ROBOATOM: LARGE-SCALE ATOMIC-ACTION SUPERVISION FOR ROBOT LEARNING

Anonymous authors
Paper under double-blind review

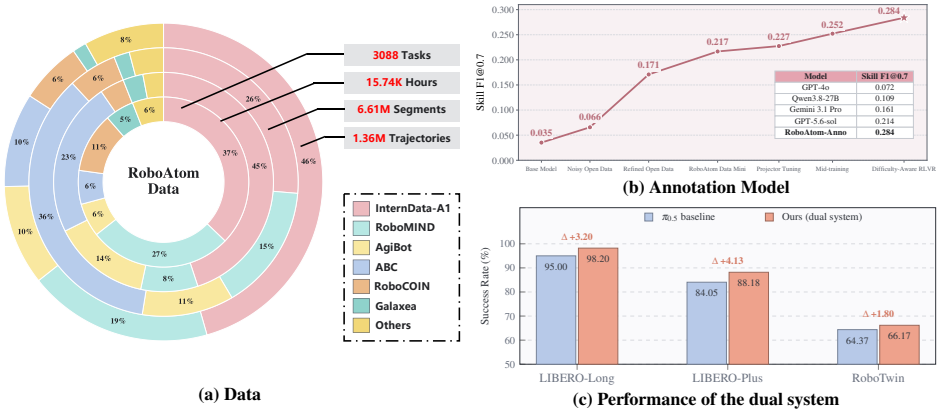

Figure 1: RoboAtom overview. (a) RoboAtom-Data comprises 1.36M trajectories, 6.61M segments, 15.7K hours, and 3,088 tasks. (b) Annotation performance across representative training configurations. (c) The dual-system method improves downstream success rates on LIBERO-Long (Liu et al., 2023), LIBERO-Plus Overall (Fei et al., 2025), and the RoboTwin (Chen et al., 2025d) Clean/Random average.

ABSTRACT

Fine-grained action annotations can support skill reuse and compositional generalization in robot policies. However, robot datasets typically provide task-level instructions without subtask annotations. Although some include subtask labels, inconsistent formats and conventions hinder cross-source joint training. We propose RoboAtom-Data, which adds unified, time-aligned atomic-action annotations to existing data from multiple sources, covering 1.36 million trajectories, 6.6 million skill-labeled segments, and 15,736.6 hours. Our pipeline combines a unified protocol, large-scale manual annotation, propagation across repeated executions with the same ordered skill sequence, and multi-stage quality control. We evaluate the dataset through automatic atomic-action annotation and robot control. We construct RoboAtom-Bench to standardize evaluation of skill recognition and temporal boundary localization. Trained on the annotated data, RoboAtom-Anno achieves higher annotation quality at lower inference cost than all evaluated open- and closed-source models on both RoboAtom-Bench and the external WGO-Bench. For robot control, atomic-action supervision paired with hierarchical execution improves task success over task-level baselines on LIBERO-Long, LIBERO-Plus, and RoboTwin. We will release the data, benchmark, model, and construction pipeline to support further research on skill-level robot learning.

1 INTRODUCTION

Time-aligned atomic-action annotations expose intermediate trajectory structure and can support skill reuse and compositional generalization. For example, "put the cup into the drawer" comprises picking up the cup, opening the drawer, and placing the cup inside. Unlike a task-level instruction that specifies the final goal, these annotations attach a skill label, an action description, and temporal boundaries to each action, providing a shared interface for automatic annotation and skill-conditioned execution. Most robot datasets provide task-level instructions without intermediate-action annotations. Open X-Embodiment (Open X-Embodiment Collaboration, 2023),

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.

DROID (Khazatsky et al., 2024), RoboMIND (Wu et al., 2025a), and InternData-A1 (Tian et al., 2025) broaden coverage across embodiments, tasks, and scenes, but existing subtask labels differ in format, vocabulary, granularity, and temporal coverage. For example, some Galaxea (Jiang et al., 2025) annotations merge Pick and Place. These inconsistencies hinder joint training and motivate shared skill definitions and segmentation rules.

To address the lack of consistent segment-level supervision across robot datasets, we introduce **RoboAtom**, a unified resource and construction pipeline for time-aligned atomic-action annotations. **RoboAtom-Data** augments trajectories from 14 sources with standardized skill categories, action descriptions, and temporal boundaries, covering 1.36 million trajectories, 6.6 million segments, and 15,736.6 hours (Table 1; Figure 2). Its pipeline combines manual seed annotation, constrained same-task propagation, skill-specific physical cues, and multi-stage quality control. We validate this supervision through automatic annotation and robot control. **RoboAtom-Bench** standardizes skill recognition and temporal boundary evaluation. **RoboAtom-Anno**, trained on the annotated data, achieves higher annotation quality at lower inference cost than all evaluated open- and closed-source models on task-disjoint RoboAtom-Bench and external WGO-Bench (Macrodata Labs, 2026). Atomic-action supervision paired with hierarchical execution also improves task success over task-level baselines on LIBERO-Long (Liu et al., 2023), LIBERO-Plus (Fei et al., 2025), and RoboTwin (Chen et al., 2025d). The main contributions of this paper are:

- **RoboAtom-Data: unified atomic-action supervision at scale.** We supplement heterogeneous trajectories with standardized action categories, descriptions, and temporal boundaries, producing 1.36 million annotated trajectories and 6.6 million skill-labeled segments through manual annotation, constrained same-task propagation, and multi-stage quality control.
- **RoboAtom-Bench: standardized evaluation.** We build a benchmark of 500 expert-reviewed trajectories and 4,018 ground-truth segments for evaluating action recognition and temporal localization under short, long, and multi-view conditions.
- **Downstream evidence.** RoboAtom-Anno achieves higher annotation quality at lower inference cost than the evaluated baselines on RoboAtom-Bench and WGO-Bench. Atomic-action supervision with hierarchical execution improves task success by **3.20**, **4.13**, and **2.28** percentage points on **LIBERO-Long**, **LIBERO-Plus Overall**, and **RoboTwin Random**, respectively.

2 Related Work

2.1 Robot Manipulation Datasets and Existing Subtask Annotations

Robot datasets support policy generalization through diverse embodiments and environments. Open X-Embodiment (Open X-Embodiment Collaboration, 2023) pools multi-institution experience; BridgeData V2 (Walke et al., 2023) and DROID (Khazatsky et al., 2024) emphasize real-world scene and operator diversity; RH20T (Fang et al., 2023) adds force and audio; AgiBot World (AgiBot World Contributors et al., 2025) and RoboMIND (Wu et al., 2025a) scale standardized collection; and Galaxea Open-World (Jiang et al., 2025) covers mobile manipulation. LIBERO (Liu et al., 2023), RoboCasa (Nasiriany et al., 2024), RoboTwin 2.0 (Chen et al., 2025d), and InternData-A1 (Tian et al., 2025) provide scalable simulated data through procedural generation and domain randomization.

AgiBot World (AgiBot World Contributors et al., 2025), Galaxea (Jiang et al., 2025), RoboInterData (Li et al., 2026), and BEHAVIOR-1K (Li et al., 2024a) also provide subtask annotations (Table 1), but inconsistent vocabularies, granularity, and temporal coverage prevent direct pooling. Galaxea sometimes merges Pick and Place, while AgiBot's Navigate maps to our Move. RoboAtom-Data unifies vocabulary and temporal alignment across existing trajectories from 14 sources.

2.2 Automation and Evaluation of Atomic-Action Annotation

Atomic-action annotation requires both semantic recognition and temporal localization. Recent work on video temporal grounding, including VTimeLLM (Huang et al., 2023), TimeChat (Ren et al., 2024), TimeLens (Zhang et al., 2026), and TimeLens2 (Zhu et al., 2026), studies how multimodal LLMs generate time-localized evidence and improve temporal localization quality. These methods motivate automatic annotation, while our setting additionally requires a robot-specific

atomic-action vocabulary and separate evaluation of semantic correctness and temporal boundaries. For robot trajectories, Cortex (Peng et al., 2026) uses reference segmentations together with state-action and visual features to infer subtask boundaries in other executions. This reuse reduces annotation effort, but variations in execution speed and pauses can complicate boundary alignment. WGO-Bench (Macrodata Labs, 2026) evaluates video-to-subtask annotation through temporal segmentation, label semantics, and joint matching. Its results identify temporal localization as a major bottleneck, with brief actions particularly prone to omission. These challenges motivate reliable boundary propagation and evaluation protocols that jointly assess action semantics and temporal accuracy.

2.3 Downstream Applications of Atomic-Action Supervision

Intermediate representations connect task instructions to continuous actions in hierarchical robot learning. RT-H (Belkhale et al., 2024), Embodied Chain-of-Thought (Zawalski et al., 2024), and $\pi_{0.5}$ (Physical Intelligence et al., 2025) use language motions, intermediate reasoning, or subtask prediction to guide action generation. Hi Robot (Shi et al., 2025), GR00T N1 (NVIDIA et al., 2025), and RoboBrain (Ji et al., 2025) separate high-level planning from low-level execution, while Cortex (Peng et al., 2026) uses canonical skills as its interface. Beyond control, REDS (Kim et al., 2025) and SARM (Chen et al., 2025c) use segmented demonstrations or subtask annotations to learn stage-aware rewards.

These works demonstrate the value of intermediate-action supervision, but typically rely on existing annotations or construct them for specific models and datasets. Cross-source annotation consistency and standalone quality evaluation remain less explored. RoboAtom-Data addresses this upstream need by constructing unified, time-aligned annotations across heterogeneous robot datasets and evaluating their quality and downstream utility.

3 RoboAtom-Data

RoboAtom comprises the dataset (RoboAtom-Data), the benchmark (RoboAtom-Bench), and the annotation model (RoboAtom-Anno). Section 3.1 defines the supervision unit, describes the data sources, and compares RoboAtom-Data with existing resources. Sections 3.2–3.4 then follow the construction pipeline in Figure 3: human-in-the-loop seed annotation and benchmark construction, skill-aware same-task extension (PhysExtend), and quality control. RoboAtom-Data is released as a reusable resource; Section 4.1 describes annotation-model training on a task-sampled subset, and Section 5.1 evaluates the model on the task-disjoint RoboAtom-Bench.

3.1 Data Definition, Sources, and Comparison

Each atomic action is annotated as a segment with three fields: a skill category, an action description, and start and end frames. The skill category is a label shared across sources, the action description retains the manipulated object and action details, and the start and end frames align both with the trajectory. A new segment starts only when the dominant skill changes; preparation, arm repositioning, and transitions are absorbed into the adjacent segment (Appendix A.1).

The released set retains the full 28-class skill vocabulary; semantically similar actions are grouped into the same category, with the specific meaning kept in the action description. RoboAtom-Bench uses 15 primary skills among them, which cover 99.4% of the skill segments in the released set. The full vocabulary, mapping rules, and the basis for category selection are given in Appendix A.3.

RoboAtom-Data contains 1,363,106 trajectories, 6,606,906 skill-labeled segments, and 15,736.6 hours of data from 14 sources, including internal data. We collect real-robot and high-fidelity simulated data, retain tasks with multiple repeated demonstrations after screening by embodied-AI practitioners, and convert all data to the LeRobot v2.1 format (Cadene et al., 2026). As shown in Table A.2, the Real sources total about 8,243 hours and the Sim sources about 7,494 hours; Appendix A.2 reports per-source statistics and counting rules. Task instructions and action descriptions span a broad range of manipulation scenarios (Table 1 and Figure 2(e–f)). For AgiBot World and Galaxea, whose existing temporal boundaries are already accurate, we reuse the existing annotations and standardize action granularity and skill semantics. Actions that map directly onto the unified

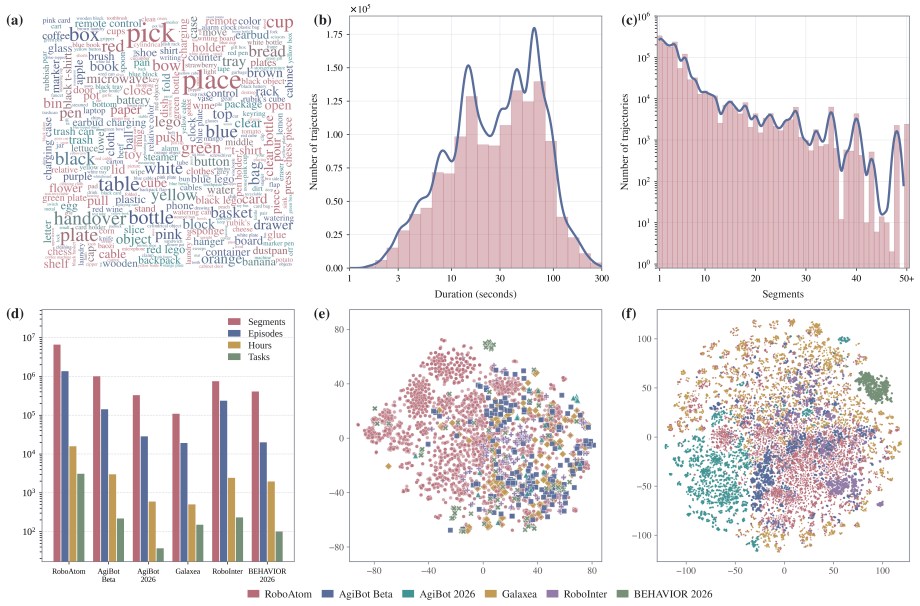

Figure 2: RoboAtom-Data scale and diversity: vocabulary, trajectory duration, segments per trajectory, dataset scale, and language distributions.

Table 1: Robot datasets with subtask annotations. † denotes statistics recomputed by us; source versions and counting rules are in Appendix A.2.

Dataset	Domain	Hours	Trajectories	Segments	Skills	Tasks	Embod.
AgiBot World Beta	Real	2,976.4	141,563†	1,001,552	77†	217	1
AgiBot World 2026	Real	591.6†	28,331†	327,355†	43†	37†	1†
Galaxea Open-World	Real	500.0	19,183†	108,775†	58	150	1
RoboInter-Data	Real	2,417.1†	235,880	752,388	15	232	6
BEHAVIOR-1K 2026	Sim	1,952.9	20,000	406,341	35	100	1
RoboAtom-Data	**Mixed**	**15,736.6**†	**1,363,106**†	**6,606,906**†	**28**	**3,088**†	**30**†

vocabulary are converted by rule; the rest, including composite actions such as PickAndPlace, are split with human priors and state features. The mapping rules are given in Appendix A.3.

3.2 Human-in-the-Loop Seed Annotation

Seed annotations provide the references for boundary extension, and their action granularity, semantics, and temporal boundaries must follow the same specification. We first pre-annotate the trajectories to be labeled with Gemini 3.6 Flash, which produces ordered candidate segments and initial temporal boundaries. In our comparison, contact-sheet input yields slightly better results at much lower cost; therefore, subsequent annotation and evaluation use a frozen 3×4 contact-sheet protocol. Annotators then correct the action semantics, granularity, and segment order against the original video and relocate inaccurate temporal boundaries.

After manual correction, Qwen3-VL-235B-A22B-Instruct (Bai et al., 2025) checks for problems such as spelling errors and inconsistencies between the action description and the skill, and outputs flags for questionable items together with revision suggestions. Annotators accept or reject each suggestion against the original video. Annotations that pass the quality check serve as references for the same-task extension. Complete instructions, annotation interface, and cleaning rules are given in Appendix A.

RoboAtom-Bench. From the human-annotated data we build the expert-reviewed benchmark RoboAtom-Bench, which evaluates atomic-action annotation ability under different sequence lengths and view conditions, using the 15 primary skills as its closed vocabulary. We screen 800

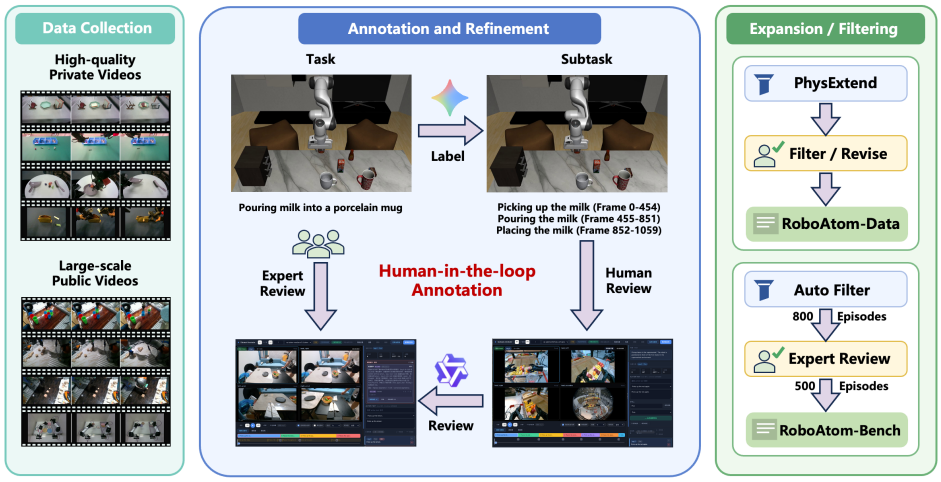

Figure 3: RoboAtom construction pipeline. Model-assisted annotation and human review produce seed annotations; PhysExtend and quality control scale them into RoboAtom-Data. Expert review further establishes RoboAtom-Bench.

candidate trajectories by data source, view information, and number of segments; embodied-data experts review their task semantics and temporal boundaries, after which 500 are retained and supplemented with view annotations. The subset division and detailed statistics are given in Appendix A.5.

3.3 PhysExtend: Skill-Aware Same-Task Extension

Repeated executions of a task often share an ordered skill sequence, allowing boundaries from a few reference trajectories to be propagated. PhysExtend localizes boundaries in the target trajectory; missing, extra, or reordered segments are handled as structural mismatches in Section 3.4.

For a target trajectory of length T with a known ordered skill sequence $y_{1:M}$, we predict internal boundaries that satisfy the following constraint:

$$0 = b_0 < b_1 < \cdots < b_{M-1} < b_M = T.$$

The problem is modeled as global sequence partitioning with order constraints:

$$\hat{\mathbf{b}} = \arg\min_{\mathbf{b}} \sum_{m=1}^{M} [C_m(b_{m-1}, b_m) + \lambda_d D_m(b_{m-1}, b_m)] + \lambda_q \sum_{m=1}^{M-1} Q_m(b_m).$$

Here, C_m, D_m, and Q_m are penalty terms for deviation from the reference position and motion progress, for segment duration, and for skill-dependent gripper events, respectively. PhysExtend selects ordered boundaries jointly by dynamic programming and then applies physical-event correction and monotonic projection. Its physical evidence is derived from state-action signals, including motion intensity, pauses, direction changes, gripper state, and holding-state consistency. For multiple reference trajectories, each reference path is decoded independently and corresponding boundaries are fused before monotonic projection, improving stability while preserving the shared skill sequence (Figure 4). Cortex (Peng et al., 2026) instead fits reference feature distributions and can miss skill transitions when speed, pauses, or retries change (Figure C.1). Implementation details are given in Appendix C.5.

The extension evaluation contains 100 tasks, each with 5 manually annotated and reviewed trajectories, and the ordered skill sequence is identical within each task. Given k of them as references, the remaining $5 - k$ trajectories are segmented. Skill F1@0.7 requires that the skill labels agree and that the temporal intersection over union is at least 0.7.

PhysExtend consistently outperforms Cortex across all reference counts (Figure 4(d)). With a single reference trajectory, it achieves 0.710 Skill F1@0.7 versus 0.485 for Cortex, a 46.3% relative

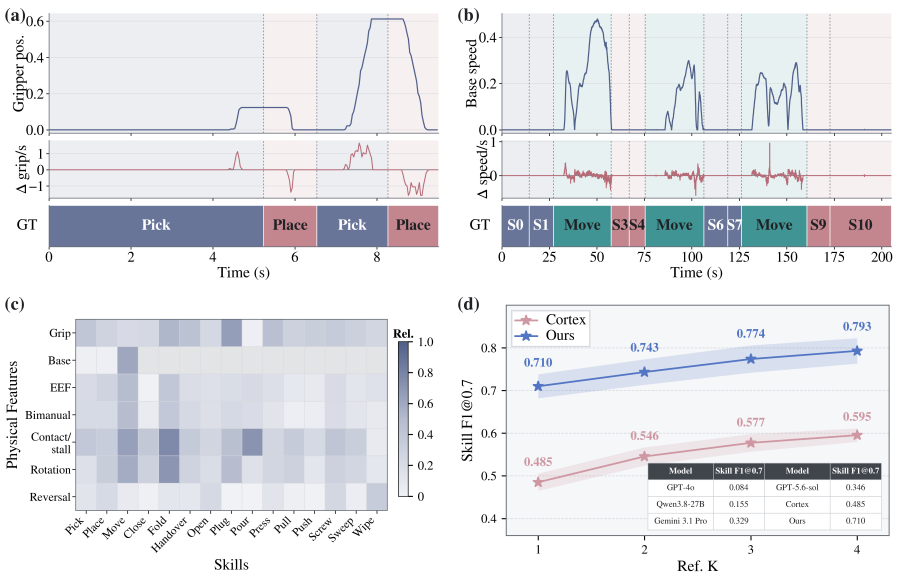

Figure 4: Skill-aware boundary evidence and extension performance: motion signals, skill-feature associations, and PhysExtend versus Cortex across reference counts.

Table 2: PhysExtend annotation quality and cost. Cost is in USD per video hour; < 0.001 denotes compute cost below display precision.

Model	Source	Serving	mIoU ↑	F1@0.7 ↑	Skill F1@0.7 ↑	Cost ↓
GPT-4o	Closed	API	0.482	0.207	0.111	0.888
Gemini 3.1 Pro	Closed	API	0.611	0.449	0.397	2.877
GPT-5.6-sol	Closed	API	0.589	0.378	0.350	3.510
Qwen3.8-27B	Open	Local	0.461	0.254	0.191	0.051
Cortex	Open	Local	0.665	0.498	0.485	< 0.001
PhysExtend	**Open**	**Local**	**0.768**	**0.716**	**0.710**	< 0.001

improvement. Increasing the reference count to four further raises its score to 0.793, demonstrating the benefit of multi-reference fusion for skill-boundary localization. Both methods use the same reference annotations; Cortex uses state-action and visual features, whereas PhysExtend uses only state-action features. Table 2 further compares with general video-language models: the strongest, Gemini 3.1 Pro, reaches 0.397 Skill F1@0.7 at 2.877 USD per video hour. PhysExtend does not read video frames and processes 15,952 trajectories per hour on a single CPU, with a compute cost below 0.001 USD per video hour; the cost accounting is given in Appendix C.5.

3.4 Quality Control

Quality control addresses three failure modes before and after extension: task structures unsuitable for extension, action descriptions that conflict with the target trajectory, and systematic errors in reference labels. Tasks whose trajectories do not share the same skill sequence are excluded from extension. When the skill sequence is shared but the action description conflicts with the target images, we revise the description under the shared skill sequence and send low-confidence cases to human review; per-source spot checks identified 14,793 trajectories. We audit skill–description pairs with a low-cost text-only pass, freeze 341 confirmed suspicious combinations as rules, and send matching trajectories to review, yielding 60,301 trajectories. Targeted visual correction then uses same-task references and field constraints; low-confidence cases are corrected by humans, involving 12,769 trajectories. The original temporal segmentation is preserved unless visual evidence shows a missing segment, extra segment, or severe boundary offset; the full protocol is given in Appendix A.6.

4 LEARNING WITH ATOMIC-ACTION SUPERVISION

The construction and quality-control pipeline provides time-aligned atomic-action annotations across heterogeneous robot trajectories. To study the value for robot learning, we use these annotations as prediction targets for video annotation and as intermediate instructions for robot control.

4.1 ROBOATOM-ANNO: AUTOMATIC ATOMIC-ACTION ANNOTATION

RoboAtom-Anno maps a video to ordered segments containing skills, action descriptions, and temporal boundaries. Temporal boundaries often correspond to brief physical events, such as gripper closure, contact, or release, making them difficult to localize under sparse video sampling. Starting from Qwen3.5-9B (Qwen Team, 2026), we progressively adapt the model to robot scenes, teach annotation of complete videos, and optimize annotation quality through three stages, each initialized from the preceding model.

Robot-domain mid-training. We first adapt the model to the states and relations needed to interpret robot behavior. Supervision includes gripper-state question answering, object grounding, spatial reasoning, and action planning from images or frame sequences. These tasks provide robot-domain knowledge before training on complete video segmentations, yielding the initialization for the next stage. Data sources and task composition are detailed in Appendix B.2.

Supervised fine-tuning. The domain-adapted model then learns to generate reference segment sequences from complete videos under RoboAtom's annotation protocol. Each target segment associates a skill and an action description with start and end frames. The full sequence jointly specifies action semantics, temporal boundaries, and ordering, supervising both skill recognition and boundary localization across the video. Training-subset construction and fine-tuning settings are detailed in Appendices A.4 and B.1.

Difficulty-aware GRPO. After learning from reference sequences, we use GRPO (Shao et al., 2024) to optimize the quality of generated annotations. Training videos are grouped by offline prediction quality on the training supervision pool, emphasizing medium and hard cases while retaining easy examples. For each video, multiple candidate annotations are sampled and scored against the reference. A weighted reward combines ordered temporal matching, continuous overlap feedback on boundary placement, skill-sequence edit similarity, and output validity. The temporal terms emphasize segmentation quality, while the sequence term accounts for missing, additional, or reordered skills. Relative rewards within the candidate group guide model updates. Reward definitions and training settings are in Appendix B.3. Figure 1(b) compares representative model configurations.

4.2 DUAL-SYSTEM POLICY LEARNING AND EXECUTION

Inspired by Hi Robot (Shi et al., 2025), we use atomic-action supervision in a dual-system architecture (Figure 5). During training, each observation–action pair receives the skill label and action description of its covering segment. Together with the task instruction, these condition the System 1 policy, which we obtain by fine-tuning $\pi_{0.5}$ (Physical Intelligence et al., 2025).

At deployment, System 2 supplies the intermediate instructions required by the trained policy. A task-plan prior from the task knowledge base provides candidate skills and action descriptions, which the planner combines with visual observations and execution history to select the current instruction. System 1 executes the selected instruction through action chunks. The resulting observations guide System 2 in retaining the current instruction or switching to another atomic action. Implementation details are in Appendix B.4; Section 5.2 evaluates the combined supervision and execution framework.

5 EXPERIMENTS

We evaluate atomic-action supervision via automatic video annotation and robot control. The annotation experiments measure skill recognition, temporal localization, and inference cost; the control one compare task-level execution with skill-conditioned post-training and hierarchical execution.

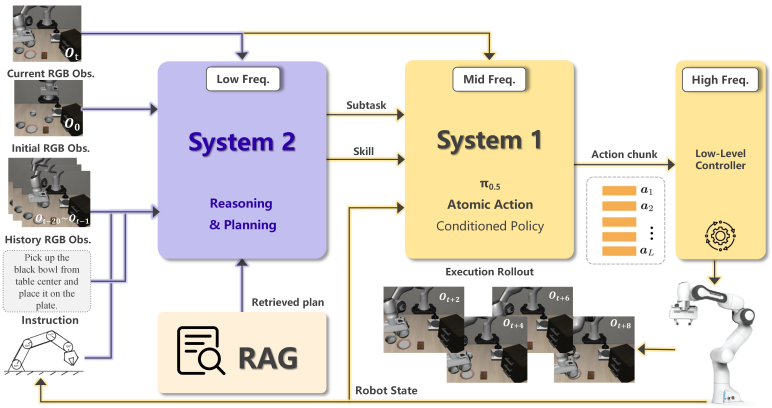

Figure 5: Dual-system execution. System 2 uses a task-plan prior, visual observations, and execution history to select an atomic-action instruction. System 1 conditions $\pi_{0.5}$ on that instruction to produce robot actions; subsequent observations inform the next planning decision.

Table 3: Annotation performance and cost on benchmarks. Cost: USD/h. WGO-Bench uses an open vocabulary, so Skill F1@0.7 is omitted. Protocols are in Append. B.5 and D.1.

Model	Source	Serving	RoboAtom-Bench				WGO-Bench Robot		
			mIoU ↑	F1@0.7 ↑	Skill F1@0.7 ↑	Cost ↓	mIoU ↑	F1@0.7 ↑	Cost ↓
Qwen3.5-9B	Open	Local	0.096	0.062	0.035	0.008	0.128	0.017	0.007
Qwen3.8-27B	Open	Local	0.384	0.201	0.109	0.103	0.513	0.243	0.096
RoboAtom-Anno	**Open**	**Local**	**0.542**	**0.378**	**0.284**	**0.006**	**0.622**	**0.488**	**0.004**
GPT-4o	Closed	API	0.315	0.139	0.072	0.971	0.461	0.146	0.800
Gemini 3.1 Pro	Closed	API	0.425	0.244	0.161	4.516	0.521	0.296	4.389
GPT-5.6-sol	Closed	API	0.440	0.280	0.214	4.330	0.419	0.313	3.375

5.1 Automatic Annotation

Evaluation setup. RoboAtom-Bench evaluates annotation models on 500 expert-reviewed trajectories whose tasks are excluded from the RoboAtom-Anno training subset. We report mIoU, F1@0.7, and Skill F1@0.7, which jointly assess temporal overlap and skill recognition. WGO-Bench provides an external evaluation with an open vocabulary, for which we report mIoU and F1@0.7. Evaluation protocols and cost accounting are given in Appendices B.5 and D.1.

Annotation quality and cost. On RoboAtom-Bench, RoboAtom-Anno achieves Skill F1@0.7 of 0.284, F1@0.7 of 0.378, and mIoU of 0.542 (Table 3). The best evaluated closed-source scores are 0.214, 0.280, and 0.440, respectively. Its inference cost is 0.006 USD per video hour, about 700 times lower than GPT-5.6-sol (4.330) under the reported costing protocol. It also outperforms the larger, locally served Qwen3.8-27B, which obtains Skill F1@0.7 of 0.109 at a cost of 0.103. These results demonstrate the annotation quality and cost efficiency of the trained model.

Training configurations. Figure 1(b) compares representative configurations. The base Qwen3.5-9B model scores 0.035 in Skill F1@0.7, the noisy open-data configuration reaches 0.066, and the cleaned open-data control reaches 0.171. The latter uses vocabulary alignment and ambiguity filtering described in Appendix A.4. With RoboAtom supervision, language-only LoRA obtains 0.217, comparable to GPT-5.6-sol (0.214). These configurations differ in initialization and update settings as well as supervision, so the full curve should not be interpreted as a controlled cumulative ablation.

The controlled visual-update comparison in Appendix C.1 reaches 0.227 when both ViT-LoRA and the visual merger are updated. With this update scope, the configuration initialized by robot-domain mid-training reaches 0.252, and subsequent GRPO reaches 0.284 (Appendix C.2). These results complement the three-stage method in Section 4.1; the visual-update comparison concerns the SFT configuration rather than an additional training stage.

External evaluation and sequence difficulty. On WGO-Bench Robot, RoboAtom-Anno improves F1@0.7 over the strongest evaluated baseline by 0.175 (0.488 vs 0.313), compared with

Table 4: Downstream success rates (%, higher is better) on LIBERO, LIBERO-Plus, and RoboTwin. Avg. is the Clean/Random mean; augmentation settings are given in Appendices C.6–C.7.

Method	LIBERO	LIBERO-Plus					RoboTwin		
	Long	Spatial	Object	Goal	Long	Overall	Clean	Random	Avg.
$\pi_{0.5}$	95.00	**93.01**	85.23	79.93	78.56	84.05	80.70	48.04	64.37
Our method	**98.20**	92.46	**91.86**	**86.34**	**82.29**	**88.18**	**82.02**	**50.32**	**66.17**

0.098 on RoboAtom-Bench (0.378 vs 0.280). The benchmarks differ in sequence complexity: their median segment counts are 2 and 5, respectively, and RoboAtom-Bench includes up to 62 segments. Within RoboAtom-Bench, Skill F1@0.7 decreases from 0.703 on Short to 0.421 on Medium and 0.151 on Long, showing the difficulty of jointly predicting skills and boundaries in long sequences. On WGO-Bench Ego, RoboAtom-Anno reaches F1@0.7 of 0.173 versus 0.141 for Gemini 3.1 Pro, suggesting partial transfer to human manipulation videos (Appendix C.3).

5.2 Robot Control

Evaluation setup. We evaluate long-horizon control on LIBERO-Long (Liu et al., 2023), robustness to perturbations on LIBERO-Plus (Fei et al., 2025), and bimanual manipulation under Clean and Random conditions on RoboTwin (Chen et al., 2025d). The task-level $\pi_{0.5}$ baseline uses complete task instructions during training and end-to-end execution. Our method uses atomic-action instructions during post-training and System 2 scheduling at inference, as described in Section 4.2. Within each benchmark, the two settings share the post-training budget, observation space, and action space. The comparison therefore measures the combined effect of skill-conditioned training and hierarchical execution, rather than isolating the low-level policy (Appendix B.4).

Image augmentation is disabled for both settings. For the baseline, this gives 95.00% on LIBERO-Long and 80.70% on RoboTwin Clean, compared with 92.40% and 70.70% with augmentation. Full augmentation controls are provided in Appendices C.6 and C.7.

Success rates and perturbation analysis. Table 4 shows improvements from 95.00% to 98.20% on LIBERO-Long, from 84.05% to 88.18% on LIBERO-Plus Overall, and from 48.04% to 50.32% on RoboTwin Random. RoboTwin Clean reaches 82.02%, giving a Clean/Random average of 66.17%, compared with 64.37% for the baseline. Figure 1(c) summarizes the LIBERO-Long, LIBERO-Plus Overall, and RoboTwin average results.

On LIBERO-Plus, the gains are concentrated in Object (+6.63), Goal (+6.41), and Long (+3.73), while Spatial changes by -0.55. The benefits thus vary across perturbation types. Overall, the results support the practical value of atomic-action instructions combined with hierarchical execution on the evaluated tasks, using the supervision format that RoboAtom-Data provides at scale.

6 Limitations

Our validation has two limitations. First, atomic-action supervision remains untested in large-scale policy pretraining. Second, real-robot data collection is costly to scale, while the much larger pool of egocentric human video remains unexplored. Its brief, fragmented actions, sometimes lasting less than a second, are difficult to localize at the current sampling rate. Robot supervision offers a starting point, but extending it to human video may require coarser action granularity.

7 Conclusion

We presented RoboAtom, which addresses missing and inconsistent subtask annotations across heterogeneous robot datasets. RoboAtom-Data unifies action semantics and temporal boundaries through a shared annotation protocol, human annotation, same-task propagation, and multi-stage quality control. RoboAtom-Bench provides standardized evaluation of action recognition and temporal localization. Experiments demonstrate the resulting data's value in two applications: RoboAtom-Anno achieves higher annotation quality at lower inference cost than the evaluated open- and closed-source models on both internal and external benchmarks, while atomic-action supervision paired with hierarchical execution improves robot task success over task-level baselines.

REFERENCES

AgiBot World Contributors, Qingwen Bu, Jisong Cai, et al. AgiBot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. *arXiv preprint arXiv:2503.06669*, 2025. URL https://arxiv.org/abs/2503.06669.

Arthur Allshire, Himanshu Gaurav Singh, Ritvik Singh, et al. Scalable behavior cloning with open data, training, and evaluation. *arXiv preprint arXiv:2606.27375*, 2026.

Shuai Bai et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Suneel Belkhale, Tianli Ding, Ted Xiao, Pierre Sermanet, Quon Vuong, Jonathan Tompson, Yevgen Chebotar, Debidatta Dwibedi, and Dorsa Sadigh. RT-H: Action hierarchies using language. *arXiv preprint arXiv:2403.01823*, 2024. URL https://arxiv.org/abs/2403.01823.

Remi Cadene et al. Lerobot: An open-source library for end-to-end robot learning. In *International Conference on Learning Representations*, 2026.

Junting Chen, Haotian Liang, et al. Owmm-agent: Open world mobile manipulation with multi-modal agentic data synthesis. *arXiv preprint arXiv:2506.04217*, 2025a.

Kaiyuan Chen, Shuangyu Xie, Zehan Ma, Pannag R Sanketi, and Ken Goldberg. Robo2vlm: Visual question answering from large-scale in-the-wild robot manipulation datasets. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, 2025b.

Qianzhong Chen, Justin Yu, Mac Schwager, Pieter Abbeel, Yide Shentu, and Philipp Wu. SARM: Stage-Aware reward modeling for long horizon robot manipulation. *arXiv preprint arXiv:2509.25358*, 2025c. URL https://arxiv.org/abs/2509.25358.

Tianxing Chen, Zanxin Chen, Baijun Chen, et al. RoboTwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025d. URL https://arxiv.org/abs/2506.18088.

Tianxing Chen, Yue Chen, et al. Robodojo: A unified sim-and-real benchmark for comprehensive evaluation of generalist robot manipulation policies. *arXiv preprint arXiv:2607.04434*, 2026.

Yi Chen, Yuying Ge, Yixiao Ge, Mingyu Ding, Bohao Li, Rui Wang, Ruifeng Xu, Ying Shan, and Xihui Liu. EgoPlan-Bench: Benchmarking multimodal large language models for Human-Level planning. *arXiv preprint arXiv:2312.06722*, 2023. URL https://arxiv.org/abs/2312.06722.

Matt Deitke et al. Molmo and pixmo: Open weights and open data for state-of-the-art multimodal models. *arXiv preprint arXiv:2409.17146*, 2024.

Zhiwen Fan et al. Vlm-3r: Vision-language models augmented with instruction-aligned 3d reconstruction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

Hao-Shu Fang, Hongjie Fang, Zhenyu Tang, Jirong Liu, Chenxi Wang, Junbo Wang, Haoyi Zhu, and Cewu Lu. RH20T: A comprehensive robotic dataset for learning diverse skills in One-Shot. *arXiv preprint arXiv:2307.00595*, 2023. URL https://arxiv.org/abs/2307.00595.

Yazan Abu Farha and Juergen Gall. MS-TCN: Multi-Stage temporal convolutional network for action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. URL https://arxiv.org/abs/1903.01945.

Senyu Fei, Siyin Wang, Junhao Shi, et al. LIBERO-Plus: In-depth robustness analysis of Vision-Language-Action models. *arXiv preprint arXiv:2510.13626*, 2025. URL https://arxiv.org/abs/2510.13626.

Ning Gao, Jinliang Zheng, et al. Ebench: Elemental diagnosis of generalist mobile manipulation policies. *arXiv preprint arXiv:2606.18239*, 2026.

Songhao Han, Boxiang Qiu, Yue Liao, Siyuan Huang, Chen Gao, Shuicheng Yan, and Si Liu. RoboCerebra: A large-scale benchmark for long-horizon robotic manipulation evaluation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. URL https://arxiv.org/abs/2506.06677.

Chengkai Hou, Kun Wu, Jiaming Liu, Zhengping Che, et al. Robomind 2.0: A multimodal, bimanual mobile manipulation dataset for generalizable embodied intelligence. *arXiv preprint arXiv:2512.24653*, 2025.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Bin Huang, Xin Wang, Hong Chen, Zihan Song, and Wenwu Zhu. VTimeLLM: Empower LLM to grasp video moments. *arXiv preprint arXiv:2311.18445*, 2023. URL https://arxiv.org/abs/2311.18445.

Yuheng Ji, Huajie Tan, Jiayu Shi, et al. RoboBrain: A unified brain model for robotic manipulation from abstract to concrete. *arXiv preprint arXiv:2502.21257*, 2025. URL https://arxiv.org/abs/2502.21257.

Tao Jiang, Tianyuan Yuan, Yicheng Liu, Chenhao Lu, Jianning Cui, Xiao Liu, Shuiqi Cheng, Jiyang Gao, Huazhe Xu, and Hang Zhao. Galaxea Open-World dataset and G0 Dual-System VLA model. *arXiv preprint arXiv:2509.00576*, 2025. URL https://arxiv.org/abs/2509.00576.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, et al. DROID: A Large-Scale In-The-Wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024. URL https://arxiv.org/abs/2403.12945.

Changyeon Kim, Minho Heo, Doohyun Lee, Jinwoo Shin, Honglak Lee, Joseph J. Lim, and Kimin Lee. Subtask-Aware visual reward learning from segmented demonstrations. *arXiv preprint arXiv:2502.20630*, 2025. URL https://arxiv.org/abs/2502.20630.

Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, et al. Behavior-1k: A human-centered, embodied ai benchmark with 1,000 everyday activities and realistic simulation. *arXiv preprint arXiv:2403.09227*, 2024a.

Hao Li et al. Robointer: A holistic intermediate representation suite towards robotic manipulation. *arXiv preprint arXiv:2602.09973*, 2026.

Kanxue Li et al. Muep: A multimodal benchmark for embodied planning with foundation models. In *International Joint Conference on Artificial Intelligence*, pp. 129–138, 2024b.

Zhenyi Liao et al. Improved visual-spatial reasoning via r1-zero-like training. *arXiv preprint arXiv:2504.00883*, 2025.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. *arXiv preprint arXiv:2306.03310*, 2023. URL https://arxiv.org/abs/2306.03310.

Gen Luo et al. Visual embodied brain: Let multimodal large language models see, think, and control in spaces. *arXiv preprint arXiv:2506.00123*, 2025.

Macrodata Labs. Segmenting robot video into actionable subtasks (WGO-Bench). https://macrodata.co/blog/annotating-robot-video-subtasks, 2026. Blog post.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. CALVIN: A benchmark for Language-Conditioned policy learning for Long-Horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 2022. URL https://arxiv.org/abs/2112.03227.

Yao Mu et al. Embodiedgpt: Vision-language pre-training via embodied chain of thought. In *Advances in Neural Information Processing Systems*, 2023.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. RoboCasa: Large-Scale simulation of everyday tasks for generalist robots. In *Robotics: Science and Systems (RSS)*, 2024. URL https://arxiv.org/abs/2406.02523.

NVIDIA, Johan Bjorck, et al. GR00T N1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025. URL https://arxiv.org/abs/2503.14734.

Open X-Embodiment Collaboration. Open X-Embodiment: Robotic learning datasets and RT-X models. *arXiv preprint arXiv:2310.08864*, 2023. URL https://arxiv.org/abs/2310.08864.

Kun Ouyang et al. Spacer: Reinforcing mllms in video spatial reasoning. *arXiv preprint arXiv:2504.01805*, 2025.

Jiaqi Peng, Xiqian Yu, Delin Feng, et al. Cortex: A bidirectionally aligned embodied agent framework for long-horizon manipulation. *arXiv preprint arXiv:2607.05377*, 2026. URL https://arxiv.org/abs/2607.05377.

Physical Intelligence. OpenPI. https://github.com/Physical-Intelligence/openpi, 2025. Open-source software repository.

Physical Intelligence, Kevin Black, Noah Brown, et al. $\pi_{0.5}$: a Vision-Language-Action model with Open-World generalization. *arXiv preprint arXiv:2504.16054*, 2025. URL https://arxiv.org/abs/2504.16054.

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

Vignesh Ramanathan et al. Paco: Parts and attributes of common objects. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Shuhuai Ren, Linli Yao, Shicheng Li, Xu Sun, and Lu Hou. TimeChat: A time-sensitive multimodal large language model for long video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. URL https://arxiv.org/abs/2312.02051.

Pierre Sermanet, Tianli Ding, Jeffrey Zhao, et al. RoboVQA: Multimodal Long-Horizon reasoning for robotics. *arXiv preprint arXiv:2311.00899*, 2023. URL https://arxiv.org/abs/2311.00899.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Lucy Xiaoyang Shi, Brian Ichter, Michael Equi, et al. Hi robot: Open-Ended instruction following with hierarchical Vision-Language-Action models. In *International Conference on Machine Learning (ICML)*, 2025. URL https://arxiv.org/abs/2502.19417.

Yang Tian, Yuyin Yang, Yiman Xie, et al. InternData-A1: Pioneering High-Fidelity synthetic data for pre-training generalist policy. *arXiv preprint arXiv:2511.16651*, 2025. URL https://arxiv.org/abs/2511.16651.

Homer Walke, Kevin Black, Abraham Lee, et al. BridgeData V2: A dataset for robot learning at scale. *arXiv preprint arXiv:2308.12952*, 2023. URL https://arxiv.org/abs/2308.12952.

Kun Wu, Chengkai Hou, Jiaming Liu, et al. RoboMIND: Benchmark on multi-embodiment intelligence normative data for robot manipulation. In *Robotics: Science and Systems (RSS)*, 2025a. URL https://arxiv.org/abs/2412.13877.

Shihan Wu et al. Robocoin: An open-sourced bimanual robotic data collection for integrated manipulation. *arXiv preprint arXiv:2511.17441*, 2025b.

Rui Yang et al. Embodiedbench: Comprehensive benchmarking multi-modal large language models for vision-driven embodied agents. In *International Conference on Machine Learning*, 2025.

Fangqiu Yi, Hongyu Wen, and Tingting Jiang. ASFormer: Transformer for action segmentation. In *British Machine Vision Conference (BMVC)*, 2021. URL https://arxiv.org/abs/2110.08568.

Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied Chain-of-Thought reasoning. *arXiv preprint arXiv:2407.08693*, 2024. URL https://arxiv.org/abs/2407.08693.

Jun Zhang, Teng Wang, Yuying Ge, Yixiao Ge, Xinhao Li, Ying Shan, and Limin Wang. Timelens: Rethinking video temporal grounding with multimodal llms. *arXiv preprint arXiv:2512.14698*, 2026. URL https://arxiv.org/abs/2512.14698.

Zichen Zhang, Yunshuang Li, Osbert Bastani, Abhishek Gupta, Dinesh Jayaraman, Yecheng Jason Ma, and Luca Weihs. Universal visual decomposer: Long-Horizon manipulation made easy. *arXiv preprint arXiv:2310.08581*, 2023. URL https://arxiv.org/abs/2310.08581.

Yuze Zhao et al. Swift: A scalable lightweight infrastructure for fine-tuning. *arXiv preprint arXiv:2408.05517*, 2024.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (System Demonstrations)*, 2024.

Enshen Zhou et al. Roborefer: Towards spatial referring with reasoning in vision-language models for robotics. In *Advances in Neural Information Processing Systems*, 2025.

Yuhan Zhu, Changlian Ma, Xiangyu Zeng, Xinhao Li, Zhiqiu Zhang, Songze Li, Jun Zhang, Tianxiang Jiang, Yuandong Yang, Ziang Yan, Zikang Wang, Xinyu Chen, Haoran Chen, Shaowei Zhang, and Limin Wang. Timelens2: Generalist video temporal grounding with multimodal llms. *arXiv preprint arXiv:2607.17423*, 2026. URL https://arxiv.org/abs/2607.17423.

A Data and Annotation Specification

A.1 Human and Model Collaborative Annotation Pipeline

Pre-annotation is produced by Gemini 3.6 Flash, which generates atomic-action segments and initial boundaries from the video and the task description; the prompt used is given below.

```
Analyze the robotic-manipulation video and return atomic action segments.

Input:
- sample_id: ${sample_id}
- total_frames: ${total_frames}
- fps: ${fps}
- task_instruction: ${task_instruction}
- allowed_skills: [Pick, Place, Handover, Move, Open, Close, Push, Pull, Press, Rotate, Screw,
      Plug, Pour, Fill, Scoop, Stir, Dip, Shake, Fold, Tie, Wipe, Sweep, Wash, Brush, Cut,
     Peel, Strike, Scan]

Use the frame IDs shown in the images. Frame ranges are closed and inclusive.

Rules:
1. Cover every frame from 0 through ${last_frame} exactly once. Segments must be ordered and
     contiguous: each start_frame equals the previous end_frame + 1.
2. Split only when the dominant atomic skill changes. Do not split tiny transitions.
3. Absorb preparation, arm repositioning, transitions, and idle frames into the nearest
     meaningful action. Never output Static.
4. Emit Pick when grasp changes from not holding to holding, and Place when a held object is
     released. Keep them separate from subsequent actions; for example: Pick tool -> Sweep/
     Wipe -> Place tool.
5. Move means physical robot-base/chassis translation only. Arm reaching, lifting, lowering,
     alignment, and retraction belong to the neighboring manipulation.
6. In bimanual scenes, label the dominant active manipulation rather than passive holding or
     stabilization. Use Handover only for a direct transfer between hands/arms.
```

```
7. Use Pull/Push for sliding drawers. Use Open/Close for hinged or articulated doors, lids,
     covers, caps, or flaps. Treat one continuous opening or closing motion as one segment.
8. Each segment has exactly one allowed skill. Decompose composite behaviors. If the video has
      one meaningful action, return one segment covering all frames.

Examples:
- Pick: "picks up the sponge" -> Wipe: "wipes the table" -> Place: "places the sponge on the
     tray".
- Pull: "pulls open the drawer"; Push: "pushes the drawer closed".
- Do not create segments such as "approaches the sponge", "moves the arm", or "remains idle".

Return valid JSON only, with no markdown or extra text:
{
  "sample_id": "${sample_id}",
  "nframes": ${total_frames},
  "fps": ${fps},
  "task": "Overall task description in English",
  "segments": [
    {
      "seg_id": 0,
      "start_frame": 0,
      "end_frame": ${last_frame},
      "subtask": "short English action",
      "primitive_skill": "One allowed atomic skill"
    }
  ]
}

primitive_skill must exactly match an allowed skill, and seg_id must be sequential from 0.
```

For human annotation and review, annotators correct the action descriptions, categories, and boundaries after pre-annotation in the interface shown in Figure A.1.



Figure A.1: Atomic-action annotation interface.

After manual correction, Qwen3-VL-235B-A22B-Instruct proposes revisions based on the video and the current annotation; the review interface is shown in Figure A.2, and the review prompt is given below. Idle and Recover in the prompt are intermediate processing labels: at release, Idle keeps its time interval and text but has an empty skill field and is not counted among valid action segments or skill counts, and Recover is assigned to the corresponding category according to the action actually performed. The released set uses the 28-class vocabulary of Appendix A.3.

```
You are a strict subtask-local reviewer for human subtask annotations of a robot manipulation
     video.

Review only annotation quality. Do not grade delivery time, service plan, or aesthetics.
First read the timeline and understand the episode flow, but judge each segment locally.
For a specific segment, the primary evidence is that segment plus its same-object Pick/Place
     pair.
```

```
Do not use later unrelated objects, global bin visibility, or inventory context alone to
    overturn a local segment.

Input:
- sample_id: <sample_id>
- task: <task_text>
- total_frames: <total_frames>
- fps: <fps>
- review_pass: <review_pass>
- available_view_keys: <available_view_keys_json>
- allowed_skills: <allowed_skills_json>

Human annotation segments use inclusive frame ranges:
<segments_json>

Evidence package:
- The first attached image is a timeline overview with segment ranges, skills, action text,
    and boundaries.
- The remaining contact-sheet images contain frames sampled strictly every N frames, where N
    is configured by --frame-stride.
- No segment-internal key frames or boundary-window frames are attached in this mode.
- Use the frame manifest to map each tile to source frame id, segment id, view key, and stride
     role.

Evidence reading guidance:
Stride-image evidence reading rules:
- Treat the dense stride frames as a frame-by-frame record. Do not decide object identity from
     a single static tabletop frame.
- If multiple views are available, compare tiles with the same source frame id across views
    before changing the annotation.
- Spend most of your attention on tiles where the gripper is visible or about to enter/leave
    the frame. Tiles with no gripper are mostly background context and should not dominate
    the decision.
- For white/light paper, tissue, wrappers, bags, or cloth, inspect the object touching or held
     by the gripper very carefully. Small white objects on a dark gripper can be low-contrast
    ; do not miss a white object simply because it is small, partially occluded, or near the
    gripper edge.
- When judging color, prioritize the object in or immediately under the gripper during grasp,
    lift, carry, and release. A white object on the gripper is stronger evidence than a
    similar white object remaining on the table.
- For every Pick/Place pair, first build a manipulated-object trace: before contact -> gripper
     closure/contact -> object lifted or stably held -> carried object -> release/after-
    release.
- Identify object color/material from the object inside the gripper during lift, transport,
    and immediately before release. This gripper-held evidence is stronger than what remains
    on the table.
- To change a color/material/object name, require held-object evidence from the Pick/Place
    pair, preferably after lift and before release. A claim based only on which object
    remains on the table is not enough.
- A table becoming empty is not proof that later Pick/Place segments are hallucinated. Check
    whether the gripper is already holding an object, whether the object is near the bin, or
    whether a failed/drop/retry sequence is visible.
- A similar object still visible on the table is not proof that the annotated object was not
    picked. There may be duplicates, partial occlusion, or a visually similar object; use
    contact and motion continuity.
- Similar-looking objects are separate instances. If two white/light crumpled papers, tissues,
     bags, wrappers, or cloth pieces are visible, track the specific instance by its initial
    position, gripper contact point, carried trajectory, and release location.
- Do not claim a placed object was returned to the table merely because another similar white/
    light object remains on the table. That remaining object may be a different instance.
- When a Pick/Place pair handles one of multiple similar objects, decide success/failure from
    whether the held instance leaves the gripper at the annotated target, not from whether a
    lookalike is still visible elsewhere.
- Do not swap colors between adjacent paper/tissue/plastic objects by inventory reasoning.
    Only swap object names when the grasped/held object across the local Pick/Place pair
    clearly has the other color.
- Small crumpled paper, tissue, bags, and wrappers can look different under gripper occlusion,
     motion blur, or shadows. If the held object's color is not clearly identifiable, keep
    the original annotation.
- Do not delete or merge retry Pick/Place segments only because the table looks empty after an
     attempted placement. If the gripper is holding, dropping, re-grasping, or carrying an
    object, keep the corresponding atomic segments.
- Conversely, do not invent a failed Pick/Place retry from brief hovering, occlusion, dragging
    , near-table motion, or a momentary pause. A failed attempt requires clear evidence that
    the gripper opens/releases the held instance, the instance is no longer held, and a later
     distinct grasp occurs.
- When the contact sheets show conflicting tabletop context and gripper-held evidence, prefer
    the gripper-held evidence. If the held object is too small, occluded, or out of view,
    keep the original annotation instead of an incorrect repair.

Segment evidence frame manifest:
```

```
<segment_manifest_json>

Global episode evidence frame manifest. Use these only as orientation/background, not as
     standalone proof of an object/bin error:
<global_manifest_json>

Judge whether the human annotation is correct along two axes, using subtask-local evidence:
1. Description accuracy: skill, action_text, object identity, target/bin/container, and
     wording.
2. Segmentation accuracy: whether each segment's dominant atomic skill is coherent and
     adjacent boundaries split only when the dominant atomic skill changes.

Use the same strict error categories internally:
- boundary_error: the boundary cuts a core atomic event incorrectly, merges the core Pick and
     Place events, puts the grasp/lift/release event in the wrong segment, or lets one segment
      contain the core event of another subtask.
- wrong_object: action_text names the wrong object, material, color, or manipulated item,
     proven by clear local evidence.
- wrong_action: action_text or skill names the wrong action, or skill is outside/contradicts
     the allowed skill set.
- wrong_bin: the destination bin/container/color/location is wrong, proven by the current
     segment or same-object pair.
- missing_action: an obvious pick/place/action is absent.
- extra_action: a segment describes an action not shown.
- naming_inconsistent: the same physical object is named inconsistently within its Pick/Place
     pair.
- format_error: text has spelling/format problems that need cleanup.
- unclear_visual: visual evidence is insufficient or occluded after careful inspection.

Segmentation rules:
- Boundaries are important, but do not judge them by duration or by a fixed frame tolerance.
     Judge whether the dominant atomic skill changed.
- For Pick, the segment should contain the grasp/lift event where the gripper changes from not
      holding to stably holding the object.
- For Pick, prioritize object continuity at the gripper: compare frames immediately before
     contact, during gripper closure/lift, and after lift. If the gripper stably holds a
     visually matching object after the grasp, treat the Pick as successful unless local
     evidence proves it held a different object.
- Do not mark a Pick failed only because a similar-colored or similar-shaped item remains on
     the table. There may be duplicate or visually similar objects; the error must be based on
      the annotated target object not entering or remaining held by the gripper, not just a
     lookalike still visible elsewhere.
- For Place, the segment should contain the placement/release event where the held object is
     released and becomes stable at the target.
- Idle is a valid atomic skill at the beginning, middle, or end of an episode when no
     deliberate task skill is being executed. A human-annotated Idle segment must not be
     merged into, or relabeled as, an adjacent action merely because it is long, short, at an
     episode edge, or contains waiting/stationary holding.
- For an explicit Idle segment, action_text should be exactly "Idle." and skill should be Idle
     . Mark it wrong only when clear local evidence shows that the core event of another
     allowed skill occurs inside it; do not invent a new boundary from approach, retreat,
     hovering, arm repositioning, occlusion, or waiting alone.
- Recover is valid only when a clear failed attempt, drop, loss of control, or corrective
     retry is the dominant action. Review existing Recover segments normally, but require
     visible failure/correction evidence.
- Do not add or expand Recover for an ordinary pause, hover, occlusion, transport, arm
     repositioning, successful action, or merely long duration. Never create multiple Recover
     segments from ambiguous micro-transitions.
- Pick and Place should not be collapsed into one segment. A Pick segment should not include
     the core release event; a Place segment should not include the core grasp/lift event.
- Only split a Pick segment into failed Pick/Place/re-Pick when local frames clearly show all
     of these: gripper release/opening, the manipulated instance becomes unsupported on the
     table or another surface, the gripper becomes empty, and a later distinct re-grasp occurs
     . Occlusion, hovering, dragging, near-table motion, or a short pause is not enough.
- Only change a Place destination to "table" when the held instance is visibly released on the
      table and remains there after gripper separation. A similar object visible on the table,
      or the gripper passing near the table, is not enough.
- A long segment is not a boundary error by itself. Transport, waiting, arm switching,
     hovering, approach, and cleanup frames are acceptable when the dominant atomic skill
     remains the same and no core event of another subtask occurs.
- Do not infer a boundary error by comparing one segment's duration against other segments. A
     Pick or Place segment may include idle, stationary holding, hovering, or settling frames
     after the core event when no new atomic skill starts before the next boundary.
- For the first and last segment of an episode, be especially tolerant of leading or trailing
     idle frames that simply cover the episode boundary. Do not require the segment to end
     exactly when the visible grasp/release event finishes.
- Mark an overlong tail as a boundary_error only when the tail contains the core event of
     another skill, or when an obvious separate atomic action begins and should be its own
     segment.
- Do not over-segment micro-transitions. Split only when the leading atomic skill changes.
```

```
- The first segment should begin at the episode start, the last segment should end at the episode end, and adjacent segments should be continuous.

Subtask-local description rules:
- Check object names primarily inside the current segment and its same-object Pick/Place pair. Do not compare against later unrelated objects as if they were the same item.
- For wrong_object, color, material, and wrong_bin, require clear local evidence from the current segment or same-object pair. If the evidence is ambiguous, keep the original annotation and use verdict "correct".
- When multiple similar objects exist, identify the manipulated object by gripper contact, being held, and motion continuity. Do not infer the gripper picked the wrong object from global inventory context or from a similar object still visible on the table.
- Treat duplicate or near-duplicate objects as separate physical instances. Use initial position, gripper contact point, held trajectory, and release location to track which instance is being manipulated.
- A similar white/light paper, tissue, bag, wrapper, or cloth still visible on the table after a Pick or Place does not prove that the held instance was not picked or not placed. It may be another instance.
- Do not label a Place segment as "placed back on the table" unless the held instance is visibly released onto the table. Seeing a lookalike object on the table is insufficient.
- Give higher weight to the object visibly held by the gripper during lift/transport/release than to objects still lying on the tabletop. Tabletop inventory is orientation context, not proof of what was picked.
- Before changing a color/material/object name, verify the object while it is held or released in the local Pick/Place pair. Evidence such as "the other object remains on the table" or "the table is empty" is not sufficient by itself.
- For small crumpled paper/tissue/bag/wrapper objects, avoid color swaps based on shadows, blur, or table-inventory reasoning. If the gripper-held color is not clear, keep the original annotation.
- Do not remove a Pick/Place pair because the tabletop is empty if the gripper is holding or transporting an object in that segment. A held object can be small, white, crumpled, or partially occluded.
- Failed attempts and retries are valid separate actions when the video shows a pick, drop/place back to the table, re-pick, or later release. Do not merge them away unless no held-object or release evidence exists in the local frames.
- When such a failed/corrective interval is labeled Recover, keep Recover if the local evidence supports it. Its action_text should describe the failed attempt or corrective recovery, rather than duplicate the later successful action as though both succeeded. Any newly proposed Recover must cite clear release/drop/loss-of-control and corrective-action frame evidence in review_comment.
- Do not invent failed attempts from ambiguous contact, occlusion, or hover frames. A failed attempt repair needs clear release-empty-regrasp evidence for the same instance.
- For claims such as "the gripper only hovers and does not grasp/lift," require clear before/during/after evidence that no object becomes stably held. If the gripper is holding a similar object, do not call the segment a failed attempt unless the visible evidence proves the held object is not the annotated target.
- Bin/container colors and target locations must match the current release/target evidence. Do not infer a wrong bin just because another bin is globally more visible.
- Use the official task instruction and inventory only as helpful context when provided. Inventory is not proof by itself and must not override clear local visual evidence or the annotated subtask flow.
- "Move" means robot base/chassis movement only. Arm approach/reach/hover should not be labeled as Move.

Output contract:
- Your only repair channel is corrected_action_config plus one overall review_comment.
- If the entire human action_config is correct, set verdict to "correct" and output the original action_config unchanged.
- "Unchanged" means the same segment count and the same start_frame, end_frame, action_text, and skill values. Do not paraphrase, polish, rename, or re-segment correct parts.
- If any part of the human action_config is wrong and the repair is clear, set verdict to "incorrect" and output a new full corrected_action_config that represents what you believe the correct complete annotation should be.
- The corrected_action_config must always be the full video config, not only the changed segments.
- Preserve every segment that does not need repair exactly as written; change only the segments or boundaries required by the visual evidence.
- If evidence is insufficient or the repair is not clear, set verdict to "correct" and output the original action_config unchanged.
- corrected_action_config must cover the entire video from frame 0 through frame <total_frames - 1> with continuous inclusive frame ranges.
- Every corrected segment must contain start_frame, end_frame, action_text, and skill.
- Use only allowed_skills for skill.
- review_comment should be a single overall comment explaining why corrected_action_config is identical or why it differs from the original. Mention affected segment ids and evidence frame ids when useful.
- If review_comment claims a wrong_object, failed Pick, failed Place, or returned-to-table action involving duplicate/similar objects, it must explain how the manipulated instance was tracked by contact/hold/release frames. If you cannot explain that instance trace, do not make that repair.
```

```
- Do not output issues, suggested_patch, patch_type, manual_only, or per-issue accept/reject
    objects.
- Set "needs_more_views": false. This run is binary: if evidence is insufficient, keep the
    original action_config and use verdict "correct".
- Binary mode is enabled. Do not output "uncertain". The only allowed verdict values are "
    correct" and "incorrect".

Return valid JSON only in this exact schema:
{
  "verdict": "correct | incorrect",
  "needs_more_views": false,
  "corrected_action_config": [
    {
      "start_frame": 0,
      "end_frame": 20,
      "action_text": "Corrected text.",
      "skill": "Pick"
    }
  ],
  "review_comment": "Brief Chinese explanation of why the corrected config is identical or
    different.",
  "confidence": 0.0
}
```

After the model suggestions are submitted, annotators accept or reject them one by one; if both the human annotation and the model suggestion are incorrect, a reviewer rewrites that annotation.

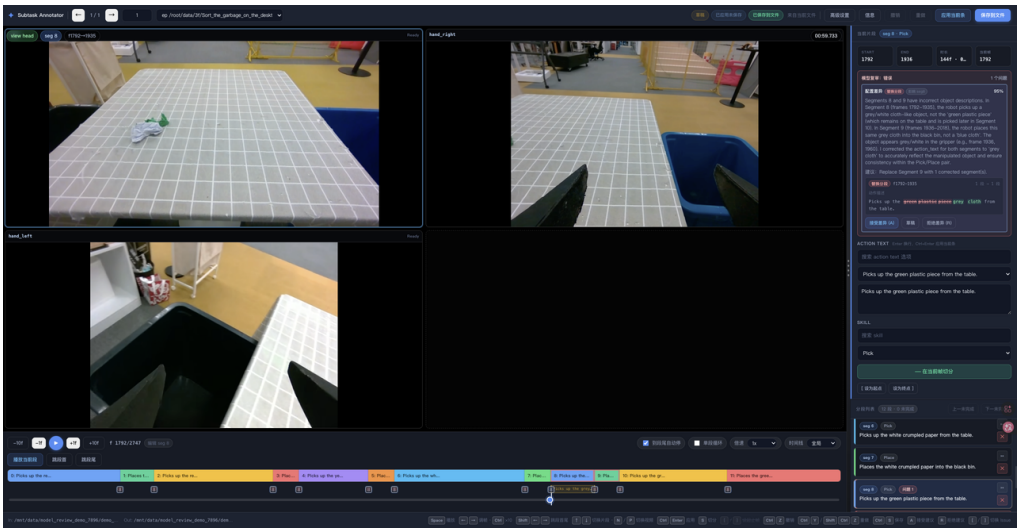

Figure A.2: Model-assisted review interface.

A.2 RoboAtom-Data

The full archived version of RoboAtom-Data contains 1,363,106 trajectories and 6,606,906 segments with non-empty skill labels from 14 sources; per-source statistics are given in Table A.2. Apart from the private data, the sources are ABC-130k (Allshire et al., 2026), EBench (Gao et al., 2026), Galaxea Open-World (Jiang et al., 2025), InternData-A1 (Tian et al., 2025), LIBERO (Liu et al., 2023), RoboCOIN (Wu et al., 2025b), RoboCasa (Nasiriany et al., 2024), RoboDojo (Chen et al., 2026), RoboMIND (Wu et al., 2025a), RoboMIND 2.0 (Hou et al., 2025), RoboTwin 2.0 (Chen et al., 2025d), AgiBot World Beta, and AgiBot World 2026 (AgiBot World Contributors et al., 2025). The evaluation sets are RoboAtom-Bench (500 trajectories) and WGO-Bench (100 trajectories).

For Table 1, hours is the total trajectory (recording) duration, and † marks statistics we recomputed after download. Each source keeps its original annotation unit and skill taxonomy.

In Table 1, AgiBot World 2026 merges three real-robot themes, and Galaxea Open-World uses locally converted annotations; Table A.2 counts only the released set of this paper and its re-segmentation results. RoboInter-Data follows the official recording unit and does not deduplicate

multi-view recordings; Segments for BEHAVIOR-1K 2026 is the number of logical skill annotations.

RoboAtom-Data is counted by unique trajectory identifier and by segments with non-empty skill labels. Tasks are deduplicated by "dataset + normalized task identifier", merging duplicate tasks across embodiments, batches, and train/validation splits; Embod. is deduplicated by the raw non-empty `robot_type` string and denotes the number of metadata types.

Table A.2: Per-source statistics of RoboAtom-Data. †: local or recomputed statistics. Private is one merged source; segments require non-empty skill labels. Skills counts mapped categories per source and is non-additive; the total vocabulary has 28 classes. Embod. counts distinct non-empty `robot_type` strings, deduplicated across sources for the total. Rounded source durations sum to 15,736.7 hours; we retain the archive total of 15,736.6 hours.

Dataset	Domain	Hours	Trajectories	Segments	Skills	Tasks	Embod.
Private	Real	54.7†	3,187†	20,088†	10†	7†	2†
ABC-130k	Real	3,585.6†	130,740†	2,353,804†	17†	198†	1†
EBench	Sim	182.9†	6,600†	42,459†	12†	26†	1†
Galaxea Open-World	Real	465.9†	19,183†	124,104†	32†	169†	1†
InternData-A1	Sim	7,073.8†	621,633†	1,740,880†	13†	1,148†	1†
LIBERO	Sim	3.9†	1,712†	3,910†	8†	40†	1†
RoboCOIN	Real	566.5†	86,023†	387,375†	18†	352†	10†
RoboCasa	Sim	178.9†	68,598†	102,737†	10†	25†	1†
RoboDojo	Sim	20.7†	3,500†	31,063†	12†	35†	1†
RoboMIND	Real	191.4†	64,217†	154,824†	11†	346†	6†
RoboMIND 2.0	Real	1,128.5†	191,342†	847,591†	16†	495†	5†
RoboTwin 2.0	Sim	33.8†	27,500†	70,396†	10†	50†	1†
AgiBot World Beta	Real	2,226.6†	136,908†	715,971†	28†	181†	1†
AgiBot World 2026	Real	23.5†	1,963†	11,704†	11†	16†	1†
RoboAtom-Data	**Mixed**	**15,736.6†**	**1,363,106†**	**6,606,906†**	**28**	**3,088†**	**30†**

A.3 Skill Vocabulary

RoboAtom-Data uses a 28-class atomic-action vocabulary: Pick, Place, Handover, Move, Open, Close, Push, Pull, Press, Rotate, Screw, Plug, Pour, Fill, Scoop, Stir, Dip, Shake, Fold, Tie, Wipe, Sweep, Wash, Brush, Cut, Peel, Strike, Scan. Semantically similar actions are grouped into the same category, for example Unfold into Fold and Unscrew into Screw; the segment descriptions of such classes keep the original verb for distinction, and only the category label is normalized.

From the 28 classes we select 15, namely Close, Fold, Handover, Move, Open, Pick, Place, Plug, Pour, Press, Pull, Push, Screw, Sweep, Wipe, as the closed vocabulary for training and RoboAtom-Bench. The selection balances coverage, cross-source distinguishability, and physical recognizability: these 15 classes cover 99.4% of the skill segments in the released set, whereas each remaining class accounts for less than 0.1% of the segments and is hard to estimate stably under Skill F1@0.7, which requires exact category matching; at the same time, their semantic boundaries are clear across data sources, whereas long-tail classes (such as Stir versus Shake and Cut versus Peel) are often entangled with adjacent actions and have low cross-source annotation consistency. The long-tail classes remain in the released set for research that needs finer granularity.

A.4 Training Data

Section 5.1 performs independent supervised fine-tuning with the open-source cleaned set and with the RoboAtom training subset. The two sets are not merged; their data sources and sampling protocols are reported separately, and the training settings are given in Appendix B.1.

The open-source data control comes from AgiBot World Beta, AgiBot World 2026, and Galaxea Open-World. The three sets are first sampled by task separately and then merged; because their skill and subtask definitions are inconsistent with the evaluation closed set, semantically determinable

aliases are then mapped to the closed set, samples that cannot be uniquely mapped or are ambiguous are deleted, and the result is reviewed by Qwen3-VL-235B-A22B-Instruct. The three sources total 425 tasks with at most 20 trajectories per task; after alignment, 261 tasks and 4,459 trajectories remain. This set is used independently for the open-source cleaned SFT control, is not merged into the RoboAtom training subset, and is not counted in the released set; its Skill F1@0.7 is 0.171 (Section 5.1).

The SFT data of the RoboAtom annotation model come from the training pool organized during supervision construction. We sample part of the tasks to build the training subset and exclude tasks that overlap with RoboAtom-Bench. The final model performs SFT with this list after robot-domain mid-training. The evaluation data for PhysExtend are described separately in Section 3.3 and Appendix C.5.

Reward optimization uses a separately organized list of 5,000 trajectories. Difficulty is divided according to the offline prediction quality on the training supervision pool, and samples are stratified by source, view, and duration, keeping 3,000 medium, 1,250 hard, and 750 easy trajectories. Reward and training settings are given in Appendix B.3.

A.5 Test-Set Distribution

RoboAtom-Bench contains 500 trajectories and 4,018 ground-truth segments, with a median of 5 segments. Short, Medium, and Long are divided by the number of ground-truth segments, with ranges of 1–4, 5–8, and 9–62, respectively, and are used to examine annotation performance under different sequence lengths; Multiview separately evaluates annotation ability with multi-view input. The trajectory counts and detailed statistics of each subset are given in Table A.5, and the model evaluation settings and results are given in Section 5.1.

Table A.5: Test-set distribution of RoboAtom-Bench. Multi-view durations are not multiplied by the number of cameras; trajectory and video counts are reported separately.

Test subset	Trajectories	Share	GT segments	Mean GT segments	Median segments	Segment range	Total frames	Duration (hours)	Videos
Short	200	40.0%	411	2.06	2	1–4	74,214	0.732	200
Medium	100	20.0%	634	6.34	6	5–8	126,762	1.193	100
Long	100	20.0%	2,302	23.02	21	9–62	298,015	2.761	100
Multiview	100	20.0%	671	6.71	5	1–48	108,105	1.119	321
Total	**500**	**100.0%**	**4,018**	8.04	5	1–62	**607,096**	**5.806**	**721**

A.6 Data Quality Inspection

Type 1: inconsistency between action and language sequences. Some data sources define tasks coarsely, and the same task differs in segment count and structure (for example whether the trash is dumped into the bin after sweeping), so reference and target cannot be aligned and the whole task can only be removed from extension. Other sources change the object, attribute, or order under the same task (taking the apple or the orange first, color blocks changing color); the skill structure can still be propagated, but the action text cannot be copied wholesale. After per-source manual spot checks flag suspicious tasks, a vision model, constrained only by the skill-sequence structure, annotation granularity, and language style, reads the task instruction and multi-view images trajectory by trajectory to verify and correct the language and outputs a confidence score; low-confidence cases go to humans. The concentrated problems flagged by the spot checks amount to 27 tasks and 14,793 trajectories.

Type 2: conflict between skill and action text. Per-dataset spot checks show that the extended segmentation largely agrees with the operation stages, so in principle the temporal boundaries are not changed; only obvious missing segments, extra segments, or severe offsets are flagged for human review. Because watching every video would cost too much time and money, we adopt a stratified strategy: a local Qwen3.8-27B audits only the deduplicated literal (skill, action_text) pairs and flags combinations whose predicate conflicts with the skill or falls outside the closed set; after item-by-item manual review, 341 rules are frozen, and a full-corpus scan yields 60,301 trajectories. Visual correction is done by GPT-5.6-sol: it receives same-task references, the closed-set vocabulary, and field constraints, and open-ended rewriting is forbidden (such as rewriting “turn on the stove” as

"rotate the knob", or merging Pick/Place into Move). Conflicting or low-confidence cases keep the source annotation, and 12,769 trajectories in total are handed to human correction.

The prompts actually used are listed below. Curly braces denote runtime fields, and for multiple views `<image>` is inserted per segment. The full-corpus matching after the 341 rules are frozen is a rule scan and does not use a model. The candidate-pair audit is allowed to propose `NEW:<name>`, which is used only to analyze vocabulary coverage and does not directly produce release labels; the release stage still uses only the frozen vocabulary.

(1) Semantic audit of candidate pairs. System Prompt:

```
You audit semantic robot-action labels. Judge the meaning of the supplied
subtask text, not frame boundaries. Be conservative. Use an existing atomic
skill when it is semantically adequate; propose NEW:<name> only when the action
is recurring and materially different. Output JSON only.
```

User Prompt:

```
Candidate ID: {candidate_id}
Current skill: {skill}
Subtask: {action_text}
Corpus occurrences: {occurrences}
Current proposed taxonomy: {candidate_taxonomy}

Return one JSON object:
{
  "verdict": "match|mismatch|ambiguous",
  "proposed_skill": "one taxonomy label or NEW:<concise English label>",
  "needs_taxonomy_expansion": true,
  "reason": "brief Chinese reason",
  "confidence": 0.0
}
The proposed skill must describe the main physical interaction. Do not call
ordinary arm transit Move. Recover is a data-recovery status rather than an
observable action; relabel it to the action stated by the subtask. Idle is valid
only for an actual no-op/waiting interval.
```

(2) Detailed semantic judgment and correction. System Prompt:

```
You review robot-trajectory subtask labels against visual evidence. Be conservative
and evidence-based. Never infer invisible details. Frame boundaries are fixed and
out of scope. Decide whether only the semantic labels match the observed video.
Output JSON only.
```

The common body of the User Prompt is as follows; when trusted same-task references are available, the retention-first constraint given below is inserted, and otherwise that slot is empty:

```
Task: {task}
Ordered semantic labels at the existing fixed video segments:
{ordered_labels_with_seg_id_skill_action_text}

{for each segment: Segment {seg_id} evidence (columns are chronological; rows are camera views
    ): <image>}
{conditional_instructions}

Keep the JSON on one compact line. Encode each issue as [seg_id, issue_type] and each
repair as [seg_id, skill, action_text, confidence]. Omit all reasons and summary.
Do not omit a mismatched segment or shorten action_text. All tuple fields retain their
full meaning and order.

Use exactly this rubric:
- usable: labels are essentially fully correct; no semantic repair is needed.
- uncertain: the trajectory is basically usable but one or more labels need a small,
  actionable repair, such as putting descriptions into the actual observed order,
  correcting object/color/container names, or correcting an action name.
- unusable: the task and observed environment/action are substantially incompatible,
  crucial evidence is absent, or semantic-only edits cannot produce reliable labels.

For order errors, keep every segment and frame boundary fixed, but put the description
of the action actually observed at each seg_id. Never propose video edits or boundary
edits. For uncertain cases, provide only changed segments in repairs. For unusable
cases, repairs must be empty. For usable cases, issues and repairs must both be empty.

Return one JSON object:
```

```
{"verdict":"usable|uncertain|unusable","issues":[[0,"order_mismatch|wrong_object|wrong_color|
    wrong_action|wrong_destination|missing_or_extra|environment_task_mismatch|
    insufficient_visual_evidence"]],"repairs":[[0,"Pick","Picks up ...",0.0]]}
```

For samples with trusted same-task references, the following retention-first constraint is inserted:

```
This run uses a retention-first, task-aligned policy:
- Preserve a reasonable current label. A task goal and its visible motor implementation
  can both be correct; do not rewrite one merely because the other is also possible.
- Never weaken Pick/Place/Open/Close/Insert/Remove into generic Move/Manipulate/Adjust.
- Use only skills present in the supplied trusted task variants. Never propose NEW:*.
- Repair only a concrete evidenced contradiction. If the evidence and trusted variants
  do not determine a unique safe repair, return uncertain with no repairs.
- Trusted same-task variants are constraints and style examples, not proof that this
  trajectory performed an action that is not visible.
Trusted task constraints:
{reference_constraints}
```

A.7 LANGUAGE-SPACE COVERAGE

RoboAtom-Data covers a broad range of task instructions and action descriptions across manipulation scenarios. Figures A.3 and A.4 provide dataset-specific views of the distributions summarized in Figure 2(e–f).

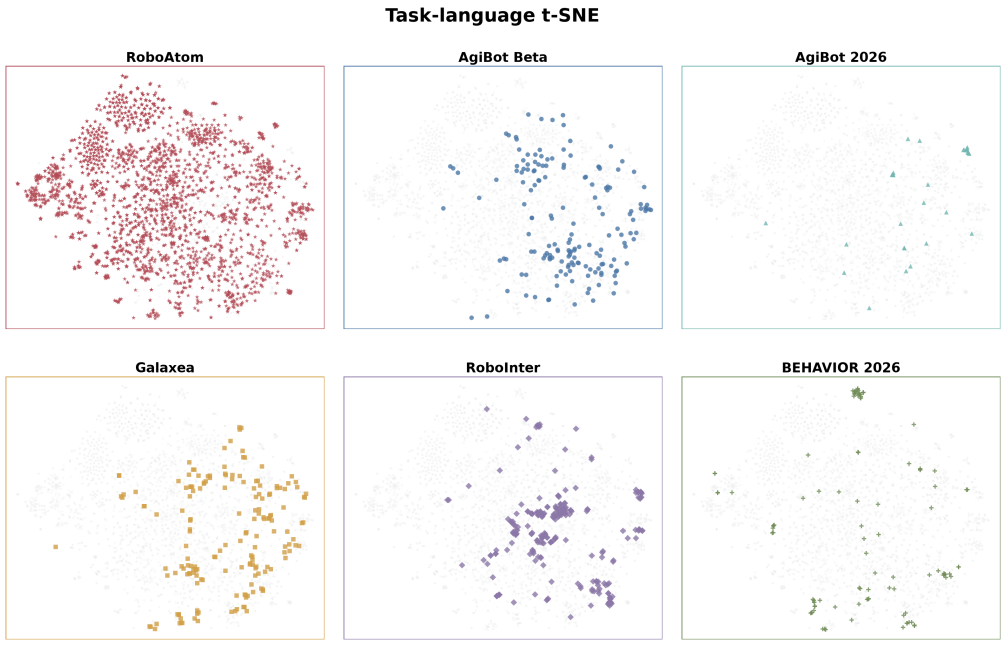

Figure A.3: Task-instruction distributions across datasets. Colored points highlight each dataset within the shared t-SNE projection.

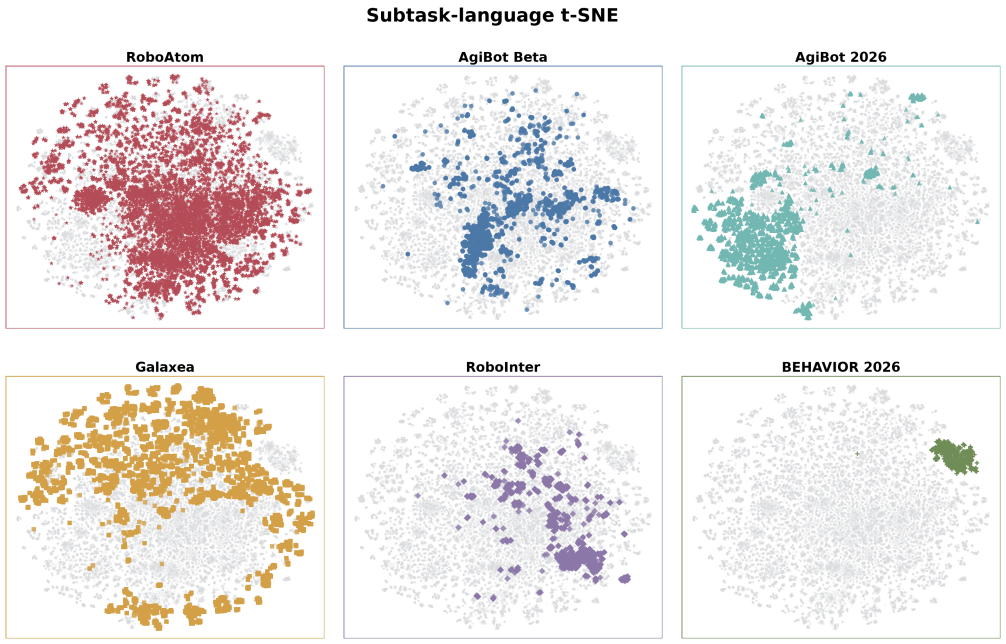

Figure A.4: Action-description distributions across datasets, illustrating the broad atomic-action coverage of RoboAtom-Data.

B TRAINING AND EVALUATION SETTINGS

RoboAtom-Anno follows the three-stage method in Section 4.1: robot-domain mid-training (Appendix B.2), supervised fine-tuning on a task-sampled subset (Appendix B.1), and GRPO (Appendix B.3). The downstream policy in Section 4.2 fine-tunes $\pi_{0.5}$ with OpenPI (Appendix B.4). Benchmark construction is described in Section 3.2, and evaluation settings are given in Section 5.1 and Appendix B.5.

B.1 SUPERVISED FINE-TUNING SETTINGS

Supervised fine-tuning is performed in LLaMA-Factory (Zheng et al., 2024) with Qwen3.5-9B (Qwen Team, 2026) as the base; the construction and task split of the training subset are given in Appendix A.4. The visual update-scope ablation is not preceded by mid-training; the final model is initialized from the robot-domain mid-training weights and then goes through supervised fine-tuning and GRPO optimization on this subset. The LoRA (Hu et al., 2022) rank is 8, applied to the linear layers on both the language and vision sides, and the visual merger is updated as well. The language-side LoRA is trained in all four visual update ablations; the switches for ViT-LoRA and the visual merger module are given in Table C.1.

Videos are sampled at 2 FPS with at most 448 frames per trajectory, using the Qwen3.5 chat template without chain of thought, and segment boundaries are output by frame index. The effective batch size is 64 (8 GPUs $\times$ 2 per GPU $\times$ 4 gradient accumulation steps), the learning rate is 2×10^{-5}, and training runs for 500 optimizer steps.

B.2 ROBOT-DOMAIN MID-TRAINING

Before supervised fine-tuning, the final model undergoes robot-domain mid-training, which is also performed in LLaMA-Factory. The corpus comes from 27 data subsets with 2,841,942 samples after merging; Table B.1 summarizes the six task categories.

Table B.1: Task composition of robot-domain mid-training.

Category	Samples	Training tasks
Locomotion	17,185	Navigation actions, movement direction, and target keypoint prediction
Manipulation	954	Grasp/release actions and manipulation keypoint prediction
Robot QA	1,007,450	Question answering on robot state, scene, and action
Grounding	1,080,004	Localization of 2D points, target boxes, affordance regions, and manipulation trajectories
Planning	326,747	Task decomposition, action sequence, and next-action prediction
Spatial reasoning	409,602	Reasoning about distance, direction, relative position, and 3D relations
Total	**2,841,942**	Total training samples of the six categories

The mid-training data sources and task descriptions are as follows. Locomotion comes from VeBrain (Luo et al., 2025); from the robot's egocentric image, it judges the movement direction, the next action, and the target landing point, and outputs navigation actions and 2D keypoints.

Manipulation: also from VeBrain. From the image, it judges the next grasp or release action and predicts manipulation keypoints. It is not full-trajectory segmentation data.

Robot QA: includes OWMM-VLM (Chen et al., 2025a), Robo2VLM (Chen et al., 2025b), and RoboTwinQA. The tasks cover gripper-state question answering, scene understanding, target search, the next robot action, and spatial and grounding question answering.

Grounding: includes RefSpatial-3D and RefSpatial-Simulator (Zhou et al., 2025), affordance, bounding box, PACO-LVIS (Ramanathan et al., 2023), PixMo points (Deitke et al., 2024), and trajectory, among others. The outputs are mainly 2D points, target boxes, affordance regions, or manipulation trajectories on the image plane; this is spatial grounding and contains no subtask temporal boundaries.

Planning: includes EgoCOT (Mu et al., 2023), EgoPlan (Chen et al., 2023), MuEP (Li et al., 2024b), and EmbodiedBench-ALFRED/Habitat (Yang et al., 2025), among others. From an image or a frame sequence, it generates a task plan, an action sequence, or the next action, mainly training step decomposition and action ordering.

Spatial reasoning: includes SR-91K (Ouyang et al., 2025), VSI-100K (Liao et al., 2025), and VLM-3R (Fan et al., 2026), and trains spatial reasoning about distance, direction, relative position, and 3D scene relations.

Training is initialized from Qwen3.5-9B and runs for one epoch with a learning rate of 2×10^{-5}, a per-GPU batch size of 8, and gradient accumulation of 4; videos are still input at 2 FPS with at most 448 frames; the LoRA rank is 8, and the visual merger module is updated.

B.3 GRPO Reward and Training Settings

GRPO (Shao et al., 2024) is implemented in ms-swift (Zhao et al., 2024) and continues optimization from the model that has completed robot-domain mid-training and supervised fine-tuning. Reward training uses the 5,000 trajectories organized separately in Appendix A.4: they are stratified by the quality of offline rollouts on the training supervision pool, with 3,000 medium, 1,250 hard, and 750 easy trajectories.

The offline prediction results on the training subset are divided into difficulty levels by prediction quality and stratified by data source, number of views, and duration.

$$R = 0.70R_{\text{temporal}} + 0.10R_{\text{soft}} + 0.18R_{\text{edit}} + 0.02R_{\text{format}}.$$

R_{temporal} is the F1@0.5 based on ordered temporal matching and does not require correct skill labels; R_{soft} is the sum of IoU over ordered matches divided by the larger of the numbers of predicted and GT segments, which provides continuous feedback for boundary changes; R_{edit} is the edit similarity of the skill sequence; and R_{format} checks strict parsing and field constraints. If the output cannot be parsed, the first three terms are set to zero, the format term is set to -1, and the total reward is -0.02.

GRPO uses 16 GPUs with a training batch size of 1 per GPU and gradient accumulation of 4; 8 candidate outputs are sampled per prompt, each prompt is updated 2 times, and a group of candidates is generated every 4 steps. The learning rate is 5×10^{-6} with cosine decay; the KL coefficient is 0.02, the clipping range is 0.2, and the maximum generation length is 4,096 tokens. Videos are sampled at 2 FPS with at most 448 input frames. The candidate advantage is computed as the group-centered reward divided by the within-group standard deviation:

$$A_i = \frac{r_i - \bar{r}}{s_r + 10^{-4}}.$$

B.4 Downstream Executor

The downstream policies for LIBERO, LIBERO-Plus, and RoboTwin all fully fine-tune $\pi_{0.5}$ with OpenPI (Physical Intelligence, 2025) using a shared global batch size of 256. LIBERO uses 8 GPUs and 30,000 training steps; RoboTwin uses 32 GPUs (4 nodes $\times$ 8 GPUs) and 60,000 training steps. The language condition of the task-level baseline is the complete task instruction for both training and inference, without the high-level scheduling of System 2 or the task-plan prior; the training language condition of our method is the skill instruction corresponding to each atomic-action segment. At inference, our method receives the atomic-action instructions issued by System 2 according to observations and the execution history.

System 2 uses Qwen3.8-27B as the planner. A task-plan prior from the task knowledge base provides candidate skills and action descriptions; the planner combines this prior with visual observations and execution history to retain the current atomic-action instruction or select another. System 1 executes the selected instruction through robot action chunks.

The difference between the two groups includes both the training language condition and the inference execution mode: the task-level baseline executes end to end from the complete task instruction,

whereas in our method, System 2 issues atomic-action instructions segment by segment and System 1 executes them.

The comparison of the $\pi_{0.5}$ baseline with augmentation on and off is given in Appendices C.6–C.7.

Each RoboTwin evaluation has 5,000 trials: the dual system succeeds 4,101 times under Clean and 2,516 times under Random, 6,617/10,000 in total, for an average of 66.17%; the end-to-end policy succeeds 4,035 times under Clean and 2,402 times under Random, 6,437/10,000 in total, for an average of 64.37%.

B.5 Visual Input and Evaluation Metrics

The main results use the 500 fixed human-annotated test trajectories of Appendix A.5, and the evaluation window is the time interval covered by the standard annotation. mIoU measures temporal overlap; F1@0.7 requires a temporal intersection over union of at least 0.7; Skill F1@0.7 further requires that the normalized skill labels agree. RoboAtom-Bench results are given in Table 3.

For visual input, RoboAtom-Anno and its SFT/GRPO configurations are evaluated at 2 FPS with at most 448 frames and deterministic decoding.

WGO-Bench has 100 trajectories, of which 75 are Robot (50 DROID, 25 Galaxea) and 25 are Ego (HomER). We use a unified evaluator that covers the whole video without removing background intervals and report mIoU and F1@0.7. WGO uses open-vocabulary annotation and has no standard labels corresponding to our unified categories, so Skill F1@0.7 is not reported. Supplementary WGO results are given in Table C.3.

The evaluation closed set is Close, Fold, Handover, Move, Open, Pick, Place, Plug, Pour, Press, Pull, Push, Screw, Sweep, Wipe. RoboAtom-Anno outputs a JSON object in which `segments` is an ordered list of segments, each containing `skill`, `subtask`, `start_frame`, and `end_frame`; start and end frames use zero-based closed-interval coordinates of the original video.

C Supplementary Experimental Results

C.1 Update Scope of the Visual Modules

On RoboAtom supervision sampled by task, we further examine the scope of SFT parameter updates. The LoRA of the Qwen3.5 vision backbone (ViT) and the visual projection module `visual.merger` can be switched on and off independently; the language-side LoRA participates in training in every cell of the 2×2 ablation. Updating both the ViT-LoRA and the merger gives the highest Skill F1@0.7, 0.227. Updating only the merger lowers Skill F1@0.7 from 0.217 to 0.214 but raises mIoU from 0.443 to 0.500; the metrics do not respond consistently. Subsequent experiments adopt the configuration that updates both the ViT-LoRA and the merger; full results are given in Table C.1.

Table C.1: Ablation of the update scope of the visual modules. All configurations update the language-side LoRA for 500 steps on the task-sampled RoboAtom subset (Appendix A.4).

Configuration	ViT-LoRA	visual.merger	mIoU	F1@0.7	Skill F1@0.7
① Language-only	Off	Frozen	0.443	0.287	0.217
② + merger	Off	Updated	**0.500**	0.287	0.214
③ + ViT-LoRA	On	Frozen	0.438	0.288	0.223
④ + ViT-LoRA + merger	On	Updated	0.487	**0.295**	**0.227**

C.2 Analysis of Annotation-Model Training Configurations

Robot-domain mid-training uses 2,841,942 samples from 27 subsets, covering six task categories: robot question answering, spatial grounding, spatial reasoning, planning, locomotion, and manipulation. After one epoch of domain training from Qwen3.5-9B, with the same task-sampled supervision and 500-step SFT configuration, Skill F1@0.7 rises from 0.227 to 0.252; continuing with

GRPO reaches 0.284. This analysis serves to explain the training configuration of the final annotation model. The mid-training data are not counted toward the scale of RoboAtom atomic-action supervision; their composition and hyperparameters are given in Appendices B.2–B.3.

C.3 Supplementary WGO-Bench Results

On WGO-Bench Robot, the F1@0.7 of RoboAtom-Anno is 0.175 higher than that of the strongest baseline (0.488 vs 0.313); on RoboAtom-Bench it is 0.098 higher (0.378 vs 0.280). The two benchmarks differ in trajectory and segment distribution: WGO-Bench Robot has a mean of 3.64 segments and a median of 2, whereas RoboAtom-Bench has a median of 5 segments and a maximum of 62. Per-subset results on RoboAtom-Bench are given in Appendix C.4. On the WGO-Bench ego subset, RoboAtom-Anno reaches mIoU/F1@0.7 of 0.268/0.173, higher than the best baseline results of 0.180/0.141; these subset results are reported in Table C.3.

Table C.3: Automatic annotation performance on each WGO-Bench subset. WGO-Bench uses an open vocabulary, so mIoU and F1@0.7 are reported.

Model	Robot		Ego		Overall	
	mIoU	F1@0.7	mIoU	F1@0.7	mIoU	F1@0.7
GPT-4o	0.461	0.146	0.096	0.053	0.230	0.102
GPT-5.6-sol	0.419	0.313	0.156	0.138	0.253	0.214
Gemini 3.1 Pro	0.521	0.296	0.180	0.141	0.305	0.215
Qwen3.8-27B	0.513	0.243	0.140	0.097	0.277	0.170
Qwen3.5-9B	0.128	0.017	0.013	0.004	0.055	0.010
RoboAtom-Anno	0.622	0.488	0.268	0.173	0.398	0.306

C.4 Annotation Difficulty of the Test Subsets

The detailed subset breakdown is given in Table C.4; Skill F1@0.7 decreases as the number of segments grows, from 0.703 on the Short subset to 0.151 on the Long subset.

Table C.4: Skill F1@0.7 of RoboAtom-Anno on each RoboAtom-Bench test subset

Subset	Trajectories	Mean GT segments	Skill F1@0.7
Short	200	2.06	0.703
Medium	100	6.34	0.421
Long	100	23.02	0.151
Multiview	100	6.71	0.339
Overall	500	8.04	0.284

C.5 PhysExtend Implementation and Feature Evidence

Boundary localization jointly considers relative position, cumulative motion progress, local feature differences, segment duration, and skill-dependent gripper events; the objective is given in Section 3.3. Candidate boundaries are generated from reference durations, cumulative motion alignment, and gripper events, and their positions are determined by joint dynamic programming over ordered candidates before correction with reliable Pick/Place events. Each reference trajectory generates boundaries independently; corresponding boundaries are fused and then monotonically projected so that the segments do not overlap and cover the whole trajectory, preserving the shared action structure while reducing sensitivity to noise in any single reference trajectory. We characterize Pick and Place by gripper-state transitions (Figure 4(a)), with event direction calibrated from the reference trajectories to accommodate different robot encodings. Move is characterized by chassis or end-effector motion; Figure 4(b) illustrates the chassis-motion evidence. Bimanual coordination, contact, pauses, rotation, and direction changes provide additional cues for skill boundaries, and Figure 4(c) summarizes the qualitative associations between the 15 primary skills and these physical features. Boundary and within-segment evidence is compared in Appendix Figure C.2. The

extension evaluation contains 100 tasks with five annotated trajectories per task; we evaluate reference counts from one to four and aggregate results across all reference combinations. Figure C.1 compares the boundary timelines of Cortex and PhysExtend with ground truth.

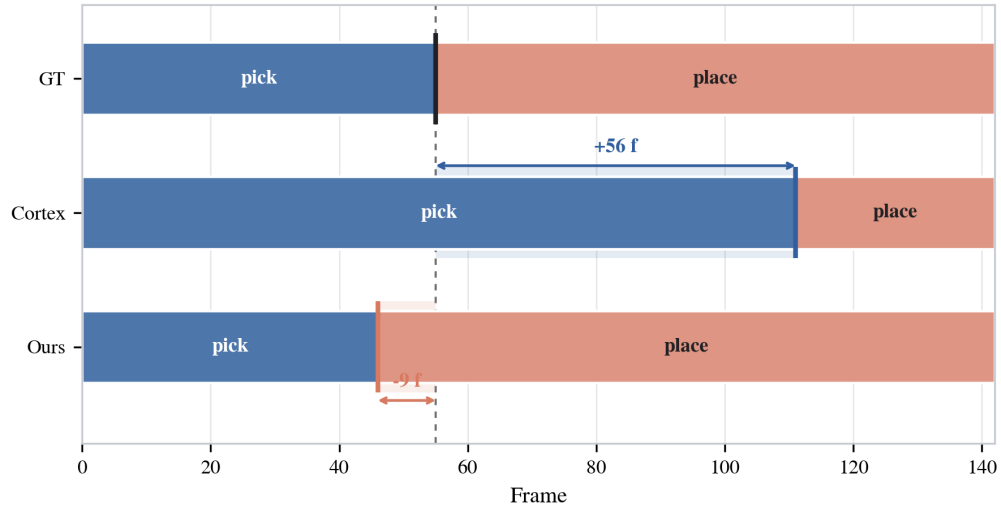

Figure C.1: Boundary localization of Cortex and PhysExtend compared with ground truth (GT).

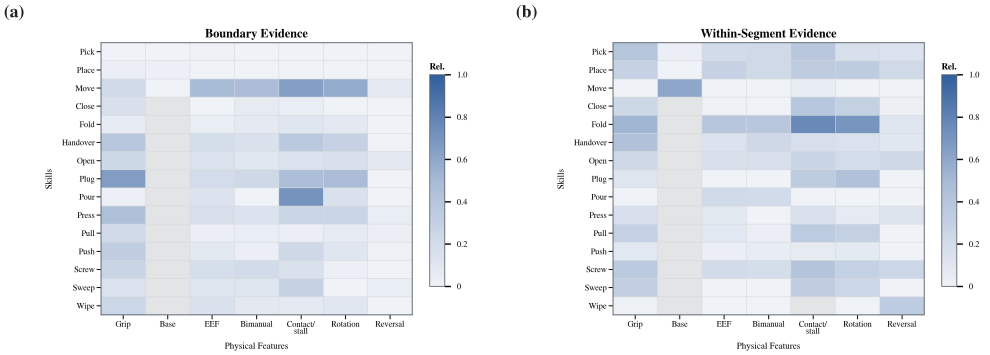

Figure C.2: Response of physical features at action boundaries versus within segments. (a) Relative evidence at the boundaries; (b) evidence of the same features within the segments (at the center).

With a single CPU and 1 reference trajectory, PhysExtend processes 15,952 trajectories per hour, about 17.1 times the throughput of Cortex. The corresponding compute costs for Cortex and PhysExtend are 0.005253 and 0.000307 USD per 1,000 trajectories, respectively, or approximately 0.000481 and 0.000028 USD per source-video hour, respectively.

C.6 Image Augmentation Control for $\pi_{0.5}$

With image augmentation turned off, the LIBERO-Long success rate of $\pi_{0.5}$ is 95.00%, 2.60 percentage points higher than the 92.40% with augmentation on; the standard LIBERO Overall results are 97.125% and 96.85%, respectively. The OpenPI maintainers also explain in issue #442 that the augmentation defaults of π_0 and π_0-FAST at the time came from historical settings and were kept during fine-tuning for consistency with the respective pretraining. Given this background and the measured performance on long-horizon tasks, the main text uniformly compares $\pi_{0.5}$ and our method with image augmentation turned off.

Tables C.6(a) and C.6(b) compare $\pi_{0.5}$ with image augmentation off (w/o aug) and on (w/ aug).

Table C.6(a): Image augmentation control of the $\pi_{0.5}$ baseline on standard LIBERO (%).

Method	Spatial	Object	Goal	Long	Overall
$\pi_{0.5}$ (w/o aug)	97.00	99.00	97.50	95.00	97.125
$\pi_{0.5}$ (w/ aug)	98.80	98.20	98.00	92.40	96.85

Table C.6(b): Image augmentation control of the $\pi_{0.5}$ baseline on LIBERO-Plus (%).

Method	Spatial	Object	Goal	Long	Overall
$\pi_{0.5}$ (w/o aug)	93.01	85.23	79.93	78.56	84.05
$\pi_{0.5}$ (w/ aug)	91.63	89.36	79.08	79.48	84.77

C.7 Image Augmentation Control on RoboTwin

Table C.7 compares the success rates of the $\pi_{0.5}$ baseline on RoboTwin with image augmentation on and off. w/ aug and w/o aug denote image augmentation on and off. Clean and Random denote evaluation in the clean and randomized settings after training on clean data, and Avg. is their arithmetic mean.

Table C.7: Image augmentation control of the $\pi_{0.5}$ baseline on RoboTwin (%).

Method	Clean	Random	Avg.
$\pi_{0.5}$ (w/ aug)	70.70	46.00	58.35
$\pi_{0.5}$ (w/o aug)	80.70	48.04	64.37

D Inference Efficiency and Cost

This appendix computes the inference cost of local models from the GPU rental unit price and the actual occupancy time. The inference unit prices in Table 3 exclude the costs of model training and data construction.

D.1 GPU Cost Timing for the Main-Table Models

The processing cost in Table 3 is reported in USD per video hour. Closed-source models are priced by the actual input, cached-read, and output token usage at undiscounted API list prices, including format repairs and failed requests, divided by the total logical video duration; local models are priced by the amortized cost of the RTX 4090 GPUs actually occupied, computed as follows.

The GPU price is amortized from the rental cost of an eight-GPU machine. With a monthly rent of 2,235 USD and 720 hours per month, the unit price is 0.3880208 USD per GPU hour. Each local 9B model runs on four RTX 4090 GPUs in parallel.

GPU cost = 0.3880208 USD per GPU hour $\times$ 4 GPUs $\div$ measured video throughput (video hours per wall-clock hour). Throughput is computed from the total source-video duration divided by the latest finish time minus the earliest start time across the four GPUs. Timing includes video decoding, frame extraction, visual encoding, and generation, but excludes model loading, storage, data transfer, labor, and device idle time.

The measured end-to-end throughput is 182.73 video hours per wall-clock hour for Qwen3.5-9B and 276.49 for RoboAtom-Anno on RoboAtom-Bench, and 214.69 and 368.46, respectively, on the 75-video WGO-Bench Robot subset.

The resulting costs are 0.00849 and 0.00561 USD per video hour on RoboAtom-Bench, and 0.00723 and 0.00421 USD per video hour on WGO-Bench Robot, for Qwen3.5-9B and RoboAtom-Anno, respectively. The main table rounds these values to three decimal places.