IMPROVING PROACTIVE AI ASSISTANCE WITH HIERARCHICAL PROCEDURAL UNDERSTANDING

Anonymous authors
Paper under double-blind review

ABSTRACT

Proactive procedural assistance aims to continuously monitor user progress on procedural tasks like cooking and provide guidance on the next step when the current step is completed. Existing approaches mainly construct step-level guidance datasets. However, step-level supervision alone provides limited information for tracking task progress and completion in a streaming setting. A step consists of multiple lower-level actions and also belongs to a higher-level phase. In understanding the progress of the current step, lower-level actions provide evidence of what has been completed and what remains, while higher-level phase context can clarify the purpose and expected outcome of the current step. Also, modeling of the phase–step–action hierarchy allows the generation of guidance at different granularities according to the user's needs. In this paper, we introduce *ProactiveCoach-Instruct*, a training dataset with verbal guidance hierarchically organized by phases, steps, and actions, and *ProactiveCoachBench*, a benchmark for evaluating multi-level proactive guidance and guidance-level adaptation. Using ProactiveCoach-Instruct, we propose *ProactiveCoach*, a method that jointly learns proactive guidance across all three levels in a single vision-language model. Our method consistently improves proactive assistance performance compared with step-only training. We further build an adaptive guidance system with a lightweight router, allowing the guidance level to change according to the user's request. An anonymous dataset preview is available at `https://anonymous.4open.science/r/dataset-review-7c3e/`.

1 INTRODUCTION

Building AI assistants like JARVIS that continuously understand their surroundings and proactively help users when needed has long been a human aspiration. For example, a future smart-glasses assistant could recognize what a user is currently doing and provide guidance at an appropriate moment, while a future humanoid assistant robot could monitor task progress and proactively provide physical assistance to the user when help is needed (Wang et al., 2026; Xia et al., 2026; Zheng et al., 2026; Zhao et al., 2026; Azad et al., 2026; Zhang et al., 2026; Yan et al., 2026a; Ma et al., 2026; Yu et al., 2026). Such AI assistants should be able to support users in procedural tasks, such as cooking, assembly, and repair, which require a series of activities to achieve a goal. To realize this capability, an assistant should be able to not only recognize individual activities from a streaming video but also understand how the procedural task is progressing and determine what guidance to provide, when to provide it, and at what level of detail. Recent studies have begun to explore proactive procedural assistance to address this challenge (Zhang et al., 2025b; Kundu et al., 2026; Liu et al., 2026).

In proactive procedural assistance, once the user provides a task goal, the assistant continuously observes the video stream and decides whether to provide new guidance or remain silent. A key challenge is tracking the progress of the current guided event and determining when it is complete, since the next guidance should be provided only after the current event ends. Prior studies mainly construct step-level guidance annotations, covering the steps needed to achieve the task goal and their guidance timing (Zhang et al., 2025b; Kundu et al., 2026). Using these annotations, models are trained to provide the next step-level guidance when the current step is completed and remain silent while it is still in progress. However, step-level supervision alone provides limited information about the progress within the current step, making it difficult to accurately track its progress and determine its completion in a streaming setting.

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

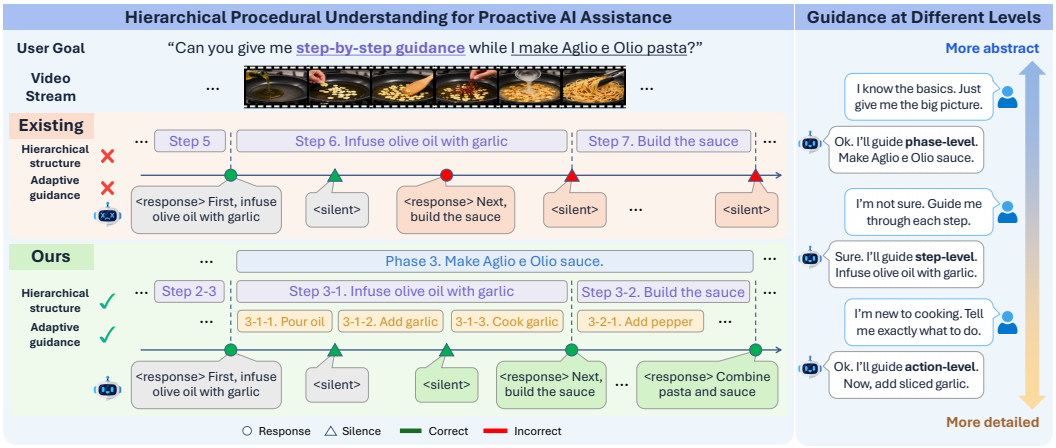


Figure 1: Overview of PROACTIVECOACH. It tracks progress and completion across phase, step, and action levels and adapts the guidance level to the user's request.

We propose to provide hierarchical supervision above and below the step level. At the level below, a procedural step may consist of multiple lower-level actions (Song et al., 2023), such as picking up a screwdriver, aligning its tip to the screw head, and applying multiple turns. In a streaming setting, however, the assistant cannot observe the entire step at once because it has access only to observations up to the current moment (Li et al., 2026a; Chen et al., 2024). This makes it difficult to track the progress of the guided step and determine whether it is complete. Understanding what actions make up the step and how each action is progressing can provide useful evidence for determining whether the current step is complete. At the level above, each step belongs to a higher-level phase that provides broader context about the role and purpose of the step in the overall procedure. Higher-level phase context can help clarify the purpose of the current step and its expected outcome, which helps the model understand how the step contributes to the overall procedure. This hierarchical procedural understanding could provide a more complete view of the ongoing task and a stronger basis for proactive guidance. However, existing datasets and benchmarks primarily focus on step-level guidance, providing limited supervision and evaluation for understanding the progress and completion of guided events across procedural levels.

Beyond understanding task progress at a fixed guidance level, practical proactive assistance should also support different guidance granularities according to the user's expertise and needs. For example, a user familiar with making pasta sauce may only need phase-level guidance such as "Make the aglio e olio sauce," whereas a less experienced user may require more detailed step- or action-level guidance. Moreover, the desired guidance level may change during an ongoing task. This motivates evaluating whether a model can provide guidance across different procedural levels and adapt when the requested guidance level changes.

To support both training and evaluation of hierarchical proactive assistants, we introduce *ProactiveCoach-Instruct* and *ProactiveCoachBench*. ProactiveCoach-Instruct provides hierarchically structured guidance for goal-relevant activities at the phase, step, and action levels. To construct this guidance, we identify activities relevant to the task goal, organize actions into steps and steps into phases, and generate level-specific guidance with guidance timing. ProactiveCoachBench evaluates proactive guidance at each level and further assesses whether models can adapt when the requested guidance level changes during an ongoing task.

Using ProactiveCoach-Instruct, we propose *ProactiveCoach*, a hierarchical guidance learning method that jointly learns phase-, step-, and action-level proactive guidance in a single vision-language model. ProactiveCoach organizes guidance across the three levels into a single output sequence that preserves their hierarchical relationships and guidance timing, enabling the model to learn what guidance to provide and when at each level. Across different backbone models, ProactiveCoach consistently improves proactive assistance performance, including step-level guidance compared with step-only supervision. We further build an adaptive guidance system in which ProactiveCoach generates multi-level guidance and a lightweight router selects the level requested by the user. This enables proactive assistance to adapt its guidance granularity as the user's needs change.

Dataset	#Videos	Hours	Proactive	Procedural Tasks	Guidance Tasks	Hierarchical Ann.
HiREST	3.4K	0.3K	✗	✓	✗	✓
GUIDE	3.5K	0.1K	✗	✓	✗	✓
Ego4D-HCap	4.5K	–	✗	✗	✗	✓
Action100M	1.2M	128K	✗	✓	✗	✓
ProAssist	3.4K	0.5K	✓	✓	✓	✗
StreamPro-Bench	0.6K	–	✓	△	△	✗
Pro^{2}Bench	29.3K	–	✓	✓	✓	✗
GuideMe	2.5K	0.2K	✓	✓	✓	✗
ProactiveCoach (Ours)	**7.2K**	**1.5K**	✓	✓	✓	✓

Table 1: Comparison of existing datasets for hierarchical video understanding and proactive procedural assistance. △ denotes partial task coverage.

In summary, our contributions are as follows:

- We introduce **ProactiveCoach-Instruct**, a training dataset with hierarchically structured guidance, and **ProactiveCoachBench**, a benchmark for multi-level proactive guidance and guidance-level adaptation.
- We propose **ProactiveCoach**, a hierarchical guidance learning method for phase-, step-, and action-level proactive guidance. It consistently improves performance across different backbones and over step-only supervision.
- We develop an **adaptive guidance system** that combines ProactiveCoach with a lightweight router, adapting the guidance level to the user's request during an ongoing task.

2 Related Work

Hierarchical Video Understanding. Hierarchical video understanding aims to understand video content at multiple levels, from fine-grained actions to higher-level activities (Song et al., 2023; Islam et al., 2024; Kang et al., 2025). HiREST (Zala et al., 2023) connects video and moment retrieval with step-by-step descriptions, while GUIDE (Liang et al., 2024) identifies common procedures across multiple videos and links them to the detailed steps in each video. Ego4D-HCap (Islam et al., 2024) recursively generates captions at multiple levels, from short action descriptions to long-form video summaries, while Action100M (Chen et al., 2026) hierarchically segments videos and provides action and context annotations for segments of different durations. However, hierarchical descriptions cannot be directly converted into guidance. In proactive procedural assistance, the assistant must identify goal-relevant activities and determine what guidance to provide, when to provide it, and at what level of detail.

Proactive Procedural Assistance. Proactive procedural assistance aims to provide task guidance, requiring models to learn what guidance to provide and when to provide it. ProAssist (Zhang et al., 2025b) and StreamPro-Bench (Li et al., 2026a) focus on providing proactive step-level guidance according to task progress. Pro^{2}Bench (Kundu et al., 2026) further considers recognizing when the user's behavior does not follow the intended procedure and providing recovery guidance, while GuideMe (Liu et al., 2026) covers task completion verification, error detection, and corrective guidance. However, existing datasets and benchmarks primarily focus on step-level guidance, providing limited supervision and evaluation for understanding progress and completion across procedural levels. They also provide limited evaluation of whether models can provide guidance across different procedural levels and adapt when the requested guidance level changes.

3 ProactiveCoachBench

In this section, we introduce ProactiveCoach-Instruct, our training dataset, and ProactiveCoach-Bench, our evaluation benchmark. we describe their design and statistics in Sec. 3.1, present the construction pipeline in Sec. 3.2, and show quality review in Sec. 3.3



Figure 2: Overview of hierarchically structured guidance in PROACTIVECOACHBENCH.

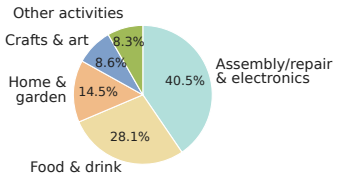


Figure 3: Domain distribution of our datset.

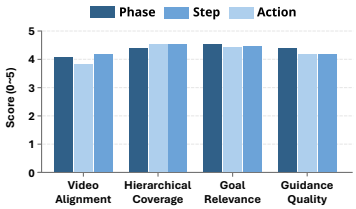


Figure 4: The quality of PROACTIVECOACHBENCH.

3.1 DATASET OVERVIEW

To support hierarchical procedural understanding for proactive procedural assistance, we structure goal-relevant guidance into three levels: phase, step, and action. This hierarchy allows lower-level progress to provide evidence for the completion of guided events, while higher-level context helps clarify their purpose and expected outcome. Based on this design, we construct PROACTIVECOACH-INSTRUCT and PROACTIVECOACHBENCH with hierarchically structured guidance and guidance timing at each level, supporting learning and evaluation across procedural levels.

Figure 2 illustrates the hierarchically structured guidance in ProactiveCoach-Instruct and Proactive-CoachBench. A phase is a high-level unit that achieves an intermediate goal through multiple steps, while a step is a procedural unit that achieves a specific outcome through multiple actions. An action is an observable fine-grained event that can be guided independently and serves as the finest guidance level. Each sample consists of a task goal, a video, and hierarchical guidance annotations, with each guidance sentence linked to its event span and guidance time. Guidance timing is defined independently at each level. The next guidance is provided immediately after the previously guided event at that level ends. For each video chunk, we combine multi-level guidance in phase, step, and action order into a single `<response>`. If no new guidance is required at any level, the chunk is labeled `<silent>`.

The data contain 11,119 samples constructed from 7,232 first-person videos across eight source datasets: Ego4D Goal-Step (Song et al., 2023), Ego-Exo4D (Grauman et al., 2024), EPIC-KITCHENS (Damen et al., 2018), CaptainCook4D (Peddi et al., 2024), EgoExoLearn (Huang et al., 2024), HD-EPIC (Perrett et al., 2025), WTaG (Bao et al., 2023), and HoloAssist (Wang et al., 2023). As shown in Figure 3, the data cover diverse domains, including assembly, repair, and electronics; food and beverage preparation; and household and gardening activities. We use 10,007 samples as ProactiveCoach-Instruct for training and 1,112 samples as ProactiveCoachBench for evaluation. On average, each sequence contains 4.14 phases, with 3.35 steps per phase and 2.47 actions per step. Further dataset statistics are provided in Appendix A.

3.2 CONSTRUCTION PIPELINE

The proposed dataset is constructed through three stages: procedural data curation, procedure hierarchy construction, and proactive guidance generation, as shown in Figure 5.

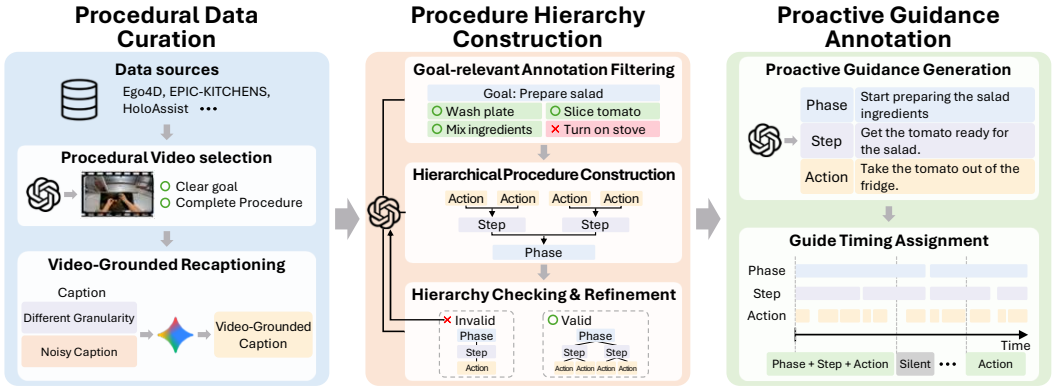


Figure 5: Construction pipeline for our proposed dataset.

Procedural Data Curation. We filter goal-oriented procedural videos from eight egocentric video datasets. The source datasets may contain videos that are unsuitable for procedural assistance, such as those with unclear task goals or insufficient coverage of key events. Therefore, in procedural video selection, we use GPT-4.1-mini to assess the clarity of the task goal and the coverage of key events based on the existing annotations, and retain videos suitable for procedural assistance. In addition, the source datasets contain annotations with different levels of granularity, inconsistencies with the video, or missing relevant events. To address these issues, we perform video-grounded recaptioning using Gemini 2.5 Flash with the selected videos and their existing annotations. The model revises captions that do not match the video, adds missing captions, and aligns each caption with its corresponding temporal segment in the video.

Procedure Hierarchy Construction. The recaptioned annotations may still include content that is not necessary for accomplishing the task goal. Therefore, in goal-relevant annotation filtering, we use GPT-4.1-mini to retain only the annotations necessary for achieving the task goal. Then, we perform hierarchical procedure construction to organize the selected annotations into a procedural hierarchy. First, we consolidate the selected annotations into actions, each representing an independent unit of execution. We then group multiple actions into a step representing a procedural unit, and multiple steps into a phase representing an intermediate goal, forming a phase–step–action hierarchy. The temporal span of each action is defined by its observed start and end times in the video. The temporal spans of each step and phase are then defined to cover their child actions and steps, respectively. This produces a hierarchical procedure structure that serves as the basis for hierarchically structured guidance. However, this process may produce redundant structures where different hierarchical levels describe the same execution. We therefore perform hierarchy checking and refinement to eliminate these redundancies and ensure hierarchical consistency. Using GPT-4.1-mini, we check whether each step is properly composed of its actions, each phase is properly composed of its steps, and their temporal spans are consistent. Based on this check, we reconstruct the hierarchy and repeat the refinement process until the structure is consistent.

Proactive Guidance Generation. To construct hierarchically structured guidance, we perform level-specific guidance generation. Given the constructed phase–step–action hierarchy and the user's goal information at each level, we generate proactive guidance for the phase, step, and action levels by using GPT-4.1-mini. Then, we perform guide timing assignment to align the generated guidance with task progress based on the temporal spans at each level. At each level, the next guidance is assigned to the frame immediately after the current guided event ends. If the current guided event ends at only one level, we provide the next guidance only for that level. If guided events end at multiple levels at the same time, we provide the next guidance for all corresponding levels together. If no guided event ends at a given frame, we label that frame as silent. Through this process, we construct HUPPA, a dataset of hierarchically structured guidance for proactive procedural assistance.

3.3 Quality Review

We conduct a human evaluation to assess whether the annotations accurately reflect the procedural structure of the video and provide appropriate guidance. We sample 50 examples from ProactiveCoachBench and evaluate them at the phase, step, and action levels using four criteria: *video*

alignment, *hierarchical coverage*, *goal relevance*, and *guidance quality*. *Video alignment* measures whether the annotated content, temporal spans, and guidance timings are consistent with the observed video. *Hierarchical coverage* evaluates whether the lower-level executions sufficiently cover the content required by the higher-level procedure, while *goal relevance* measures whether the generated guidance contributes to achieving the task goal. *Guidance quality* evaluates whether each guidance sentence clearly and accurately describes what the user should do at the corresponding level. As shown in Figure 4, the annotations achieve scores above 4.0 on most evaluation criteria, with an overall score of 4.27. These results indicate that the annotations reliably capture video-grounded procedural hierarchies and provide goal-relevant guidance at each level. Details of quality review are provided in Appendix D.

4 PROACTIVECOACH

We propose ProactiveCoach, a hierarchical guidance learning method that jointly learns phase-, step-, and action-level proactive guidance in a single vision-language model (VLM). Using ProactiveCoach-Instruct, we train the model to learn hierarchical procedural understanding across the three levels. For supervised fine-tuning (SFT), we organize guidance across the three levels into a single output sequence that preserves their hierarchical relationships and guidance timing. Each guidance sentence is paired with its level and a hierarchical ID that encodes the parent–child structure. This makes the relationship between lower-level events and higher-level procedural units explicit in the training targets.

Let I denote the instruction specifying the task goal, V_t^w the video frames from up to w consecutive chunks ending at the current chunk t, and y_t the ground-truth output for that chunk. We denote the ground-truth output history by $H_t^{\mathrm{GT}} = (y_1, \ldots, y_{t-1})$. Given I, V_t^w, and H_t^{GT}, the model is trained to output `<silent>` when no new guidance is needed and `<response>` followed by the corresponding guidance otherwise. If new guidance is needed at one level, the response contains guidance for that level. If new guidance is needed at multiple levels, the corresponding guidance is combined into a single response. During training, we use teacher forcing and provide previous ground-truth outputs as the output history.

In procedural videos, `<silent>` labels are more frequent than `<response>` labels. To prevent repeated silence from dominating the loss, we focus supervision on transitions between response and silence, following a prior work (Li et al., 2026b). Specifically, we select all `<response>` chunks and each `<silent>` chunk immediately following a `<response>` chunk for supervision. We then randomly sample additional `<silent>` chunks from the remaining chunks, matching the total number of chunks selected above. We compute the loss only on output tokens selected for supervision. The objective is as follows:

$$\mathcal{L}_{\mathrm{SFT}}(\theta) = -\frac{1}{M} \sum_{t=1}^{T} m_t \log p_\theta \left(y_t \mid I, V_t^w, H_t^{\mathrm{GT}} \right). \tag{1}$$

Here, T is the total number of chunks, and $m_t \in 0, 1$ indicates whether chunk t is selected for supervision. We normalize the loss by $M = \sum_{t=1}^{T} m_t$, the total number of selected chunks. Training on hierarchically structured guidance encourages the model to understand how actions make up each step and how each step belongs to a higher-level phase. Understanding how each action is progressing can provide useful evidence for determining whether the current guided step is complete, while higher-level phase context can help clarify the purpose of the current step and its expected outcome. Together with the observed video, this hierarchical understanding provides a stronger basis for deciding what guidance to provide and when to provide it at each level.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Datasets and Baselines. We evaluate on ProactiveCoachBench and the publicly available validation split of EgoProactive, an existing benchmark for proactive procedural assistance. We evaluate guidance performance at the phase, step, and action levels on ProactiveCoachBench, and at the step level on EgoProactive (Kundu et al., 2026). We compare against four groups of models: Proprietary Frontier Models (Comanici et al., 2025; Anthropic, 2025; Singh et al., 2025), Multimodal

Models (Bai et al., 2025; Team, 2026; Yan et al., 2026b; Zhu et al., 2025; Zhang et al., 2025a; 2024), Guidance Assistant Models (Zhang et al., 2025b), and Streaming Models (Chen et al., 2024; Wang et al., 2026; Li et al., 2026b). For guidance assistant models, we use ProAssist, whose model weights are publicly available, and for streaming models, we use VideoLLM-Online, MMDuet2, and VideoChat3-4B as baselines. The details of baselines are provided in Appendix E.2.

Evaluation Protocol. Following prior evaluation protocols (Kundu et al., 2026; Liu et al., 2026), we evaluate models at predefined anchor points. These anchor points are categorized into *Response Anchors* and *Silent Anchors*. At *Response Anchors*, the model is expected to generate appropriate guidance, whereas at *Silent Anchors*, it is expected to remain silent and avoid providing unnecessary guidance. For ProactiveCoachBench, we use all ground-truth guidance timestamps as *Response Anchors*. For *Silent Anchors*, we randomly sample a timestamp with no ground-truth guidance from the central 30–70% of each phase, step, or action segment. For EgoProactive, we use the original evaluation anchor points. At each anchor, all models receive the most recent 10 seconds of video preceding the anchor, together with the complete ground-truth dialogue history prior to that point. We use ground-truth dialogue history rather than model-generated responses to prevent previous incorrect or missed guidance from affecting subsequent decisions, enabling a fair comparison across models under the same dialogue history. This protocol therefore evaluates both whether the model correctly decides when to provide guidance and whether the provided guidance is appropriate.

Metrics. We use sF1, gF1, and PQS to evaluate proactive procedural assistance, following prior work (Kundu et al., 2026; Liu et al., 2026).

sF1 jointly evaluates how well the generated guidance matches the ground-truth guidance and whether the model provides all necessary guidance. We first match ground-truth and generated guidance one-to-one based on both timing and content, and sum the semantic similarities of the matched pairs. Then, we compute sPrecision and sRecall by dividing this sum by the numbers of generated and ground-truth guidance instances, respectively. sF1 is calculated as the harmonic mean of sPrecision and sRecall (Liu et al., 2026).

gF1 evaluates whether the model correctly decides when to provide guidance and when to remain silent. It considers only whether the model responds or remains silent at each anchor point and does not consider the guidance content. We compute precision and recall separately for response and silence, obtain the corresponding F1 scores, and calculate gF1 as their geometric mean (Kundu et al., 2026).

PQS jointly evaluates response–silence decisions and the quality of the provided guidance. At a silent anchor, the model receives a score if it remains silent and receives zero otherwise. At a response anchor, the model receives a score based on the relevance, specificity, actionability, and conciseness of the generated guidance, and receives zero if it remains silent. PQS is computed by averaging these scores across all anchor points (Kundu et al., 2026). We use GPT-5.2 to evaluate guidance content quality. Details of the metrics are provided in Appendix F.

Implementation Details. We use Qwen3-VL-4B/8B and Qwen3.5-4B/9B as backbone models and train for three epochs using eight B200 GPUs. Videos are sampled at 1 FPS and grouped into 2-second chunks, with the model deciding whether to respond at each chunk. We apply a causal block-sparse attention mask that restricts each token to frames from the current and preceding 10 seconds, while preserving access to the ground-truth instruction and all previous response and silence tokens. Detailed training configurations are provided in Appendix E.1.

5.2 Main Results

As shown in Table 2, our method consistently achieves higher overall performance than the baselines across all guidance levels on ProactiveCoachBench. A higher sF1 indicates that the generated guidance is semantically well aligned with the ground-truth guidance while sufficiently covering the task goal. A higher gF1 indicates that the model more accurately determines when to provide guidance and when to remain silent. A higher PQS indicates better overall performance in both response–silence decisions and guidance quality. Our method shows strong performance across all metrics, with consistent improvements across different backbones.

Model	Ours	ProactiveCoachBench												EgoProactive			
		Phase				Step				Action							
		sF1	gF1	PQS	Avg.↑	sF1	gF1	PQS	Avg.↑	sF1	gF1	PQS	Avg.↑	sF1	gF1	PQS	Avg.↑
Proprietary Frontier Models																	
Gemini 3 Flash Preview		43.01	70.36	52.87	55.41	41.15	59.00	41.88	47.34	45.08	57.45	36.88	46.47	38.62	64.69	50.73	51.35
Claude Haiku 4.5		43.25	27.70	18.18	29.71	41.51	22.54	22.89	28.98	41.60	23.90	19.80	28.43	38.81	52.63	34.20	41.88
GPT-5.4 mini		33.66	50.03	31.23	38.31	32.47	51.21	34.47	39.38	34.47	53.87	35.82	41.39	39.71	53.60	36.18	43.17
Guidance Assist Models																	
ProAssist-8B		11.06	38.86	49.59	33.17	9.45	33.72	49.03	30.73	9.83	30.96	48.45	29.75	6.07	24.64	45.74	25.49
Streaming Models																	
VideoLLM-Online		0.45	8.97	49.97	19.80	0.32	6.48	49.95	18.92	0.34	5.99	49.98	18.77	0.52	9.44	46.17	18.71
MMDuet2		9.97	40.41	41.98	30.79	19.96	48.25	37.93	35.38	19.80	49.25	36.38	35.14	30.63	44.71	27.53	34.29
VideoChat3-4B		35.91	19.12	14.38	23.14	35.12	25.63	19.85	26.87	37.83	18.00	16.64	24.16	39.46	36.32	24.21	33.33
Multimodal Models																	
VideoChat-R1.5-7B		0.05	2.41	49.81	17.42	0.07	3.93	49.72	17.91	0.11	3.33	49.43	17.62	0.20	4.67	45.69	16.85
InternVL3-8B		13.86	45.11	41.13	33.37	25.12	52.14	38.96	38.74	34.71	54.90	34.36	41.32	18.04	38.08	43.38	33.17
VideoLLaMA3-7B		15.90	44.46	35.63	31.99	21.66	48.82	38.15	36.21	27.74	50.08	34.16	37.33	1.07	9.14	41.80	17.34
LLaVA-Video-7B		20.34	47.74	35.26	34.45	30.19	51.65	34.90	38.92	25.34	51.55	40.10	39.00	35.73	47.65	33.05	38.81
Qwen3-VL-32B		41.56	38.50	22.11	34.06	38.46	23.89	21.71	28.02	38.85	20.84	17.25	25.65	32.06	49.81	40.14	40.67
Qwen3-VL-4B		35.93	25.82	18.19	26.65	36.50	26.01	20.99	27.83	37.46	27.78	18.65	27.96	**39.93**	35.90	24.33	33.39
	✓	**53.15**	**74.40**	**60.53**	**62.69**	**49.30**	**71.58**	**55.99**	**58.96**	**49.72**	**64.69**	**45.32**	**53.24**	36.48	**62.78**	**46.50**	**48.59**
Qwen3-VL-8B		40.93	9.54	17.35	22.60	38.39	9.56	20.03	22.66	37.70	8.99	15.82	20.84	**41.96**	2.47	23.06	22.50
	✓	**54.63**	**76.21**	**61.37**	**64.07**	**50.24**	**72.02**	**55.83**	**59.37**	**50.07**	**64.12**	**44.77**	**52.99**	35.28	**63.06**	**48.55**	**48.96**
Qwen3.5-4B		19.20	47.85	35.71	34.25	20.86	49.10	39.90	36.62	21.76	47.81	40.81	36.79	14.84	36.68	44.22	31.91
	✓	**52.87**	**75.15**	**60.79**	**62.94**	**49.35**	**72.27**	**56.27**	**59.30**	**50.43**	**64.75**	**45.15**	**53.44**	**35.76**	**60.06**	**45.61**	**47.14**
Qwen3.5-9B		40.10	11.65	17.42	23.06	38.38	26.90	21.54	28.94	40.71	26.86	18.92	28.83	**37.10**	25.85	21.87	28.27
	✓	**50.50**	**72.79**	**60.08**	**61.13**	**48.24**	**71.29**	**56.90**	**58.81**	**50.56**	**66.17**	**47.58**	**54.77**	36.67	**62.17**	**46.73**	**48.52**

Table 2: Proactive guidance results on ProactiveCoachBench and the EgoProactive validation set.

Our method also shows strong performance across all metrics on the EgoProactive validation set. Some base models achieve relatively high sF1 but very low gF1, indicating that they tend to respond even when no guidance is needed. Conversely, some models show relatively low sF1 but high PQS, indicating that they tend to remain silent even when a response is required. Our method consistently achieves higher Avg. scores than the base models across various backbones.

5.3 Ablation Studies and Analysis

Hierarchical vs. Step-Only Supervision. Table 3 compares step-only supervision with hierarchical supervision that jointly learns phase-, step-, and action-level guidance across different backbones. Hierarchical supervision consistently improves sF1, gF1, PQS, and Avg. across all backbones. These results show that learning hierarchically structured guidance not only enables guidance at multiple levels but also improves step-level proactive assistance.

Effect of Phase- and Action-Level Supervision. Table 4 compares step-only, phase–step, step–action, and phase–step–action supervision to analyze the effect of additional guidance levels. Adding either phase- or action-level supervision improves step-level performance over step-only supervision, and using all three levels achieves the highest performances. These results suggest that action-level supervision helps the model understand what actions make up each step and how they are progressing, providing useful evidence for determining whether the step is complete. Phase-level supervision provides complementary context about the purpose and expected outcome of the current step and how it contributes to the overall procedure. These results support the benefit of learning hierarchically structured guidance for step-level proactive assistance.

Zero-Shot Multi-Level Guidance. We examine whether a base model can perform multi-level proactive guidance through prompting alone, without fine-tuning on hierarchically structured guidance. Table 5 compares zero-shot Qwen3.5-4B with our fine-tuned model under the same multi-level prompt and three-level GT history. Fine-tuning substantially improves performance across all three guidance levels. These results suggest that prompting alone is insufficient for reliable multi-level proactive guidance, while training on hierarchically structured guidance helps the model learn the relationships across procedural levels and provide appropriate guidance at each level.

Method	HUPPA-Bench Step			
	sF1	gF1	PQS	Avg.
Qwen3-VL-4B				
Step-only	42.85	63.34	41.78	49.32
Multi-level (Ours)	**49.30**	**71.58**	**55.99**	**58.96**
Qwen3-VL-8B				
Step-only	45.82	66.12	47.27	53.07
Multi-level (Ours)	**50.24**	**72.02**	**55.83**	**59.37**
Qwen3.5-4B				
Step-only	41.01	64.53	44.62	50.06
Multi-level (Ours)	**49.35**	**72.27**	**56.27**	**59.30**
Qwen3.5-9B				
Step-only	42.56	65.21	45.77	51.18
Multi-level (Ours)	**48.24**	**71.29**	**56.90**	**58.81**

Table 3: Step-only vs. hierarchical supervision.

Method	HUPPA-Bench Step			
	sF1	gF1	PQS	Avg.
Qwen3.5-4B				
Step-only	41.01	64.53	44.62	50.06
Phase-Step	48.21	66.22	46.37	53.60
Step-Action	46.16	70.81	55.02	57.33
Phase-Step-Action	**49.35**	**72.27**	**56.27**	**59.30**

Table 4: Ablation on supervision levels.

Transition	FT	sF1	gF1	PQS	Avg.
Step→Phase		24.18	1.52	2.69	9.46
	✓	**67.66**	**93.78**	**95.17**	**85.54**
Step→Action		34.24	8.43	13.28	18.65
	✓	**54.14**	**67.95**	**51.69**	**57.93**
Phase→Step		15.89	5.50	4.07	8.49
	✓	**47.89**	**72.93**	**78.86**	**66.56**
Action→Step		38.06	9.73	8.32	18.70
	✓	**59.98**	**82.68**	**78.45**	**73.70**

Table 6: Guidance-level adaptation.

FT	Phase				Step				Action			
	sF1	gF1	PQS	Avg.	sF1	gF1	PQS	Avg.	sF1	gF1	PQS	Avg.
	8.53	32.52	46.92	29.32	7.40	29.23	44.33	26.99	38.44	11.10	15.67	21.74
✓	**52.87**	**75.15**	**60.79**	**62.94**	**49.35**	**72.27**	**56.27**	**59.30**	**50.43**	**64.75**	**45.15**	**53.44**

Table 5: Zero-shot multi-level guidance.

Guidance-Level Adaptation. A practical proactive procedural assistant should adapt its guidance level when the user's request changes during an ongoing task. As illustrated in Figure 6, our system separates guidance generation from guidance-level selection. Our trained model generates guidance at the three levels, while a lightweight LLM router selects which level to present based on the user's request. Because the two modules operate asynchronously and independently, the displayed guidance level can change without modifying or fine-tuning the guide model.

We evaluate this on four level transitions: Step→Phase, Step→Action, Phase→Step, and Action→Step. For the baseline, the level-change request is added to the dialogue history, and the model is prompted to adjust its guidance level accordingly. In our system, the router handles the level change while the guide model continues to generate multi-level guidance. As shown in Table 6, our system substantially improves performance across all transition scenarios. For the baseline, the previous dialogue history contains guidance consistently provided at the original level, making it difficult to adapt to a new level through in-context prompting alone. In contrast, separating guidance generation from level selection allows our system to change the displayed guidance level without being constrained by the previous guidance history. These results show that our design can reliably adapt guidance granularity to the user's request without additional fine-tuning of the guide model.

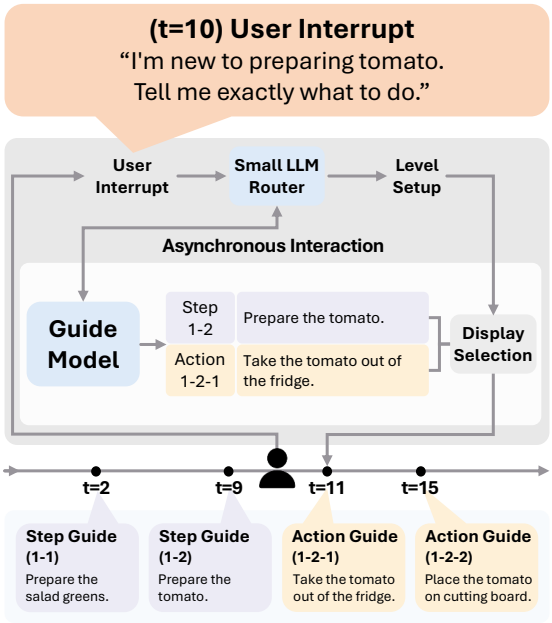


Figure 6: Guidance-level adaptation with a lightweight router. ProactiveCoach generates multi-level guidance, while the router selects the level requested by the user.

6 Conclusion

In this work, we explored hierarchical understanding for proactive procedural assistance. Existing datasets primarily focus on step-level guidance, providing limited supervision and evaluation across procedural levels and for guidance-level adaptation. We introduced ProactiveCoach-Instruct for training and ProactiveCoachBench for evaluation, providing hierarchically structured guidance at the phase, step, and action levels with guidance timing. We further proposed ProactiveCoach, a hierarchical guidance learning method that jointly learns this multi-level guidance in a single vision-language model. Our method consistently improves proactive assistance across different backbones and supports guidance-level adaptation during an ongoing task. We believe that our proposed dataset and method will provide a useful foundation for future research on proactive procedural assistance.

AI USE STATEMENT

In this work, we used generative AI tools for literature exploration, research and experimental design, code development, manuscript drafting, translation and editing, figure preparation, dataset construction, and automated evaluation. We have reviewed all AI-assisted work and take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

Our proposed dataset is constructed from publicly available video datasets and does not involve the collection of new personal information. We use the source datasets in accordance with their respective licenses and terms of use, and any released data will follow the same conditions.

REPRODUCIBILITY STATEMENT

To support reproducibility, we provide details of the dataset construction pipeline, training and evaluation settings, prompts, and metrics in the main paper and appendix. The supplementary materials include the training and inference code, evaluation implementation, and additional details for reproducing our experiments. We also provide an anonymous repository with a dataset preview and related materials, and plan to release the data and code.

REFERENCES

Anthropic. Claude haiku 4.5, 2025. URL https://www.anthropic.com/claude/haiku. Large language model.

Shehreen Azad, Vibhav Vineet, and Yogesh S Rawat. Streamready: Learning what to answer and when in long streaming videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 40494–40504, 2026.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Yuwei Bao, Keunwoo Yu, Yichi Zhang, Shane Storks, Itamar Bar-Yossef, Alex de la Iglesia, Megan Su, Xiao Zheng, and Joyce Chai. Can foundation models watch, talk and guide you step by step to make a cake? In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 12325–12341, 2023.

Delong Chen, Tejaswi Kasarla, Yejin Bang, Mustafa Shukor, Willy Chung, Jade Yu, Allen Bolourchi, Theo Moutakanni, and Pascale Fung. Action100m: A large-scale video action dataset. *arXiv preprint arXiv:2601.10592*, 2026.

Joya Chen, Zhaoyang Lv, Shiwei Wu, Kevin Qinghong Lin, Chenan Song, Difei Gao, Jia-Wei Liu, Ziteng Gao, Dongxing Mao, and Mike Zheng Shou. Videollm-online: Online video large language model for streaming video. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18407–18418. IEEE, 2024.

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

Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. Scaling egocentric vision: The epic-kitchens dataset. In *European conference on computer vision*, pp. 753–771. Springer, 2018.

Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, et al. Ego-exo4d: Understanding skilled human activity from first-and third-person perspectives. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 19383–19400, 2024.

Yifei Huang, Guo Chen, Jilan Xu, Mingfang Zhang, Lijin Yang, Baoqi Pei, Hongjie Zhang, Lu Dong, Yali Wang, Limin Wang, et al. Egoexolearn: A dataset for bridging asynchronous ego-and exo-centric view of procedural activities in real world. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22072–22086. IEEE, 2024.

Md Mohaiminul Islam, Ngan Ho, Xitong Yang, Tushar Nagarajan, Lorenzo Torresani, and Gedas Bertasius. Video recap: Recursive captioning of hour-long videos. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18198–18208. IEEE, 2024.

Hyolim Kang, Yunsu Park, Youngbeom Yoo, Yeeun Choi, and Seon Joo Kim. Open-ended hierarchical streaming video understanding with vision language models. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 20715–20725. IEEE, 2025.

Kaustav Kundu, Ritvik Shrivastava, Maxim Arap, Nanshu Wang, Xianhui Zhu, Quintin Fettes, Gautam Tiwari, Parth Suresh, Théo Moutakanni, Alejandro Castillejo Munoz, et al. Plan, watch, recover: A benchmark and architectures for proactive procedural assistance. *arXiv preprint arXiv:2606.04970*, 2026.

Ao Li, Zihan Xiao, Zihao Yue, Boshen Xu, Linli Yao, Jiaze Li, Pei Fu, Jianzhong Ju, Jian Luan, and Qin Jin. Streampro: From reactive perception to proactive decision-making in streaming video. *arXiv preprint arXiv:2605.16381*, 2026a.

Xinhao Li, Yuhan Zhu, Xiangyu Zeng, Yuhao Dong, Haoning Wu, Zhiqiu Zhang, Yuandong Yang, Changlian Ma, Qingyu Zhang, Yansong Shi, et al. Videochat3: Fully open video mllm for efficient and generalist video understanding. *arXiv preprint arXiv:2607.14935*, 2026b.

Jiafeng Liang, Shixin Jiang, Zekun Wang, Haojie Pan, Zerui Chen, Zheng Chu, Ming Liu, Ruiji Fu, Zhongyuan Wang, and Bing Qin. Guide: a guideline-guided dataset for instructional video comprehension. *arXiv preprint arXiv:2406.18227*, 2024.

Fang Liu, Jinpeng Chen, Ke Xu, Yuhao Liu, Huankang Guan, Xudong Lu, Yang Bo, Gerhard Hancke, Rui Liu, and Rynson WH Lau. Guideme: Multi-domain task guidance and intervention in streaming video. In *European Conference on Computer Vision*, pp. 57–74. Springer, 2026.

Ke Ma, Jiaqi Tang, Bin Guo, Xueting Han, Ruonan Xu, Qingfeng He, Ziheng Wang, Xu Wang, Qifeng Chen, Zhiwen Yu, et al. Response-g1: Explicit scene graph modeling for proactive streaming video understanding. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 44139–44153, 2026.

Rohith Peddi, Shivvrat Arya, Bharath Challa, Likhitha Pallapothula, Akshay Vyas, Bhavya Gouripeddi, Qifan Zhang, Jikai Wang, Vasundhara Komaragiri, Eric Ragan, et al. Captaincook4d: A dataset for understanding errors in procedural activities. *Advances in Neural Information Processing Systems*, 37:135626–135679, 2024.

Toby Perrett, Ahmad Darkhalil, Saptarshi Sinha, Omar Emara, Sam Pollard, Kranti Kumar Parida, Kaiting Liu, Prajwal Gatti, Siddhant Bansal, Kevin Flanagan, et al. Hd-epic: A highly-detailed egocentric video dataset. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23901–23913. IEEE, 2025.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

Yale Song, Eugene Byrne, Tushar Nagarajan, Huiyu Wang, Miguel Martin, and Lorenzo Torresani. Ego4d goal-step: Toward hierarchical understanding of procedural activities. *Advances in neural information processing systems*, 36:38863–38886, 2023.

Qwen Team. Qwen3. 5-omni technical report. *arXiv preprint arXiv:2604.15804*, 2026.

Xin Wang, Taein Kwon, Mahdi Rad, Bowen Pan, Ishani Chakraborty, Sean Andrist, Dan Bohus, Ashley Feniello, Bugra Tekin, Felipe Vieira Frujeri, et al. Holoassist: an egocentric human interaction dataset for interactive ai assistants in the real world. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 20213–20224. IEEE, 2023.

Yueqian Wang, Songxiang Liu, Disong Wang, Nuo Xu, Guanglu Wan, Huishuai Zhang, and Dongyan Zhao. Mmduet2: Enhancing proactive interaction of video mllms with multi-turn reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 23257–23270, 2026.

Jiaer Xia, Peixian Chen, Mengdan Zhang, Xing Sun, and Kaiyang Zhou. Streaming video instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 31219–31229, 2026.

Weicai Yan, Yuhong Dai, Qi Ran, Haodong Li, Wang Lin, Tao Jin, Xing Xie, Hao Liao, and Jianxun Lian. Proact-vl: A proactive videollm for real-time ai companions. *arXiv preprint arXiv:2603.03447*, 2026a.

Ziang Yan, Yinan He, Xinhao Li, Zhengrong Yue, Xiangyu Zeng, Yali Wang, Yu Qiao, Limin Wang, and Yi Wang. Videochat-r1. 5: Visual test-time scaling to reinforce multimodal reasoning by iterative perception. *Advances in Neural Information Processing Systems*, 38:119152–119184, 2026b.

Xueyang Yu, Cheng Shi, Yang Wang, and Sibei Yang. Eyes wide open: Ego proactive video-llm for streaming video. *Advances in Neural Information Processing Systems*, 38:13420–13463, 2026.

Abhay Zala, Jaemin Cho, Satwik Kottur, Xilun Chen, Barlas Oguz, Yashar Mehdad, and Mohit Bansal. Hierarchical video-moment retrieval and step-captioning. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23056–23065. IEEE, 2023.

Boqiang Zhang, Kehan Li, Zesen Cheng, Zhiqiang Hu, Yuqian Yuan, Guanzheng Chen, Sicong Leng, Yuming Jiang, Hang Zhang, Xin Li, et al. Videollama 3: Frontier multimodal foundation models for image and video understanding. *arXiv preprint arXiv:2501.13106*, 2025a.

Kairui Zhang, Zhenyu Yang, Bing Wang, Shengsheng Qian, and Changsheng Xu. Querystream: Advancing streaming video understanding with query-aware pruning and proactive response. In *The Fourteenth International Conference on Learning Representations*, 2026.

Yichi Zhang, Xin Luna Dong, Zhaojiang Lin, Andrea Madotto, Anuj Kumar, Babak Damavandi, Joyce Chai, and Seungwhan Moon. Proactive assistant dialogue generation from streaming egocentric videos. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 12044–12068, 2025b.

Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. Llava-video: Video instruction tuning with synthetic data. *arXiv preprint arXiv:2410.02713*, 2024.

Ruixiang Zhao, Jie Yang, Zijie Xin, Tianyi Wang, Fengyun Rao, Jing LYU, and Xirong Li. Omnipro: A comprehensive benchmark for omni-proactive streaming video understanding. *arXiv preprint arXiv:2605.18577*, 2026.

Yikai Zheng, Xin Ding, Yifan Yang, Shiqi Jiang, Hao Wu, Qianxi Zhang, Weijun Wang, Ting Cao, and Yunxin Liu. Em-garde: A propose-match framework for proactive streaming video understanding. *arXiv preprint arXiv:2603.19054*, 2026.

Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, et al. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*, 2025.

Appendices

A More Details of ProactiveCoach-Instruct and ProactiveCoachBench **14**

B Dataset Construction Details **14**

B.1 Procedural Data Curation 14

B.2 Procedure Hierarchy Construction 15

B.3 Proactive Guidance Generation 16

C Examples of ProactiveCoachBench **17**

D Details of Quality Review **21**

E Details of Evaluation Protocol **22**

E.1 Training Configurations 22

E.2 Evaluation Prompts 22

F Details of Metrics **23**

F.1 Soft F1 (sF1) 24

F.2 G-Mean F1 (gF1) 24

F.3 Proactive Assistance Quality Score (PQS) 24

G Additional Quantitative Results **25**

H Qualitative Results **25**

I Limitations **29**

A More Details of ProactiveCoach-Instruct and ProactiveCoachBench

Our dataset contains 11,119 samples constructed from 7,232 first-person videos across eight source datasets. As shown in Table 7, 10,007 samples are used for ProactiveCoach-Instruct and 1,112 samples for ProactiveCoachBench. Samples from the same video are assigned to the same split. As shown in Figure 7, the average sample duration is 7.8 minutes and the median is 8.8 minutes, with durations ranging from 40 seconds to 18 minutes. The total duration is 1,452.1 hours. Each sample includes the corresponding events, level-specific guidance, and guidance times. The data contain 582,084 guidance targets in total: 46,074 at the phase level, 154,507 at the step level, and 381,503 at the action level. On average, each sample contains 4.14 phases, with 3.35 steps per phase and 2.47 actions per step.

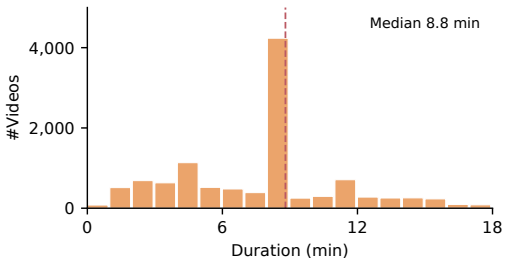


Figure 7: Sample duration distribution.

Property	Train	Test	Total
Samples	10,007	1,112	11,119
Unique video files	6,517	715	7,232
Duration (hours)	1,306.9	145.1	1,452.1

Table 7: Dataset split statistics.

B Dataset Construction Details

We construct ProactiveCoach-Instruct and ProactiveCoachBench following the pipeline described in Sec. 3.2 and Figure 5: procedural data curation, procedure hierarchy construction, and proactive guidance generation. We provide prompt details for each stage. We use Gemini 2.5 Flash for video-grounded recaptioning, GPT-4.1-mini for the remaining generation and filtering steps, and GPT-4.1 for hierarchy checking and refinement. Guide timing assignment follows a rule-based procedure.

B.1 Procedural Data Curation

Figures 8 and 9 present the prompts for procedural video selection and video-grounded recaptioning, respectively.

Procedural Video Selection (Stage 1-1)

System.
You select videos for goal-directed procedural assistance.

Input.
`video_metadata` and `source_captions`.

Requirements.
1. Assess whether the ordered captions describe a clear task goal and a recognizable outcome.
2. Score goal clarity, coverage of key operations, and logical order from 1 to 10. Repeated motions do not count as distinct procedural stages.
3. Return an overall suitability score and a short evidence-based reason. Do not assume missing operations occurred.
4. Based on these scores and reasoning, decide whether the video should be selected for procedural assistance.

Output.
A JSON object with `goal_guess`, `goal_score`, `coverage`, `order`, `overall_score`, `reasoning`, and `selected`.

Figure 8: Prompt for Procedural Video Selection.

Video-grounded Recaptioning (Stage 1-2)

System.
You produce video-grounded captions for an egocentric procedural video.

Input.
`video_clip`, `clip_metadata`, and `source_captions`.

Requirements.
1. Use source captions as references and verify each caption against the video. Preserve supported content, correct inaccurate descriptions, supplement visible omissions, and reject unsupported captions.
2. Make the caption granularity consistent across source datasets. Each caption should describe a single observable fine-grained manipulation.
3. Align each caption's event interval with the corresponding event observed in the video.
4. Describe only actions, objects, and state changes that are supported by the video. Do not infer unobserved actions or states. Use English descriptions.

Output.
A JSON object with `captions`. Each caption contains `id`, `description`, `object`, `state_change`, `event_interval`, and its source caption reference. `object` and `state_change` may be null when they are not clearly observable in the video. Newly added captions may have no source caption reference.

Figure 9: Prompt for Video-grounded Recaptioning.

B.2 Procedure Hierarchy Construction

Figures 10, 11, and 12 present the prompts for goal-relevant annotation filtering, hierarchical procedure construction, and hierarchy checking and refinement, respectively.

Goal-Relevant Annotation Filtering (Stage 2-1)

System.
You select annotations that contribute to the supplied task goal. Output only JSON.

Input.
`task_goal` and `video_grounded_annotations`.

Requirements.
1. Judge each annotation in the context of the task goal and surrounding annotations. Keep annotations that directly contribute to the task or support a necessary operation.
2. Retain preparation and supporting events, such as fetching an ingredient, measuring, or opening a container, when they serve the goal. Repeated events are not automatically irrelevant.
3. Exclude activity unrelated to the task goal, such as an unrelated conversation or camera adjustment. Do not reject an annotation merely because it does not yet fit a Step.
4. Preserve annotation IDs, descriptions, objects, state changes, and event intervals. Give an evidence-based reason for exclusions; do not invent missing operations.

Output.
A JSON object with `retained_ids` and `excluded`; each excluded entry contains `annotation_id` and `reason`. Account for every input annotation exactly once.

Figure 10: Prompt for Goal-Relevant Annotation Filtering.

Hierarchical Procedure Construction (Stage 2-2)

System.
You organize observed activity into a Phase–Step–Action hierarchy.

Input.
`task_goal`, `retained_annotations`, and `review_feedback` (if any).

Requirements.
1. Consolidate the retained annotations into observable Actions that are meaningful to guide independently. Use their descriptions, objects, state changes, and event intervals to determine which annotations belong to the same Action. Merge annotations that describe the same event while preserving distinct events, and set each Action interval to cover the event intervals of its annotations.
2. Group Actions that together form a procedural task into a Step, and group Steps that together achieve an intermediate goal into a Phase. Define levels by procedural scope, not sentence length.
3. Set each Step interval to cover its Action intervals, and each Phase interval to cover its Step intervals.
4. Each Action belongs to one Step and each Step to one Phase. Descriptions at a higher level must express the collective scope of the lower-level units they contain. Avoid a Phase–Step–Action chain that merely renames the same operation; do not create additional Actions or Steps only to increase branching.
5. When review feedback is supplied, reconstruct the affected groups using the retained annotations and preserve valid assignments. Return explicit Phase–Step, Step–Action, and annotation references.

Output.
A JSON object with `phases`, `steps`, and `actions`. Each unit contains `id`, `description`, `event_interval`, and `annotation_ids`. Each Step contains its `phase_id`, and each Action contains its `step_id`.

Figure 11: Prompt for Hierarchical Procedure Construction.

Hierarchy Checking and Refinement (Stage 2-3)

System.
You review the scope and evidence support of a proposed procedural hierarchy.

Input.
`task_goal`, `retained_annotations`, and `candidate_hierarchy`.

Requirements.
1. Check whether each lower-level unit contributes to the higher-level unit that contains it. Use the retained annotation descriptions, objects, state changes, and event intervals as evidence, and check whether relevant observed operations are missing or duplicated.
2. Flag higher-level units whose scope is not supported by the lower-level units they contain, as well as Phase–Step–Action chains that express the same operation across levels. A lower-level unit count alone is not sufficient; inspect the actual procedural scope.
3. Check Phase–Step and Step–Action references, chronological order, and whether each higher-level event interval covers the event intervals of the lower-level units it contains. Separate temporal errors from semantic grouping errors.
4. Return evidence-based reconstruction feedback for affected units to 2-2. Preserve valid groups and do not invent unobserved operations.

Output.
A JSON object with `passed`, `issues`, and `reconstruction_feedback`. Each issue identifies the affected unit, issue type, annotation IDs, and requested correction.

Figure 12: Prompt for Hierarchy Checking and Refinement.

B.3 Proactive Guidance Generation

Figure 13 presents the prompt for level-specific guidance generation. In Guide Timing Assignment (Stage 3-2), guidance timing is determined independently at the Phase, Step, and Action levels. Within the same level, the next guidance is assigned to the first frame after the current event ends. The first guidance at each level is assigned to the first frame of the video. When guidance from multiple levels is assigned to the same frame, we combine it into a single `<response>` in Phase, Step, and Action order. If no guidance is assigned at any level, the frame is labeled `<silent>`.

Level-Specific Guidance Generation (Stage 3-1)

System.
You write proactive instructions that match the scope of each Phase, Step, and Action.

Input.
`task_goal`, `hierarchy_units`, and `supporting_annotations`.

Requirements.
1. Use the task goal, hierarchy context, and supporting annotations to write an actionable imperative instruction in English.
2. Phase guidance conveys an intermediate goal; Step guidance conveys a procedural task; Action guidance conveys a concrete manipulation.
3. Do not describe a higher-level task using only its first lower-level operation. Do not include content from another unit or add unsupported details. Avoid filler.
4. Check each instruction against its corresponding Phase, Step, or Action and supporting annotations, correcting inaccurate objects, unsupported details, or scope mismatches. Preserve the supplied unit IDs. Generate content only; timing is assigned separately.

Output.
A JSON object with `guidance`; each entry contains `unit_id`, `level`, and `instruction`.

Figure 13: Prompt for Level-Specific Guidance Generation.

C EXAMPLES OF PROACTIVECOACHBENCH

These six examples illustrate the hierarchically structured guidance in ProactiveCoachBench. Red denotes the task goal, and blue shows phase-, step-, and action-level guidance at the first six consecutive guidance times.

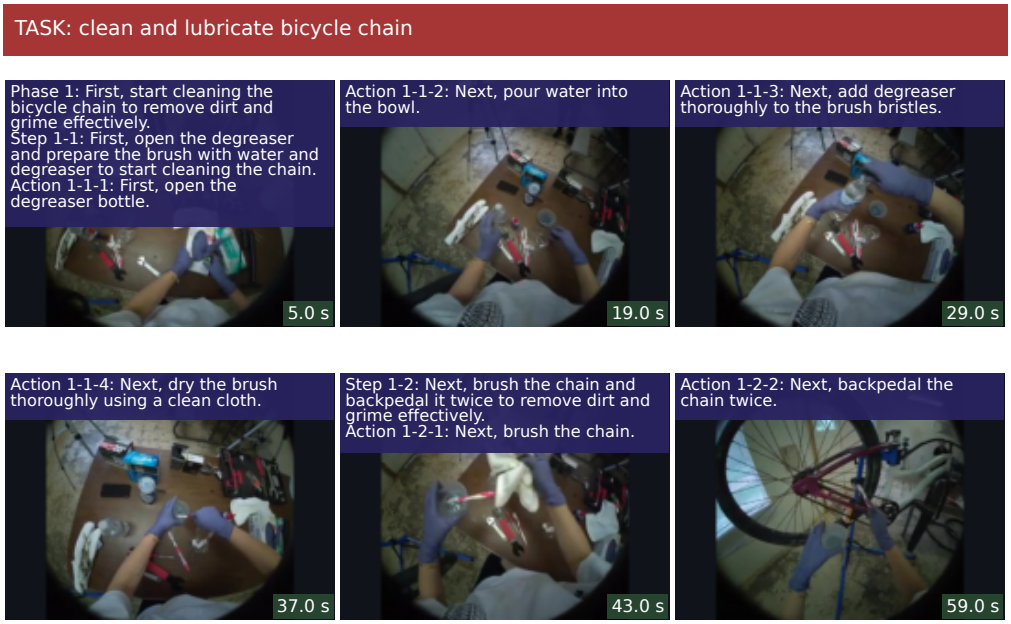


Figure 14: ProactiveCoachBench example-1 (goal: clean and lubricate a bicycle chain).

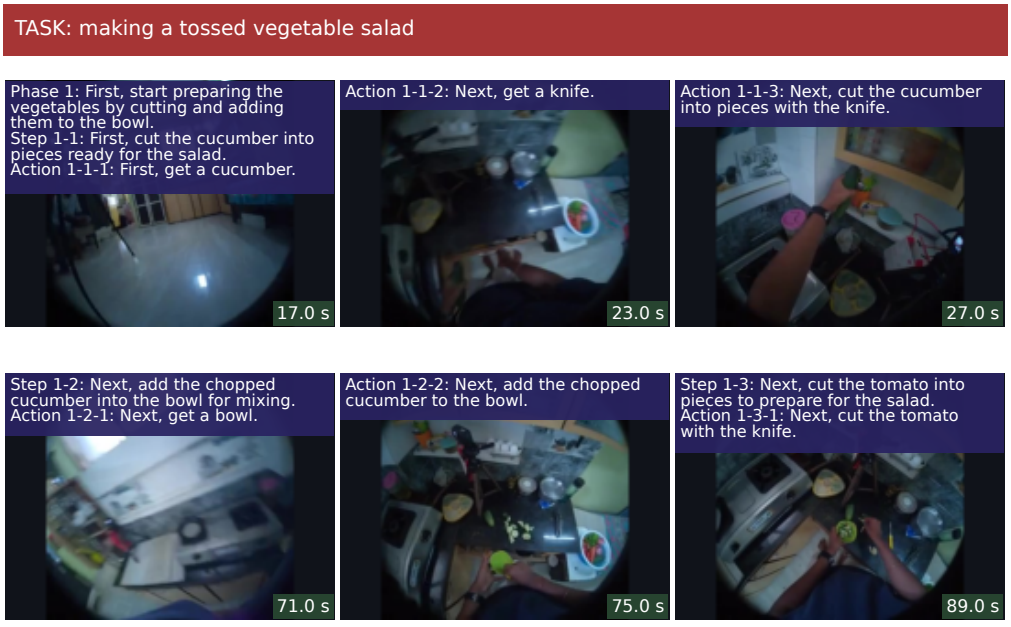


Figure 15: ProactiveCoachBench example-2 (goal: make a tossed vegetable salad).

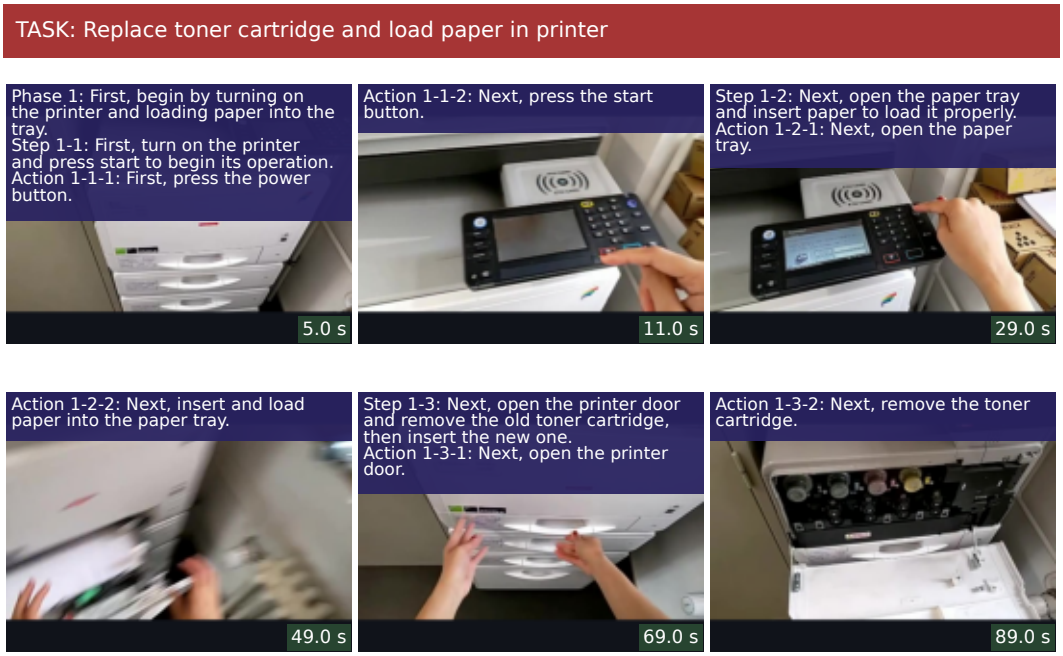


Figure 16: ProactiveCoachBench example-3 (goal: replace a toner cartridge and load paper).

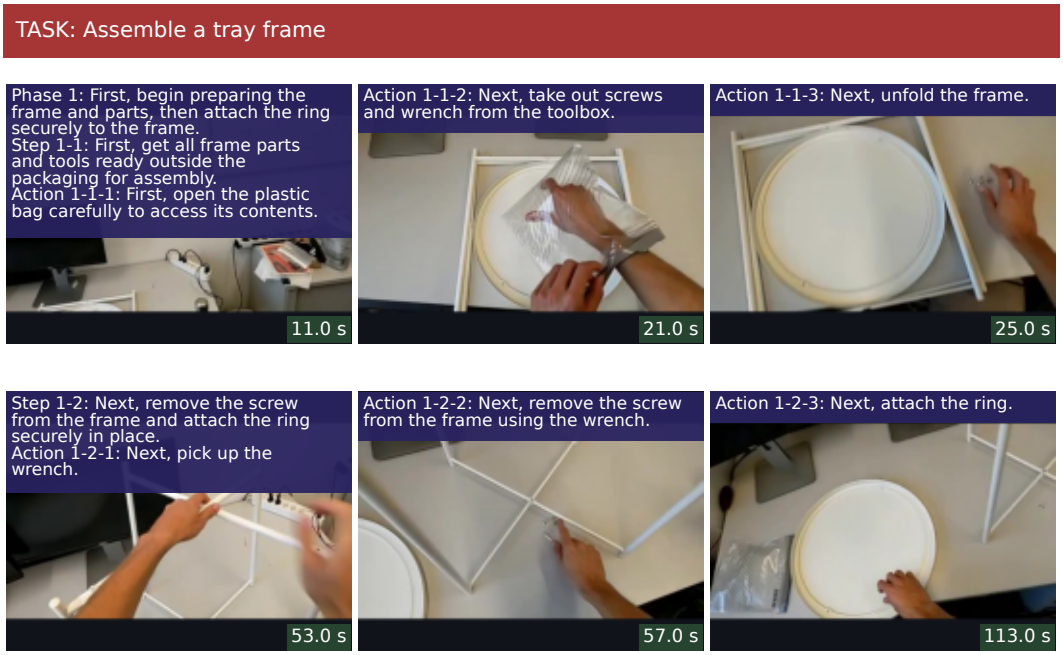


Figure 17: ProactiveCoachBench example-4 (goal: assemble a tray frame).

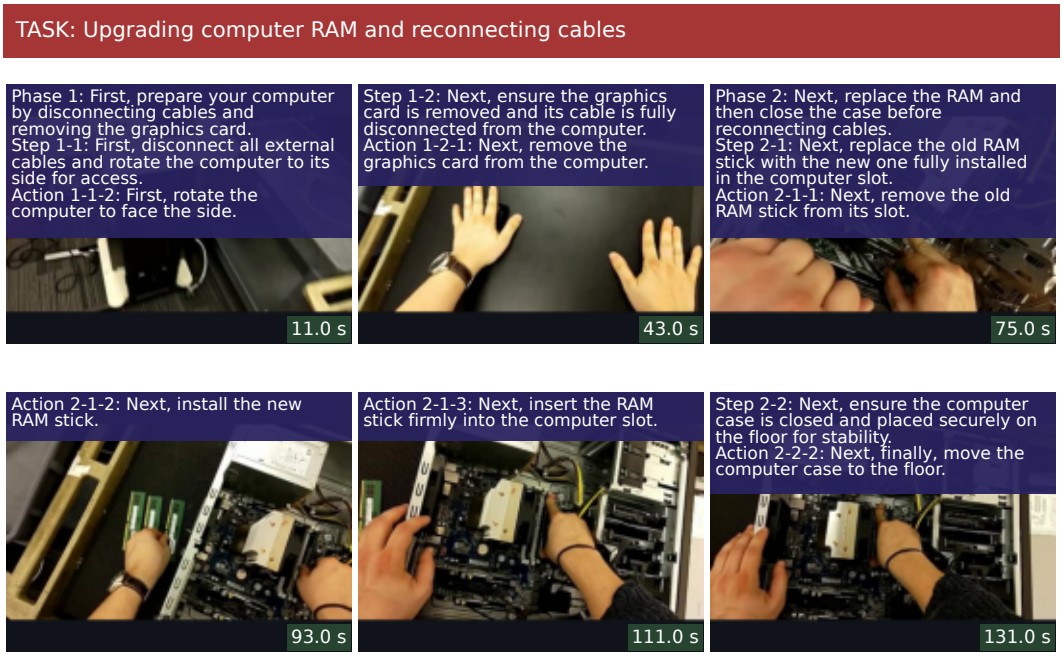


Figure 18: ProactiveCoachBench example-5 (goal: upgrade computer RAM and reconnect cables).

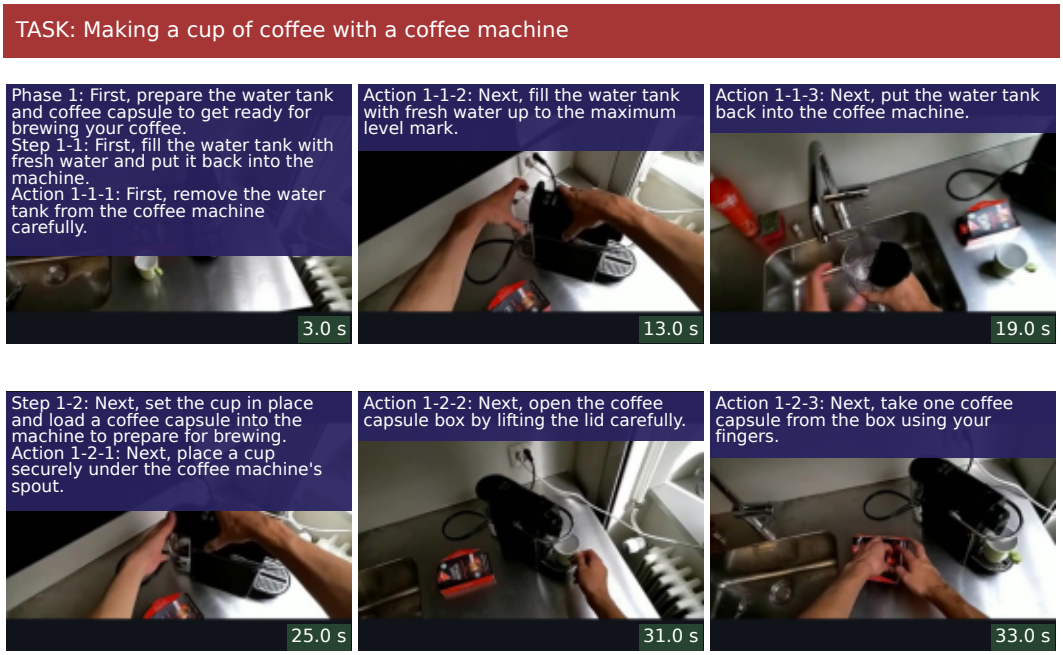


Figure 19: ProactiveCoachBench example-6 (goal: make a cup of coffee).

D Details of Quality Review

We conduct a human quality review on 50 test examples at the phase, step, and action levels using four criteria: *video alignment*, *hierarchical coverage*, *goal relevance*, and *guidance quality*. Figure 20 shows the interface used for the evaluation.

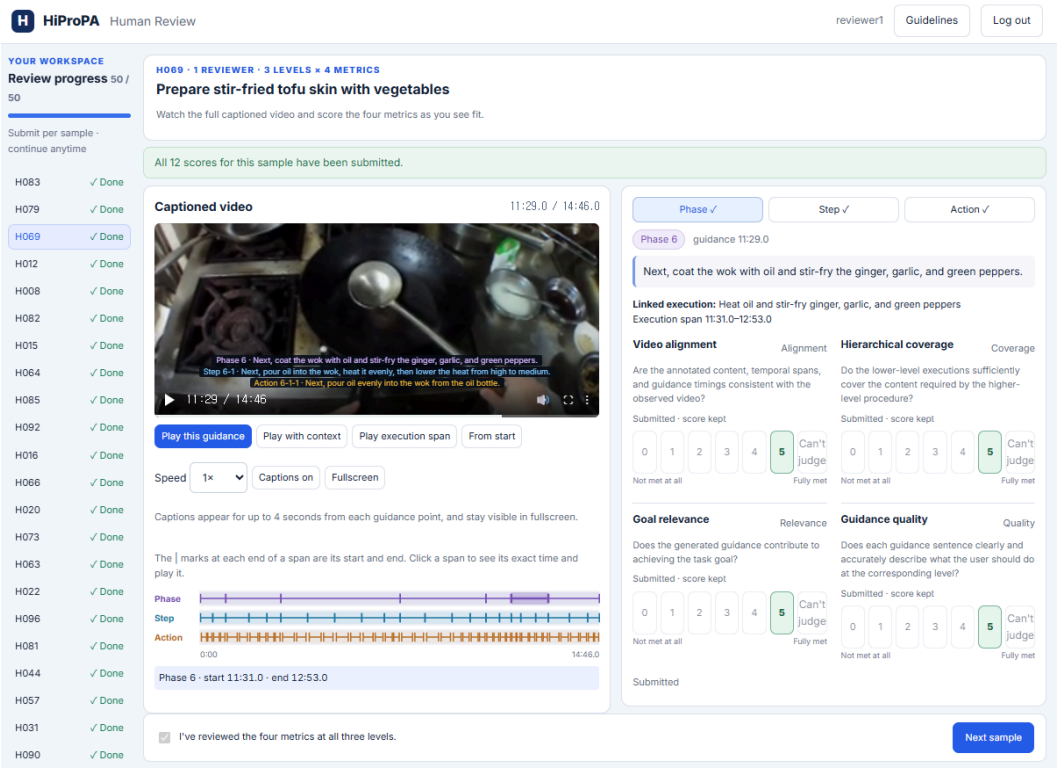


Figure 20: Human quality review interface. Reviewers inspect the task video and its hierarchical annotations at the phase, step, and action levels and evaluate them using four criteria: video alignment, hierarchical coverage, goal relevance, and guidance quality.

Video alignment evaluates whether the annotated content, temporal spans, and guidance timing are consistent with the observed video. Guidance may be provided before the next execution begins. However, guidance that is provided excessively early or late, or unnecessarily repeats a completed execution, is rated lower. The scores are 4.05, 4.14, and 3.80 for the phase, step, and action levels, respectively, showing that the annotations generally align well with the video progress.

Hierarchical coverage evaluates whether lower-level executions sufficiently cover the content required by the higher-level procedure. We examine whether the phases cover the activities needed to achieve the goal, the steps cover each phase, and the actions cover each step, focusing on whether any essential execution is missing. The scores are 4.35, 4.50, and 4.48, respectively, indicating strong coverage across all levels.

Goal relevance evaluates whether the selected executions and guidance contribute to achieving the task goal. Preparatory, safety-related, and recovery events that are necessary to achieve the task goal are considered relevant, whereas events unrelated to the goal are rated lower. The scores are 4.48, 4.41, and 4.39, respectively, showing high goal relevance across all levels.

Guidance quality evaluates whether each guidance sentence clearly and accurately describes what the user should do at the corresponding level. Lower scores are assigned when the guidance is unclear, difficult to follow, or does not match the intended level of detail. The scores are 4.37, 4.16, and 4.16, respectively, indicating that the generated guidance is generally clear and appropriate for each level.

E Details of Evaluation Protocol

E.1 Training Configurations

Table 8 summarizes the training settings. We keep the original task instructions. Video chunks and target outputs are ordered by time, with each output placed at its reference timestamp. We used sliding-window causal attention, restricting direct access to video tokens from the latest five chunks (10 seconds), including the current chunk, while retaining task instructions and prior output text within the context limit.

Setting	Shared configuration
Training hardware	8 NVIDIA B200 GPUs
Optimizer	AdamW
Learning rate	2×10^{-5}
Learning-rate schedule	Cosine decay; 5% warmup
Weight decay	0.1
Gradient clipping norm	1.0
Training epochs	3
Effective batch size	32 sequences
Maximum sequence length	98,304 tokens
Numerical precision	BF16
Distributed training	DeepSpeed ZeRO stage 1
Video sampling rate	1 fps
Video chunk	2 frames

Table 8: Shared SFT configurations.

E.2 Evaluation Prompts

Figure 21 shows the inputs shared by all baseline models: the task goal, the most recent 10 seconds of video, the ground-truth dialogue history, and the requested guidance level. Frontier models and multimodal baselines, including Qwen, use the shared prompt in Figure 22. Figure 23 shows the prompts for streaming and proactive assistance models, and Figure 24 shows the prompt for our model.

Common Inputs

Task Goal.
{task_goal}

Requested Guidance Level.
{Phase / Step / Action}
Phase: a high-level unit that achieves an intermediate goal through multiple steps.
Step: a procedural unit that achieves a specific outcome through multiple actions.
Action: an observable fine-grained event that can be guided independently.

Ground-Truth Dialogue History.
At {past_time}s: {past_GT_guidance}
Additional ground-truth guidance strictly preceding the current anchor, if any.
Past guidance describes what was said, not proof of completion.

Current Observation.
Current anchor: {time}s.
The most recent 10 seconds of video preceding the current anchor.

Figure 21: Common inputs for evaluation.

Frontier and Multimodal Baselines

You are a proactive assistant helping a user perform a procedural task. Use the current observation and ground-truth dialogue history to determine whether new guidance is needed at the requested guidance level. If guidance is needed, provide one concise instruction that helps achieve the task goal. Output one line: `$interrupt$` followed by the instruction, or `$silent$` when no new guidance is needed.

Based on the current observation and previous guidance, determine whether new guidance is needed at the requested guidance level. If needed, provide one concise instruction for the task.

Figure 22: Shared evaluation prompt for frontier models and multimodal baselines, including Qwen.

Streaming and Proactive Assistance Models

VideoLLM-Online and ProAssist.

You are a proactive assistant helping a user perform a procedural task. Use the current observation and previous guidance to determine whether new guidance is needed at the requested guidance level. Use the native streaming gate to remain silent when no new guidance is needed. When the gate selects a response, provide one concise instruction that helps achieve the task goal.

Based on the current observation and previous guidance, determine whether new guidance is needed at the requested guidance level. If needed, provide one concise instruction for the task.

VideoChat3-4B.

You are a proactive visual assistant helping a user perform a procedural task. Use the current observation and previous guidance to determine whether new guidance is needed at the requested guidance level. If guidance is needed, output `</Response>` followed by one concise instruction that helps achieve the task goal. If no useful new guidance is needed, output `</Silence>`.

Based on the current observation and previous guidance, determine whether new guidance is needed at the requested guidance level. If needed, provide one concise instruction for the task.

MMDuet2.

You are a proactive assistant helping a user perform a procedural task. Use the current observation and previous guidance to determine whether new guidance is needed at the requested guidance level.

Based on the current observation and previous guidance, determine whether new guidance is needed at the requested guidance level. If needed, provide one short imperative instruction for the task.

Figure 23: Prompts for streaming and proactive assistance models.

Our Multi-Level Model

You are a proactive procedural assistant watching a first-person video stream. Use the goal, observed frames, and prior outputs to decide whether to guide. Do not repeat a guide or guide an already finished unit.
Output exactly `<silent>` when no new guidance is needed. Otherwise output `<response>` followed only by the guidance entries needed now, separated by " / ":
`phase P: guidance / step P-S: guidance / action P-S-A: guidance`
Positive numbered IDs express parent–child relationships. Number units in temporal order within the clip and restart child numbers under each parent. Put entries in Phase, Step, Action order and omit inactive levels.

Figure 24: Evaluation prompt for our model.

F Details of Metrics

We provide detailed descriptions of sF1, gF1, and PQS. sF1 considers both the semantic alignment between generated and GT guidance and the coverage of GT guidance through temporal–semantic matching (Liu et al., 2026). gF1 considers both response F1 and silent F1 at anchor points. PQS

reflects the correctness of response–silence decisions and the quality of generated guidance (Kundu et al., 2026). All metrics are computed independently for each dataset and guidance level.

F.1 Soft F1 (sF1)

Following GuideMe (Liu et al., 2026), we partition each video at the midpoints $b_j = (\tau_j + \tau_{j+1})/2$ between consecutive GT response timestamps τ_j and τ_{j+1}. Within each segment, we perform minimum-cost one-to-one matching between predicted responses and GT anchors. For a predicted response $(\hat{y}_i, t_i)$ and a GT response (y_j, τ_j), the semantic similarity s_{ij} and matching cost c_{ij} are

$$s_{ij} = \left|\cos!\left(e(\hat{y}_i), e(y_j)\right)\right|, \qquad c_{ij} = (1 - s_{ij}) + \left[1 - \exp!\left(-0.01(t_i - \tau_j)^2\right)\right]. \tag{2}$$

Here, e is the `all-mpnet-base-v2` sentence encoder, and timestamps are measured in seconds. Since Silent Anchors contain no GT guidance, we set the semantic term $(1-s_{ij})$ to 1 when matching a predicted response to a Silent Anchor. The final matching set $\mathcal{M}$ includes only pairs in which both the prediction and GT are responses.

sPrecision (sP) measures how well the generated responses semantically match the GT responses, while sRecall (sR) measures how completely the GT responses are covered. Let P and R denote the numbers of predicted and GT responses, respectively, and C the sum of semantic similarities over matched pairs. We compute sPrecision and sRecall by dividing C by P and R, respectively, and sF1 as their harmonic mean:

$$C = \sum_{(i,j)\in\mathcal{M}} s_{ij}, \qquad \mathrm{sP} = \frac{C}{P}, \qquad \mathrm{sR} = \frac{C}{R}, \qquad \mathrm{sF1} = \frac{2,\mathrm{sP},\mathrm{sR}}{\mathrm{sP}+\mathrm{sR}} = \frac{2C}{P+R}. \tag{3}$$

We multiply sPrecision, sRecall, and sF1 by 100 when reporting results. Predicted responses without a matched GT response lower sPrecision, while GT responses without a matched prediction lower sRecall. The temporal cost affects the matching assignment but does not directly penalize the final sF1 score.

F.2 G-Mean F1 (gF1)

gF1 is computed from response–silence decisions at the original anchors. At a GT Response Anchor, a predicted response counts as TP and predicted silence as FN. At a GT Silent Anchor, predicted silence counts as TN and a predicted response as FP. Response F1 (RF1) treats response as the positive class and evaluates whether the model responds when guidance is needed. Silent F1 (SF1) treats silence as the positive class and evaluates whether the model remains silent when guidance is not needed. Following PWR (Kundu et al., 2026), we compute gF1 as the geometric mean of these two scores:

$$\mathrm{RF1} = \frac{2\mathrm{TP}}{2\mathrm{TP}+\mathrm{FP}+\mathrm{FN}}, \qquad \mathrm{SF1} = \frac{2\mathrm{TN}}{2\mathrm{TN}+\mathrm{FP}+\mathrm{FN}}, \qquad \mathrm{gF1} = 100\sqrt{\mathrm{RF1},\mathrm{SF1}}. \tag{4}$$

gF1 does not use temporal matching or assess response content; an inaccurate response at a GT Response Anchor still counts as TP.

F.3 Proactive Assistance Quality Score (PQS)

PQS jointly evaluates decision correctness and response quality at the original anchors. For each TP response, GPT-5.2 evaluates the generated guidance against the GT guidance using the evaluation prompt and scoring rubric from PWR (Kundu et al., 2026). The judge assigns ratings $r_{i,d} \in 1, \ldots, 5$ for relevance, specificity, actionability, and conciseness. We normalize the mean rating to a content score $g_i \in [0, 1]$ and compute PQS as

$$\bar{r}_i = \frac{1}{4}\sum_{d=1}^{4} r_{i,d}, \qquad g_i = \frac{\bar{r}_i - 1}{4}, \qquad \mathrm{PQS} = \frac{100}{N}\left(\mathrm{TN} + \sum_{i\in\mathcal{T}} g_i\right). \tag{5}$$

Here, $\mathcal{T}$ is the set of TP responses, and $N = \mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}$ is the total number of anchors. Each anchor receives a score of 1 for TN (correct silence), g_i for a TP response, and 0 for FP or FN. The final score is the average over all anchors, multiplied by 100. On ProactiveCoachBench, which contains equal numbers of Response and Silent Anchors, an always-silent predictor obtains a PQS of 50 but a gF1 of 0. We therefore interpret PQS alongside gF1.

G Additional Quantitative Results

As discussed in Section F, our evaluation metrics are computed from sPrecision, sRecall, Response F1, and Silence F1. Tables 9–12 report these four metrics in detail. Table 9 shows that our method achieves a better balance between Response F1 and Silence F1, together with substantial gains in sPrecision. Tables 10–11 further show the effects of hierarchical supervision and different supervision levels. Table 12 provides the results for zero-shot and fine-tuned multi-level guidance.

Model	Ours	ProactiveCoachBench												EgoProactive			
		Phase				Step				Action				Step			
		sPrecision	sRecall	Response F1	Silence F1	sPrecision	sRecall	Response F1	Silence F1	sPrecision	sRecall	Response F1	Silence F1	sPrecision	sRecall	Response F1	Silence F1
Proprietary Frontier Models																	
Gemini 3 Flash Preview		53.05	39.68	66.25	74.73	41.47	44.11	59.66	58.35	39.92	54.68	65.26	50.59	52.05	33.04	60.74	68.91
Claude Haiku 4.5		35.41	60.87	64.90	11.82	32.05	62.78	65.77	7.73	32.12	62.96	66.14	8.64	35.13	46.02	63.84	43.39
GPT-5.4 mini		33.06	38.50	55.81	44.84	33.41	36.17	54.25	48.33	37.06	35.83	53.94	53.80	39.05	43.05	61.03	47.08
Guidance Assistant Models																	
ProAssist-8B		21.55	8.22	22.33	67.63	28.19	6.28	17.14	66.33	37.18	6.28	14.48	66.17	19.18	3.83	9.64	63.02
Streaming Models																	
VideoLLM-Online		1.06	0.30	1.21	66.72	2.16	0.18	0.63	66.66	4.95	0.18	0.54	66.68	2.17	0.30	1.41	63.27
MMDuet2		15.11	8.73	27.08	60.30	26.16	19.90	42.45	54.85	27.98	19.02	43.16	56.20	32.04	33.28	52.56	38.03
VideoChat3-4B		29.93	49.55	60.62	6.03	27.80	50.95	63.34	10.37	29.08	57.83	65.61	4.94	34.59	48.71	61.12	21.59
Multimodal Models																	
VideoChat-R1.5-7B		0.09	0.04	0.09	66.67	0.66	0.04	0.23	66.54	1.41	0.06	0.17	66.29	0.62	0.12	0.34	65.02
InternVL3-8B		19.25	12.23	34.11	59.66	31.73	24.23	48.41	56.15	37.34	37.08	57.48	52.42	39.95	12.63	23.89	60.71
VideoLLaMA3-7B		19.75	15.40	37.38	52.87	28.13	20.77	43.33	54.99	33.90	27.95	48.30	51.93	2.88	0.73	1.36	61.27
LLaVA-Video-7B		23.34	20.20	43.18	52.79	33.51	32.17	53.36	50.00	39.21	22.15	44.54	59.67	41.00	34.89	48.50	46.82
Qwen3-VL-32B		34.84	56.47	64.77	22.89	29.86	57.66	65.45	8.72	29.88	59.25	66.23	6.55	44.01	28.20	44.73	55.47
Qwen3-VL-4B		29.37	50.65	63.88	10.43	28.41	54.59	65.45	10.34	29.04	56.56	66.05	11.68	35.86	48.62	59.59	21.63
	✓	73.95	44.26	68.88	80.38	59.13	44.45	68.04	75.30	46.69	56.37	68.22	61.34	47.66	31.86	60.00	65.69
Qwen3-VL-8B		32.56	59.98	66.44	1.37	29.24	59.49	66.47	1.38	28.62	58.92	66.60	1.21	32.94	59.47	70.02	0.09
	✓	74.08	45.87	71.33	81.41	58.70	46.40	69.05	75.12	46.55	57.65	68.05	60.41	50.38	29.57	58.78	67.65
Qwen3.5-4B		23.00	18.98	42.70	53.64	28.08	19.43	41.67	57.86	34.81	18.77	38.16	59.91	34.34	10.35	21.77	61.80
	✓	72.89	44.05	69.86	80.83	59.73	44.23	69.02	75.68	47.18	57.04	68.56	61.15	48.56	30.44	56.32	64.04
Qwen3.5-9B		32.14	58.17	65.30	2.08	30.19	56.64	64.38	11.24	31.69	61.09	64.98	11.10	29.68	51.08	63.18	10.57
	✓	72.37	41.48	66.52	79.65	61.57	41.72	66.92	75.94	49.19	54.43	67.77	64.61	48.43	31.75	58.98	65.52

Table 9: Detailed metrics for the main results in Table 2.

Supervision	ProactiveCoachBench Step				EgoProactive			
	sPrecision	sRecall	Response F1	Silence F1	sPrecision	sRecall	Response F1	Silence F1
Qwen3-VL-4B								
Step-only	40.72	48.23	66.87	59.99	50.61	27.95	55.10	66.30
Multi-level (Ours)	59.13	44.45	68.04	75.30	47.66	31.86	60.00	65.69
Qwen3-VL-8B								
Step-only	48.58	46.09	66.02	66.22	50.18	28.67	56.88	67.09
Multi-level (Ours)	58.70	46.40	69.05	75.12	50.38	29.57	58.78	67.65
Qwen3.5-4B								
Step-only	42.76	41.77	63.85	65.22	47.85	30.92	57.02	64.42
Multi-level (Ours)	59.73	44.23	69.02	75.68	48.56	30.44	56.32	64.04
Qwen3.5-9B								
Step-only	44.82	42.89	63.77	66.69	51.17	26.64	53.07	66.69
Multi-level (Ours)	61.57	41.72	66.92	75.94	48.43	31.75	58.98	65.52

Table 10: Detailed metrics for hierarchical vs. step-only supervision in Table 3.

Supervision	ProactiveCoachBench Step				EgoProactive			
	sPrecision	sRecall	Response F1	Silence F1	sPrecision	sRecall	Response F1	Silence F1
step-only	42.76	41.77	63.85	65.22	47.85	30.92	57.02	64.42
phase-step	48.32	51.12	67.89	64.59	48.29	29.26	55.16	64.39
step-action	56.57	41.09	66.56	75.32	45.60	35.52	61.19	62.32
phase-step-action	59.73	44.23	69.02	75.68	48.56	30.44	56.32	64.04

Table 11: Detailed metrics for the supervision-level ablation in Table 4.

FT	Phase				Step				Action			
	sPrecision	sRecall	Response F1	Silence F1	sPrecision	sRecall	Response F1	Silence F1	sPrecision	sRecall	Response F1	Silence F1
	15.06	6.76	15.17	69.69	14.71	6.05	13.22	64.65	29.21	59.74	63.38	1.94
✓	72.89	44.05	69.86	80.83	59.73	44.23	69.02	75.68	47.18	57.04	68.56	61.15

Table 12: Detailed metrics for zero-shot multi-level guidance in Table 5.

H Qualitative Results

Figures 25–30 present six qualitative examples from ProactiveCoachBench and EgoProactive. Compared with the baselines, our model more consistently provides appropriate guidance at response anchors while remaining silent at silent anchors. Its responses are also generally well aligned with the ground-truth guidance.



Figure 25: Qualitative comparison on ProactiveCoachBench-1 (goal: make a peanut butter and jelly tortilla roll).

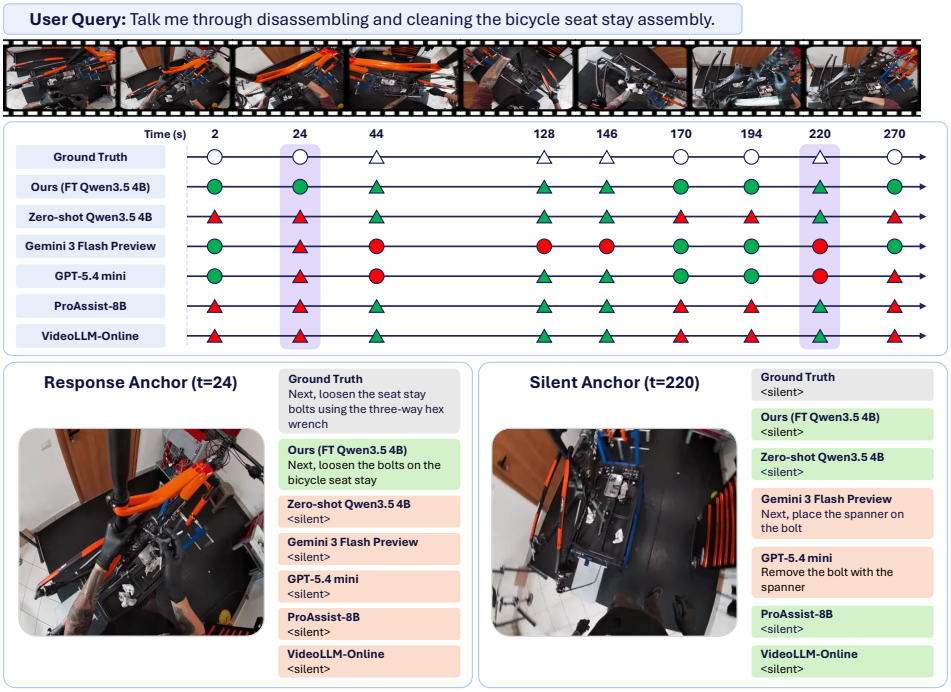


Figure 26: Qualitative comparison on ProactiveCoachBench-2 (goal: disassemble and clean a bicycle seat stay assembly).



Figure 27: Qualitative comparison on ProactiveCoachBench-3 (goal: remove dry sticks from a fence).



Figure 28: Qualitative comparison on ProactiveCoachBench-4 (goal: perform a COVID-19 rapid antigen test).

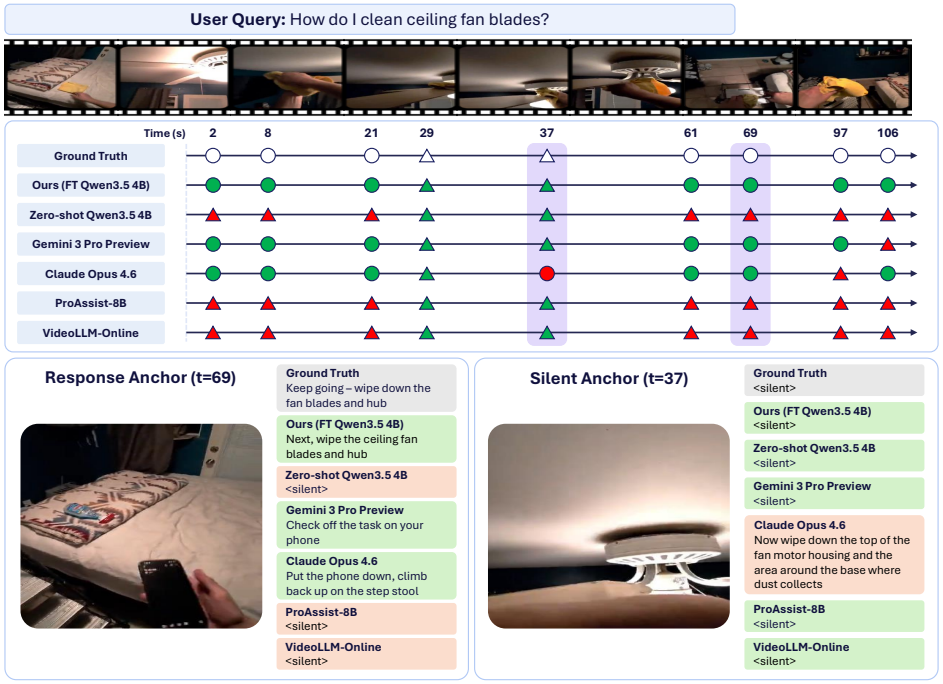


Figure 29: Qualitative comparison on EgoProactive-5 (goal: clean ceiling fan blades).

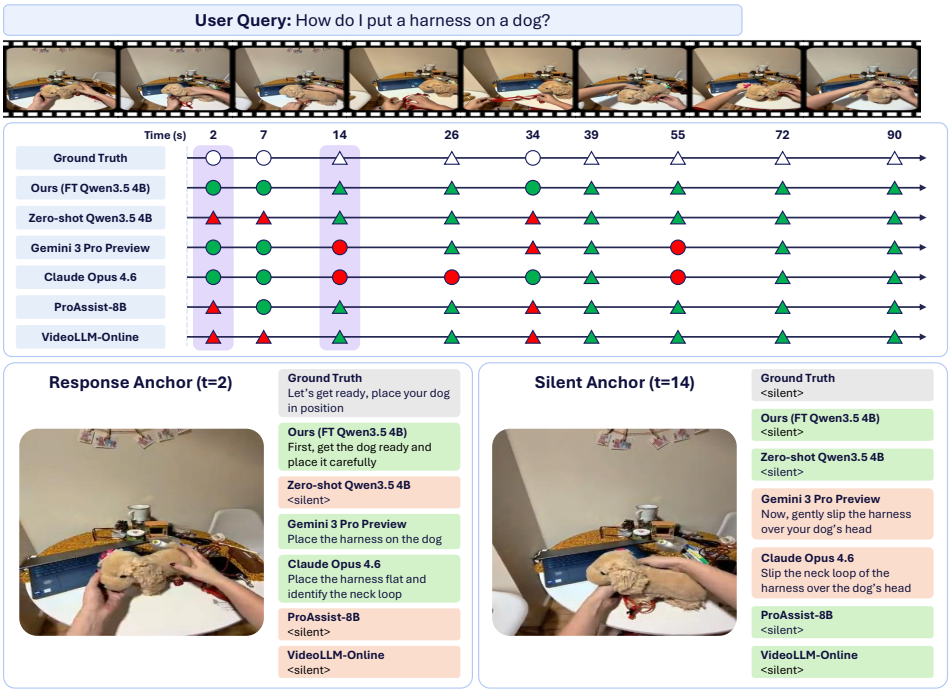


Figure 30: Qualitative comparison on EgoProactive-6 (goal: put a harness on a dog).

I LIMITATIONS

Our evaluation uses ground-truth dialogue history rather than model-generated outputs. When the model uses its own output history, further evaluation is needed to assess error accumulation and its ability to revise plans. In addition, guidance-level adaptation focuses on user-requested changes. Further research could explore how to infer the user's expertise and understanding to adjust the guidance level automatically, potentially making assistance more useful in real-world interactions. Also, since some activities can be performed in different orders to achieve the same goal, evaluation methods that consider these alternatives are needed. Future work should extend training and evaluation to support long-term task tracking and plan revision, personalized guidance, and multiple valid execution orders.