-part pointing, and 302 part affordance examples. Destination-point grounding VQA uses 947 spatial-referring placement and 673 action-goal point examples.

Each arm has 1,620 single-image, single-point examples with matched difficulty and identical GRPO hyperparameters.

Pool overlap. The three capability-specific pools are disjoint by sample ID and total 14,200 examples. The expanded pool contains the 18,023 Generalist VQA examples and the 5,112 supplemental examples.

E Recipe Composition and Evaluation Protocol

E.1 Recipe Composition and Comparison Controls

Reference and capability-specific recipes. Unadapted Qwen3.5-4B (Qwen Team, 2026a) provides the reference for measuring the benefit of VQA adaptation. *Generalist VQA* uses GRPO on 18,023 examples covering multiple embodied tasks. To examine whether focusing on a capability family yields a more useful initialization, we compare it with *Spatial VQA* (4,600 examples), *Pointing VQA* (5,400), and *Affordance & Placement VQA* (4,200). These recipes emphasize spatial relations, object localization, and interaction or placement targets, respectively. They are selected from a 23,135-example pool that includes Generalist VQA and 5,112 supplemental examples. Thus this comparison evaluates practical choices of supervision content and budget. Every adaptation starts from Qwen3.5-4B.

Task splits within each capability family. A family-level comparison can conceal differences between the tasks it contains. We therefore compare two narrower recipes within each family, matching example counts, reported difficulty distributions, and GRPO hyperparameters within each pair. The pairs address the following choices:

- **Spatial VQA.** *Cross-view consistency VQA* uses paired images from RoboCasa multi-view reasoning and phase and task consistency questions. *Single-view spatial VQA* uses single-image spatial relation questions from RoboCasa, Spatial-SSRL, and WanderDream. Each recipe contains 2,000 multiple-choice examples and uses 125 adaptation updates. This pair compares spatial supervision built around different visual contexts. Cross-view consistency VQA denotes the multiple-choice (MCQ) recipe in the main-study tables and figures.
- **Pointing VQA.** *Visible object pointing VQA* contains 1,600 present-object examples. *Visible + absent object pointing VQA* retains 800 of these examples and replaces the other 800 with absent-object questions on the retained images. This pair examines whether allocating half of a fixed budget to absence questions benefits control.
- **Affordance & Placement VQA.** *Interaction-point grounding VQA* supervises interaction locations, while *Destination-point grounding VQA* supervises destinations. Both contain 1,620 single-image examples with a single-point answer. The common answer interface allows us to compare recipes that predict different manipulation targets.

We additionally examine training reasoning and two extensions of Generalist VQA in Appendix J: *Generalist VQA + Spatial/Temporal Data* adds 5,112 examples, and *Generalist VQA + Depth Geometry* adds 2,000 examples. The depth recipe contains 20,023 examples in total and is a data extension rather than a 2,000-example task split. Together these ablations add three policies to the eleven main-study initializations. A fourth additional policy uses a direct-generation variant of the 2,000-example Cross-view pool: it removes answer options and predicts the correct answer phrase (Sec. 5.6).

Each comparison uses a reference appropriate to its question. The unadapted backbone measures the net benefit of VQA adaptation. The within-family pairs compare equal-sized task recipes. Data extensions are compared with Generalist VQA under the same training reasoning setting. Cross-recipe comparisons then show how transfer profiles differ.

Common policy training. To compare the initializations under a common action-learning setup, we use StarVLA (StarVLA Community, 2026) with the QwenGR00T configuration and a DiT-B action head. The reference configuration has 16 action-head layers and a hidden size of 1,024. The VLM, interface, and action model are trained jointly for 30,000 optimizer steps on LIBERO 4-in-1 (Liu et al., 2023), comprising 1,693 episodes and 273,465 frames across 40 tasks. Each policy receives agent-view and wrist-view images at 224×224 resolution together with the task instruction, and

predicts chunks of eight 7D actions. Evaluation uses no proprioceptive input. All initializations use this recipe. Detailed hyperparameters and configuration coverage are provided in Appendix D.

E.2 Evaluation Coverage and Score Conventions

Policy performance. We evaluate on all 10,030 LIBERO-plus variants. The Goal, Spatial, Object, and LIBERO-10 suites contain 2,591, 2,402, 2,518, and 2,519 variants, respectively. Each suite includes perturbations of camera viewpoint, robot initial state, language, lighting, background, sensor noise, and object layout. Every policy is trained with two random seeds, and all reported success rates are their mean. We report count-weighted success across the full benchmark, then examine perturbation categories and their intersections with task suites. For the paired recipes, we also align individual configurations with the Qwen3.5-4B reference to distinguish new successes from losses of previously successful configurations.

All success-rate differences are computed from counts before rounding and reported as absolute differences in percentage points (%).

VLM performance. We evaluate direct answers with thinking disabled. The main comparison uses the *Embodied VLM Avg.*, the unweighted mean of 19 EmbodiedEvalKit (Yuan et al., 2026b) benchmark scores common to the reported aggregates. It excludes CV-Bench and RoboVQA from the full set of 21 and includes two trajectory scores conditional on valid outputs. The 19 benchmarks cover spatial understanding (BLINK (Fu et al., 2024), SAT (Ray et al., 2025), EmbSpatial-Bench (Du et al., 2024), VSI-Bench (Yang et al., 2025), RoboSpatial-Home (Song et al., 2025)), pointing and affordance (PixMo-Points (Deitke et al., 2024), PointBench (Cheng et al., 2025), RoboRefIt (Lu et al., 2023), RefSpatial-Bench (Zhou et al., 2025a), Where2Place (Yuan et al., 2024), PartAfford (Yuan et al., 2025b), RoboAfford (Hao et al., 2025), PIO (Xue et al., 2025), VABench-Point (Yuan et al., 2025a)), trajectory prediction (VABench-Visual-Trace (Yuan et al., 2025a), ShareRobot-Trajectory (Ji et al., 2025)), and embodied reasoning and planning (ERQA (Gemini Robotics Team et al., 2025), EgoPlan-Bench2 (Qiu et al., 2024), Cosmos (NVIDIA et al., 2025a)). Benchmark sources follow EmbodiedEvalKit (Yuan et al., 2026b). CV-Bench (Tong et al., 2024) and RoboVQA (Sermanet et al., 2024) are reported in the full-21 results (Appendix G). Complete per-benchmark reports are available for Qwen3.5-4B, Generalist VQA, the three capability-specific recipes, and the two Spatial VQA splits. The depth extension adds a complete report under the same convention, and analyses that include it keep the original eleven and seven models for reference. Appendix G also recomputes the full 21-benchmark mean while holding initialization coverage fixed. The separate evaluation batch used for the reasoning and spatial/temporal ablation is reported in Appendix J. Its non-trajectory average is reported separately. Depth metrics and source conventions appear in Appendix K. The cross-view interface comparison adds a full VLM report and a policy result in Appendix L. It is reported separately from the fixed model sets in the transfer figures and correlation checks.

Data provenance. Policy training uses LIBERO 4-in-1 demonstrations, and policy evaluation uses LIBERO-plus variants. VQA training sources and evaluation benchmarks are documented in Appendices D and F.

E.3 Main Table Provenance and Conventions

OpenVLA-OFT and π_0-FAST follow the LIBERO-plus benchmark (Fei et al., 2025). The StarVLA LIBERO-plus results, as well as the π_0, ABot-M0, and Cosmos-Policy rows, are taken from Qwen-RobotManip (Yuan et al., 2026a). Our results are rounded to one decimal for display; published rows retain source precision and aggregation. Architectures, training, and inputs differ from ours. The published StarVLA row is distinct from our StarVLA-based runs. Bold and underline mark the highest and second-highest scores, respectively, in each column across listed models, based on unrounded results. + denotes a separate VQA adaptation of Qwen3.5-4B before policy training, not cumulative additions. Split controls are described in Sec. 4.1.

Table 12: **Complete 21-benchmark VLM results for seven policy initializations.** Generalist VQA uses 18,023 examples. Spatial VQA, Pointing VQA, and Affordance & Placement VQA are capability-specific mixtures. Cross-view consistency VQA and Single-view spatial VQA are the 2,000-example splits. † denotes trajectory scores conditional on valid outputs.

Benchmark	Qwen3.5-4B	Generalist VQA	Spatial VQA	Pointing VQA	Affordance & Placement VQA	Cross-view consistency VQA	Single-view spatial VQA
BLINK	79.84	77.26	78.55	76.49	75.19	77.26	76.74
CV-Bench	83.40	85.37	84.50	84.76	83.93	86.88	85.82
PixMo-Points	55.57	58.90	52.42	55.69	54.89	55.76	54.95
PointBench	64.73	66.37	62.39	64.14	64.55	63.51	66.54
ERQA	43.75	42.25	39.75	42.75	39.50	43.50	41.75
EmbSpatial-Bench	77.61	77.34	77.64	76.51	77.61	76.76	77.55
SAT	61.33	64.00	64.00	63.33	66.00	60.00	65.33
PartAfford	19.50	21.85	13.65	17.40	21.25	16.85	15.90
RoboAfford	62.43	66.94	57.42	65.35	63.68	60.82	60.50
RoboRefIt	83.12	81.67	78.83	81.97	82.29	81.97	83.25
RefSpatial-Bench	34.12	38.66	34.66	35.39	33.57	35.38	35.74
RoboSpatial-Home	52.86	54.29	58.86	54.00	58.86	57.14	59.71
Where2Place	60.00	53.00	48.00	52.00	54.00	46.00	53.00
PIO	49.32	49.08	40.57	44.87	47.27	43.93	46.00
VABench-Point	28.67	20.44	18.00	17.78	12.28	22.67	24.33
VABench-Visual-Trace†	80.92	79.56	80.83	78.29	78.98	80.11	80.35
ShareRobot-Trajectory†	76.61	75.18	76.28	76.09	74.28	76.67	76.49
VSIBench	51.43	50.16	50.51	50.43	51.21	51.10	51.96
EgoPlan2	34.52	36.03	35.58	35.73	35.35	35.12	33.54
Cosmos	58.24	53.14	54.71	54.51	54.90	57.65	57.65
RoboVQA	45.85	46.04	43.60	44.73	45.14	43.81	45.23
Embodied VLM Avg.	56.56	56.11	53.82	54.88	55.04	54.85	55.86

F Complete VLM Evaluation

Table 12 reports all 21 benchmarks for the seven initializations with complete VLM evaluations. The four Pointing and Affordance & Placement splits enter Fig. 2 through their Embodied VLM Avg., which is 56.39 for Visible object pointing VQA, 56.17 for Visible + absent object pointing VQA, 56.55 for Interaction-point grounding VQA, and 55.51 for Destination-point grounding VQA. Appendix J separately reports the reasoning-disabled Generalist VQA and Generalist VQA + Spatial/Temporal Data recipes. Appendix K reports the complete evaluation of the Depth Geometry extension.

Embodied VLM Avg. excludes CV-Bench and RoboVQA and is computed before rounding, so averages recomputed from the displayed entries can differ by 0.01. VABench-Visual-Trace and ShareRobot-Trajectory use $100-$normalized RMSE on valid outputs. Appendix F.1 examines output validity for point answers.

Changes on the benchmarks closest to each recipe's target are also uneven. Spatial VQA improves its target benchmarks by 0.37, while Pointing VQA declines on its targets by 0.61. Affordance & Placement VQA declines by 3.35, combining a 1.50 gain on affordance benchmarks with a 7.65 drop on placement. Target-resolved scores thus carry information that the overall average hides, which motivates the condition-level analysis in Sec. 5.3.

F.1 Output Validity and Geometric Correctness

On 300 VABench-Point questions, Generalist VQA emits 305 parseable points with 63 in target masks. Affordance & Placement VQA emits 306 with 38 hits. A question can yield several points, so these counts differ from question-level parse rates. Their native scores are 20.44 and 12.28, respec-

tively. Native per-question aggregation can differ from raw point weighting. Where2Place provides a simpler example: Generalist VQA emits 100 parseable points with 53 hits, below Qwen3.5-4B's score of 60.

These examples motivate distinguishing output validity from target correctness. Multiple predicted points, cardinality errors, and native aggregation rules affect how a parseable answer contributes to a benchmark score. The complete native metrics are retained in Appendix F.

G Sensitivity to VLM Score Definition and Model Coverage

We vary benchmark definition and initialization coverage separately. The original seven complete reports and six adapted models are retained as reference cohorts. Adding Depth Geometry gives eight complete reports and seven adapted models. For each cohort, we recompute the selected 19-benchmark mean, the full 21-benchmark mean, and a 17-benchmark mean that removes the conditional trajectory scores from the selected 19. Every comparison within a cohort uses the same models. Table 13 reports the resulting correlations, and Table 14 lists the per-model scores.

Table 13: **Sensitivity of aggregate VLM and VLA associations.** Pearson r and Spearman ρ relate each VLM mean to overall LIBERO-plus success. The original cohorts are retained to show the change after adding Generalist VQA + Depth Geometry.

Benchmarks	Initializations	n	r	ρ
Selected 19	Original complete	7	-0.287	-0.393
Selected 19	Original adapted	6	+0.013	-0.029
Selected 19	Complete + depth	8	-0.490	-0.595
Selected 19	Adapted + depth	7	-0.408	-0.393
All 21	Original complete	7	-0.252	-0.321
All 21	Original adapted	6	+0.035	+0.086
All 21	Complete + depth	8	-0.475	-0.548
All 21	Adapted + depth	7	-0.398	-0.321
Non-trajectory 17	Original complete	7	-0.263	-0.393
Non-trajectory 17	Original adapted	6	+0.018	-0.029
Non-trajectory 17	Complete + depth	8	-0.468	-0.595
Non-trajectory 17	Adapted + depth	7	-0.387	-0.393

The full 21-benchmark mean retains the negative aggregate association in both complete-report cohorts. Adding the depth recipe changes the association more than switching among these benchmark definitions. For the original adapted models alone, the relation is close to zero. The sign and strength of the association thus depend on which recipes are evaluated, so a VLM average alone cannot be relied on to rank policies.

Table 14: **Per-initialization scores on fixed benchmark sets.** VLM means are recomputed from recorded benchmark entries. Policy success uses the full LIBERO-plus evaluation.

Initialization	Selected 19	All 21	Policy success (%)
Qwen3.5-4B	56.56	57.32	80.20
Generalist VQA	56.11	57.03	81.27
Spatial VQA	53.82	54.80	80.56
Pointing VQA	54.88	55.82	81.36
Affordance & Placement VQA	55.03	55.94	81.69
Cross-view consistency VQA	54.85	55.85	83.16
Single-view spatial VQA	55.86	56.78	80.81
Generalist VQA + Depth Geometry	56.76	57.67	79.60

Single-view spatial VQA retains its higher VLM average than Cross-view consistency VQA under all three definitions, while Cross-view consistency VQA retains higher policy success. That reversal

also holds after excluding any single benchmark from the full 21. The depth extension has a higher mean than Generalist VQA under all three definitions, while its policy score is lower. The Qwen3.5-4B reference also retains a lower VLM mean and higher policy success than the depth extension. These ranking reversals are robust to the choice of benchmark set.

H LIBERO-PLUS RESULTS BY TASK SUITE AND PERTURBATION

Table 15 complements the perturbation breakdown in Table 2 with results on LIBERO Goal Plus, LIBERO Spatial Plus, LIBERO Object Plus, and LIBERO 10 Plus. Cross-view consistency VQA leads on Goal and Object as well as Overall, while Affordance & Placement VQA leads on Spatial and Visible object pointing VQA leads on LIBERO 10. Visible object pointing VQA gains 78 successes over Qwen3.5-4B in LIBERO-10, most of its 85-success overall improvement. The best initialization therefore depends on the task suite, and supervision that is not the strongest overall can still be the most useful for particular task demands.

Table 15: **LIBERO-plus results by task suite.** Success rates (%) for the Table 2 models with complete suite results. All listed policies share the Qwen3.5-4B backbone and downstream training recipe. Overall pools the four suites.

	LIBERO-plus evaluation				
Model	LIBERO Goal Plus	LIBERO Spatial Plus	LIBERO Object Plus	LIBERO 10 Plus	Overall
Reference initializations					
Qwen3.5-4B	74.99	89.59	83.88	72.93	80.20
+ Generalist VQA	78.73	89.51	82.53	74.75	81.27
Capability-specific VQA					
+ Spatial VQA	77.88	88.72	83.84	72.25	80.56
+ Pointing VQA	77.46	89.47	85.27	73.72	81.36
+ Affordance & Placement VQA	77.23	**90.55**	84.11	75.43	81.69
Spatial VQA splits					
+ Cross-view consistency VQA	**80.97**	90.01	**86.50**	75.55	**83.16**
+ Single-view spatial VQA	77.34	88.30	82.64	75.39	80.81
Pointing VQA splits					
+ Visible object pointing VQA	74.80	89.84	84.11	**76.02**	81.05
+ Visible + absent object pointing VQA	76.80	85.89	83.80	75.31	80.36
Affordance & Placement VQA splits					
+ Interaction-point grounding VQA	77.50	89.09	84.47	71.26	80.46
+ Destination-point grounding VQA	78.50	87.80	81.49	72.53	79.98

Bold marks the highest score in each column across the listed models. + denotes a separate VQA adaptation of Qwen3.5-4B before policy training.

H.1 PERTURBATION-LEVEL ANALYSES

Similar Camera gains can accompany different Robot outcomes. The depth extension provides another recipe with Camera gains. Generalist VQA + Depth Geometry improves Camera by 5.88% over Qwen3.5-4B, while Robot changes by -2.00%. Its aggregate Camera score is close to Cross-view consistency VQA, differing by only 1.19%, but its Robot score is 11.23% lower (Fig. 1, Finding 2). Camera and Robot thus distinguish different transfer profiles even when both recipes teach geometric information. Appendix J uses Generalist VQA as the direct reference for the depth extension and further locates the depth recipe's strongest camera benefit in LIBERO-10.

Pointing VQA: replacing positives changes the robustness profile. Replacing half of the positive examples with absent-object questions changes overall success from 81.05% to 80.36%. To understand this difference, we compare the two recipes within each perturbation. Visible + absent object pointing VQA is lower on Camera by 6.82% and Language by 2.93%, but higher on Robot by 2.58%. The camera difference has the same sign in every suite. The robot-state advantage is concentrated in

Goal, which contributes 52 additional successes while the other three suites together lose 12. Thus adding absence questions does not produce a uniform change in robustness. Language perturbations retain present targets while rewriting the instruction, making positive localization a relevant training target. Under a fixed budget, trading positive examples for absence questions therefore costs camera and language robustness while strengthening robot-state robustness in Goal.

Affordance & Placement VQA: similar totals conceal different target-specific profiles. Interaction-point grounding VQA and Destination-point grounding VQA share a single-point answer interface and the same example budget. Their overall scores differ by only 0.48%. However, Interaction-point grounding VQA leads on Camera by 3.56% and on the Object suite by 2.98%, while Destination-point grounding VQA leads on Robot by 2.19%. Resolving the latter difference by suite reveals 53 additional successes in LIBERO-10 and 19 fewer across the other suites. Interaction locations and destination points therefore accompany different transfer profiles despite having the same answer format. The broader Affordance & Placement VQA recipe, which combines grasp locations and placement targets, gains 6.39% on Robot relative to Qwen3.5-4B. Together, these comparisons show why both target content and task conditions are needed to interpret affordance transfer.

Broader supervision helps some forms of robustness but not others. We next examine the remaining categories to determine whether the Camera and Robot preferences generalize to other perturbations. Generalist VQA leads Language, Light, Background, and Layout among our main-study initializations, with gains over Qwen3.5-4B of 1.17%, 3.77%, 1.58%, and 3.34%, respectively. Its coverage includes localization, interaction, state, planning, and temporal questions. Spatial VQA also improves Layout by 2.82%, whereas Destination-point grounding VQA changes by -0.20%. Layout robustness therefore benefits more from broad spatial context than from destination targets alone.

Sensor noise exhibits a different pattern. Nine of the ten main-study adapted recipes score below Qwen3.5-4B on Noise. Single-view spatial VQA is the exception, with a 0.12% increase corresponding to two successes. Spatial VQA and Affordance & Placement VQA lose 6.93% and 5.81%, respectively, despite their Robot improvements. Hence a recipe that improves control under changed geometry need not improve control under degraded visual evidence. Appendix I examines whether individual VLM benchmarks can separate these condition-specific benefits.

I Stability of VLM–Policy Associations

Checking benchmark–condition associations with an additional recipe. To examine information hidden by the aggregate, we compare all 21 VLM benchmarks against the seven perturbation categories on the seven fully evaluated main-study initializations. RoboRefIt has opposite associations with Noise and Layout, while EgoPlan2 closely aligns with Language. We keep these benchmark–condition pairs fixed and add the depth recipe, testing how their associations change without selecting new pairs to fit the additional result.

Figure 6(b) shows that the two RoboRefIt associations retain their directions. With the additional recipe, Pearson correlation changes from 0.78 to 0.63 for Noise and from -0.70 to -0.63 for Layout. The same localization benchmark therefore remains associated with different policy orderings under visual degradation and changed object arrangements.

EgoPlan2 provides a more discriminating check (Fig. 6(c)). The depth recipe has the highest EgoPlan2 score among these eight initializations, yet its Language success is the second lowest. Pearson correlation falls from 0.96 to 0.48, and Spearman correlation from 0.89 to 0.43. A strong benchmark–condition association among the original recipes can thus weaken sharply once a new kind of supervision is added.

Perturbation profiles explain why the aggregate relation changes. The depth recipe combines a high VLM average with low Noise success. Consequently, the aggregate VLM–Noise correlation weakens from 0.40 to 0.17 when it is added to the eleven original initializations (Fig. 6(a)). The aggregate association with overall policy success instead becomes more negative, from -0.37 to -0.47. These different responses to the same added recipe show why a single correlation is an incomplete account of transfer. Appendix G separates changes in benchmark selection from changes

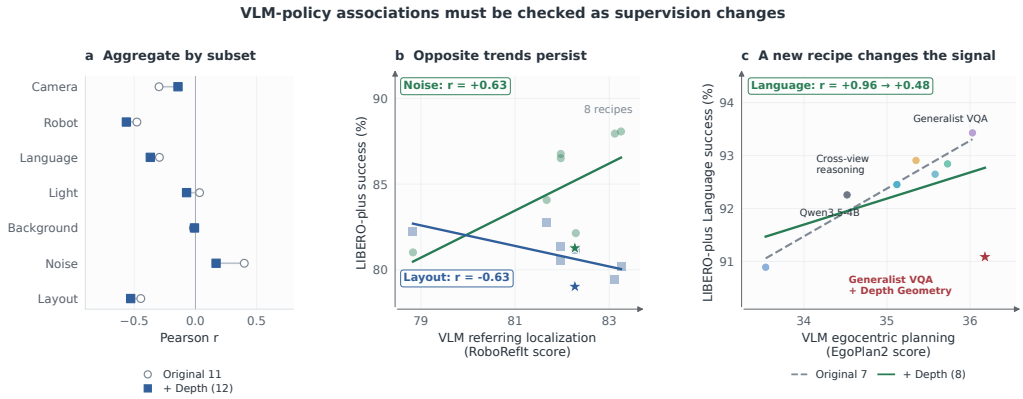

Figure 6: **VLM–policy associations as supervision changes.** (a) Aggregate VLM–subset correlations before and after adding the depth extension to the eleven main-study initializations. (b) RoboRefIt retains opposite associations with Noise and Layout across the eight complete reports. Stars mark the depth recipe. (c) The original seven-report EgoPlan2–Language fit (dashed) and the updated eight-report fit (solid). The depth recipe has the highest EgoPlan2 score but the second-lowest Language success, weakening the association. Benchmark–condition pairs are kept fixed when the depth recipe is added.

in initialization coverage. A useful VLM measure should therefore be tied to a stated control condition and rechecked as the supervision pool grows.

J Reasoning and Additional Supervision

We compare reasoning-enabled and disabled adaptation on Generalist VQA, then extend the latter with Spatial/Temporal Data and the former with Depth Geometry. Both extensions retain the 18,023 Generalist examples.

Table 16: **Added supervision improves the targeted condition but lowers overall success.** Each extension keeps the 18,023 Generalist VQA examples and is compared with the Generalist VQA policy trained under the same VLM reasoning setting. Success rates are in %.

		LIBERO-10	All suites		
VQA supervision	#VQA	Camera↑	Camera↑	Noise↑	Overall↑
Reasoning enabled					
Generalist VQA	18,023	38.66	60.98	84.07	81.27
+ Depth Geometry	+2,000	48.69 *(+10.02)*	66.60 *(+5.63)*	81.26 *(−2.81)*	79.60 *(−1.67)*
Reasoning disabled					
Generalist VQA	18,023	38.42	63.48	86.57	80.95
+ Spatial/Temporal Data	+5,112	47.02 *(+8.59)*	62.91 *(−0.56)*	80.26 *(−6.31)*	79.10 *(−1.84)*

Reasoning-disabled Generalist VQA raises the direct-answer capability average (55.38 to 57.14) but not overall policy success (81.27% to 80.95%). Both data extensions improve LIBERO-10/Camera over their Generalist controls, yet overall success falls to 79.60% and 79.10%, respectively (Table 16). Per-benchmark gains, coverage, and suite–perturbation trade-offs are analyzed below and in Appendix K.

Interpretation of the policy profiles. At fixed Generalist VQA data, disabling reasoning raises the capability average from 55.38 to 57.14 and the trajectory score from 77.37 to 80.67. Camera and Noise each improve by about 2.50%, but Language, Background, and Layout decline, leaving

overall success at 80.95% rather than 81.27%. Better direct answers therefore do not determine the overall policy ranking.

With reasoning disabled, Spatial/Temporal Data raises the trajectory score from 80.67 to 82.65. PixMo and PointBench improve by 7.75 and 5.06, while valid trajectory outputs increase (Table 18). LIBERO-10 Camera and Robot improve by 8.59% and 6.11%, whereas Spatial/Camera drops by 10.37%. Noise, Light, and Language account for 93.5% of the net success-count decrease, leaving 79.10% overall. The LIBERO-10 gains are thus offset by lost successes elsewhere, the preservation effect analyzed in Sec. 5.4.

Both data extensions improve LIBERO-10/Camera relative to their own Generalist controls, by 10.02% for Depth Geometry and 8.59% for Spatial/Temporal Data. Two different kinds of added supervision thus converge on the same condition, which makes LIBERO-10/Camera a clear target for geometric and temporal supervision. The depth-specific VLM and condition analysis is in Appendix K.

Composition and training. Generalist VQA + Spatial/Temporal Data retains all 18,023 Generalist VQA examples and adds 5,112 examples from the three sources in Table 16. The 2,000 MolmoAct examples comprise 1,000 Home and 1,000 Tabletop examples, rendered as multi-point 2D trajectories in ShareRobot answer format before difficulty screening. The trajectory pool increases from 23 to 2,023 examples, or from approximately 0.13% to 8.74% of training examples. The augmented pool contains 11,912 choice, 7,200 point, 2,000 point-or-no-object, and 2,023 trajectory examples. The name Spatial/Temporal Data denotes this specific supplemental pool, distinct from the Spatial VQA capability-specific recipe in Table 2.

All runs start from Qwen3.5-4B. Difficulty screening uses four non-thinking rollouts, retaining examples with one to three correct answers. The 18,023-example and 23,135-example adaptations use 1,127 and 1,446 updates. The reasoning-enabled runs use four rollout and four training devices. The reasoning-disabled runs use six rollout and two training devices. Policy training uses the common LIBERO 4-in-1 recipe for 30,000 steps.

VLM metrics. The three recipes with policy results have complete 21-benchmark reports with thinking disabled at evaluation. The ablation's capability average uses the 18 non-trajectory benchmarks excluding RoboVQA BLEU. Its trajectory score averages the two scores obtained as $100 - \mathrm{RMSE}/10$. It is reported separately from the capability average. Table 17 summarizes the VLM and policy scores, and Table 18 reports the native RMSE and valid prediction counts for the three recipes with policy results. Errors are conditional on each model's valid predictions. These results use the ablation evaluation batch.

Table 17: **VLM and policy scores for the reasoning ablation.** VLM scores use the common evaluation batch described above. The capability mean covers 18 non-trajectory benchmarks excluding RoboVQA. The trajectory score averages two conditional native scores. Bold marks the maximum among these three recipes.

VQA supervision	Training reasoning	Capability average	Trajectory score	LIBERO-plus success (%)
Generalist VQA	On	55.38	77.37	**81.27**
Generalist VQA	Off	**57.14**	80.67	80.95
+ Spatial/Temporal Data	Off	54.56	**82.65**	79.10

At the suite level, Generalist VQA with reasoning disabled achieves 78.35%, 90.09%, 83.36%, and 72.49% on Goal, Spatial, Object, and LIBERO-10. Adding Spatial/Temporal Data changes these scores to 75.45%, 87.51%, 81.93%, and 72.01%, respectively. The nearly unchanged LIBERO-10 aggregate combines 60 additional successes under Camera and Robot with 73 fewer successes under Language and Noise. The other categories account for the remainder. Thus suite totals and perturbation totals offer complementary views of the added data.

Table 18: **Trajectory precision and output coverage in the ablation.** + Spatial/Temporal Data always augments Generalist VQA. Lower RMSE is better. Valid counts are reported alongside conditional errors.

	Training	VABench Visual Trace		ShareRobot Trajectory	
VQA supervision	reasoning	RMSE	Valid	RMSE	Valid
Generalist VQA	On	204.37	250/300	248.24	193/200
Generalist VQA	Off	167.05	273/300	219.45	192/200
Generalist VQA + Spatial/Temporal Data	Off	164.09	292/300	183.00	200/200

Table 19: **Perturbation profiles for the reasoning and data ablation.** Success rates (%) use the same categories as Table 2.

VQA supervision	Training reasoning	Camera	Robot	Language	Light	Background	Noise	Layout
Generalist VQA	On	60.98	61.87	93.43	98.16	97.77	84.07	82.75
+ Depth Geometry	On	66.60	57.55	91.09	95.01	96.28	81.26	79.02
Generalist VQA	Off	63.48	61.23	91.35	97.72	95.07	86.57	80.39
Generalist VQA + Spatial/Temporal Data	Off	62.91	61.35	89.26	94.22	95.63	80.26	79.67

K Depth Geometry Extension

Conditional benefit versus overall transfer. Relative to reasoning-enabled Generalist VQA, Depth Geometry raises Where2Place from 53.00 to 63.00, RoboAfford from 66.94 to 69.08, and PartAfford from 21.85 to 24.05, while RefSpatial-Bench and VABench-Point decline by 0.68 and 1.94. On LIBERO-10/Camera, policy success rises from 38.66% to 48.69%, the highest among the reported policies for that condition. Camera improves by 5.63% across suites, but all six other perturbation categories decline and overall success is 79.60%. The interaction gains thus come with the strongest camera robustness on LIBERO-10 among all reported policies.

Composition and adaptation. Generalist VQA + Depth Geometry retains the 18,023 Generalist VQA examples and adds 2,000 depth geometry examples. The added pool covers pixel-pair 3D distances, relative depth ordering, exact depth, gripper/target depth, and temporal distance. It contains 1,000 plain-answer and 1,000 multiple-choice examples. The separately recorded reward types comprise 680 numeric-range, 800 choice, and 520 point examples.

Adaptation starts from Qwen3.5-4B with thinking enabled, using the checkpoint after 1,200 updates. It uses six rollout and two training devices, while Generalist VQA with thinking enabled uses 1,127 updates with four rollout and four training devices. All policies use the common 30,000-step LIBERO training recipe. The depth pool supplies VQA targets, not a dense depth-map prediction objective.

Direct-answer VLM evaluation. Table 20 reports all 21 benchmarks for Qwen3.5-4B, Generalist VQA, and the depth extension. Changes are measured against Generalist VQA, the reference for the data extension. The selected 19 mean excludes CV-Bench and RoboVQA. Both follow the convention of Table 12.

VABench-Visual-Trace has 192 valid outputs among 300 questions, and ShareRobot-Trajectory has 172 among 200. Relative to Generalist VQA, the depth extension improves RoboAfford, PartAfford, RoboSpatial-Home, and Where2Place, while RefSpatial-Bench and VABench-Point decline. The gains therefore concentrate on interaction and placement benchmarks.

Policy profiles and aligned outcomes. Table 21 gives all task–perturbation intersections. The corresponding suite scores are 76.96% on Goal, 88.47% on Spatial, 81.45% on Object, and 72.01% on LIBERO-10. Overall success is 79.60%. All seven aggregate perturbation scores appear in Table 19.

Table 20: **Complete VLM evaluation of the depth extension.** Generalist denotes Generalist VQA. Depth denotes Generalist VQA + Depth Geometry. Changes are native score differences from Generalist VQA. † denotes conditional trajectory scores, for which valid-output coverage is reported below.

Benchmark	Qwen3.5-4B	Generalist	Depth	Change
BLINK	79.84	77.26	76.74	-0.52
CV-Bench	83.40	85.37	85.18	-0.19
PixMo-Points	55.57	58.90	58.23	-0.67
PointBench	64.73	66.37	63.13	-3.24
ERQA	43.75	42.25	43.50	+1.25
EmbSpatial-Bench	77.61	77.34	75.08	-2.26
SAT	61.33	64.00	61.33	-2.67
PartAfford	19.50	21.85	24.05	+2.20
RoboAfford	62.43	66.94	69.08	+2.14
RoboRefIt	83.12	81.67	82.27	+0.60
RefSpatial-Bench	34.12	38.66	37.98	-0.68
RoboSpatial-Home	52.86	54.29	56.00	+1.71
Where2Place	60.00	53.00	63.00	+10.00
PIO	49.32	49.08	47.30	-1.78
VABench-Point	28.67	20.44	18.50	-1.94
VABench-Visual-Trace†	80.92	79.56	81.09	+1.53
ShareRobot-Trajectory†	76.61	75.18	75.63	+0.45
VSIBench	51.43	50.16	51.44	+1.28
EgoPlan2	34.52	36.03	36.18	+0.15
Cosmos	58.24	53.14	57.84	+4.70
RoboVQA	45.85	46.04	47.55	+1.51
Selected 19 mean	56.56	56.11	56.76	+0.64
Full 21 mean	57.32	57.03	57.67	+0.65

Table 21: **Full task–perturbation profile of the depth extension.** Each cell reports success (%) followed by its absolute change from Qwen3.5-4B. Changes use exact counts before rounding. The Camera gains are concentrated in Goal and LIBERO-10, while Noise declines in all four suites.

Suite	Camera	Robot	Language	Light	Background	Noise	Layout
Goal	71.81 (+8.82)	56.48 (-1.96)	82.68 (-2.93)	98.57 (+14.34)	97.86 (+6.41)	84.43 (-3.96)	61.41 (-1.88)
Spatial	71.81 (-0.80)	69.71 (-0.29)	96.67 (-2.82)	95.21 (-4.11)	98.84 (-0.39)	93.73 (-0.85)	96.62 (+1.04)
Object	75.25 (-0.76)	47.74 (-8.04)	96.61 (+8.19)	98.32 (-1.35)	100.00 (+0.40)	83.65 (-11.85)	81.39 (-0.50)
LIBERO-10	48.69 (+15.27)	57.76 (+2.54)	89.30 (-6.27)	87.59 (-6.20)	89.27 (-5.88)	66.59 (-8.69)	78.21 (+0.00)

On LIBERO-10/Camera, the depth extension achieves 48.69% compared with 33.41% for Qwen3.5-4B, 38.66% for Generalist VQA, and 42.24% for Cross-view consistency VQA. Generalist VQA without training reasoning reaches 38.42%, and its Spatial/Temporal Data extension reaches 47.02%. The strongest main-study task split in this condition is Visible object pointing VQA at 42.96%. The additional Cross-view direct-generation policy reaches 44.39% (Appendix L). The depth recipe is highest among all policies reported for this condition, despite its lower overall success.

Aligning configurations yields 7,367 successes shared by Qwen3.5-4B and the depth recipe, 677 reference-only successes, 617 depth-only successes, and 1,369 shared failures. The 1,294 differing outcomes account for 12.90% of configurations, so a small aggregate change can hide substantial turnover in which configurations succeed. Relative to Generalist VQA, the depth recipe gains 509 successes and loses 676, mostly outside camera perturbations, for a net decline of 167. These counts support the preservation analysis in Sec. 5.4.

Stability of benchmark–condition associations. We keep the benchmark–condition pairs selected from the original seven reports. Adding Depth Geometry changes EgoPlan2–Language from $r = 0.96$ to $r = 0.48$, while RoboRefIt retains its opposite associations with Noise and Layout.

L Detailed Cross-view Interface Comparison

Construction. The direct-generation conversion retains the 2,000 sample identities, paired images, question stems, order, and choice-based difficulty metadata. It removes the candidate options and replaces the correct option letter with its answer phrase. The pool comprises 673 phase-consistency, 667 multi-view-reasoning, and 660 task-consistency examples. Both adaptations use thinking-enabled GRPO for 125 updates. Direct generation uses a `text_phrase` verifier with token-F1 threshold 0.7 and a 24-word limit.

Direct generation has 8,111 successes versus MCQ's 8,341, a 230-success deficit. Camera is 5.88% above Qwen3.5-4B and only 1.19% below MCQ, whereas Robot is 8.19% below MCQ. The 127-success Robot deficit explains 55.2% of the overall gap. Relative to Qwen3.5-4B, direct generation gains 128 fewer new successes and loses 102 more previous successes.

Tables 22, 23, and 24 compare the MCQ policy from the main study with the direct-generation policy on VLM benchmarks, policy success, and success preservation.

Direct-generation trajectory coverage is 275/300 for VABench-Visual-Trace and 188/200 for ShareRobot-Trajectory. ERQA multi-view accuracy is 32.43% (12/37) for direct generation and 27.03% (10/37) for MCQ.

Despite lower category totals, direct generation exceeds MCQ on Object/Camera (84.85% versus 83.33%) and LIBERO-10/Camera (44.39% versus 42.24%). It is lower on Goal/Camera (69.36% versus 73.53%) and Spatial/Camera (69.15% versus 73.67%). Robot success is lower than MCQ in every suite. The gap therefore concentrates in robot-state perturbations, while most of the camera benefit is retained.

M Real-Episode Source Comparisons

Source-matched question budgets. The DROID/RoboCasa Generalist recipe follows the original 18,023-item family histogram. It contains 13,940 DROID questions from 2,959 recorded episodes and 4,083 retained RoboCasa questions from 2,162 simulated episodes. Its reward interfaces comprise 8,800 choices, 7,200 points, 2,000 point-or-no-object answers, and 23 trajectories. The DROID Cross-view recipe contains 2,000 choices from 842 recordings, with 673 phase-consistency, 667 multi-view reasoning, and 660 task-consistency questions. It matches the size and answer interface of the RoboCasa Cross-view recipe, so both comparisons center on where the episodes come from. Extraction details are given in Appendix B.

Aligned downstream outcomes. The DROID + RoboCasa Generalist policy has 680 gained and 480 lost successes relative to Qwen3.5-4B. DROID Cross-view has 707 gained and 536 lost successes. Outcomes are aligned by configuration across all four suites. The DROID + RoboCasa Generalist policy records 8,244 successes across 10,030 configurations.

Table 22: **All 21 VLM benchmarks.** Differences are direct generation minus MCQ, from displayed scores. † denotes a score conditional on valid trajectory predictions.

Benchmark	Qwen3.5-4B	MCQ	Direct	Difference
BLINK	79.84	77.26	77.26	+0.00
CV-Bench	83.40	86.88	87.41	+0.53
PixMo-Points	55.57	55.76	56.40	+0.64
PointBench	64.73	63.51	65.07	+1.56
ERQA	43.75	43.50	40.25	-3.25
EmbSpatial-Bench	77.61	76.76	77.28	+0.52
SAT	61.33	60.00	66.67	+6.67
PartAfford	19.50	16.85	16.80	-0.05
RoboAfford	62.43	60.82	61.24	+0.42
RoboRefIt	83.12	81.97	81.95	-0.02
RefSpatial-Bench	34.12	35.38	35.02	-0.36
RoboSpatial-Home	52.86	57.14	57.43	+0.29
Where2Place	60.00	46.00	51.50	+5.50
PIO	49.32	43.93	45.23	+1.30
VABench-Point	28.67	22.67	27.00	+4.33
VABench-Visual-Trace†	80.92	80.11	80.01	-0.10
ShareRobot-Trajectory†	76.61	76.67	76.65	-0.02
VSIBench	51.43	51.10	50.70	-0.40
EgoPlan2	34.52	35.12	32.40	-2.72
Cosmos	58.24	57.65	56.27	-1.38
RoboVQA	45.85	43.81	45.50	+1.69
Embodied VLM Avg. (19)	56.56	54.85	55.53	+0.68

Table 23: **Policy results by suite and perturbation.** Difference is direct generation minus MCQ (%), and ΔS is the corresponding difference in success counts.

Condition	N	Qwen3.5-4B	MCQ	Direct	Difference	ΔS
Goal	2,591	74.99	80.97	78.23	-2.74	-71
Spatial	2,402	89.59	90.01	89.13	-0.87	-21
Object	2,518	83.88	86.50	83.76	-2.74	-69
LIBERO-10	2,519	72.93	75.55	72.81	-2.74	-69
Camera	1,599	60.73	67.79	66.60	-1.19	-19
Robot	1,550	59.55	68.77	60.58	-8.19	-127
Language	1,537	92.26	92.45	89.59	-2.86	-44
Light	1,142	94.40	97.37	95.53	-1.84	-21
Background	1,076	96.19	95.91	95.26	-0.65	-7
Noise	1,601	87.95	86.51	85.88	-0.62	-10
Layout	1,525	79.41	81.38	81.25	-0.13	-2
Overall	10,030	80.20	83.16	80.87	-2.29	-230

Table 24: **Success preservation relative to Qwen3.5-4B.** Each row is aligned with Qwen3.5-4B by configuration.

Recipe	Shared successes	Lost successes	New successes	Net
MCQ	7,581	463	760	+297
Direct generation	7,479	565	632	+67
Direct − MCQ	-102	+102	-128	-230

Table 25: **Complete LIBERO-plus source-pair results.** Success rates (%) cover all seven perturbation categories and their count-weighted Overall. VLM Avg. is the selected 19-benchmark mean.

Model / source	Overall	Camera	Robot	Language	Light	Background	Noise	Layout	VLM Avg.
Qwen3.5-4B	80.20	60.73	59.55	92.26	94.40	96.19	87.95	79.41	56.56
Generalist VQA: 18,023 questions									
RoboCasa	81.27	60.98	61.87	93.43	98.16	97.77	84.07	82.75	56.11
DROID + RoboCasa	82.19	67.10	62.00	91.54	97.20	96.28	88.07	81.77	55.51
Cross-view VQA: 2,000 questions									
RoboCasa	83.16	67.79	68.77	92.45	97.37	95.91	86.51	81.38	54.85
DROID	81.90	69.42	63.94	92.65	96.76	96.28	83.51	79.48	55.45

N Extended Discussion

Connect data selection to operating conditions. The results support a staged approach to choosing supervision. The deployment setup or an existing reference policy identifies conditions of interest, such as changing camera views, object layouts, or instructions. The content of candidate VQA pools and their VLM evaluations then guide which recipes to evaluate. The observed profiles offer concrete reference points: Cross-view consistency VQA helps both camera and robot-state changes across suites, Generalist VQA supports Language, Light, and Background robustness, and the depth extension yields particularly strong LIBERO-10 camera performance. Closed-loop results determine whether such benefits carry to the intended conditions and whether other successes are retained. Policy evaluation on the target conditions then makes the final choice.

Use a profile of VLM evaluations and check new recipes. Additional supervision can improve measurable capabilities: depth geometry improves selected affordance and placement benchmarks over Generalist VQA, while the spatial/temporal extension improves trajectory prediction. These gains are useful evidence about what adaptation teaches the VLM. Their value for control must also account for the conditions in which the policy uses that information. The depth check of the EgoPlan2–Language association further shows that a benchmark–condition relationship can change when a new supervision recipe is introduced. Reporting target accuracy, output validity, and conditional policy outcomes provides a more informative basis for selecting data than a single VLM average.

Use reusable evidence to investigate supervision choices. The construction pipeline makes the target and supporting evidence explicit for each question. This allows tasks involving localization, interaction, and motion to reuse an episode while retaining distinct semantics. The empirical comparisons identify concrete choices for such studies, including the balance of positive and absent-object questions, interaction and destination targets, and depth geometry and trajectory supervision. The cross-view interface comparison also highlights candidate availability and answer verification as explicit recipe choices: removing options while changing the answer and verifier retains much of the observed Camera benefit but less of the Robot benefit. Shared evidence can support more closely matched data comparisons and a clearer assessment of the resulting gains and losses.