Appendix

In appendix, we present the following: limitations (Sec. A), additional quantitative results and analysis (Sec. B), additional qualitative analysis (Sec. C), additional implementation details (Sec. D).

A. Limitations

While AVP achieves strong performance and efficiency gains across multiple LVU benchmarks, it also has several practical limitations that point to promising future work rather than fundamental constraints.

First, we primarily evaluate AVP in the standard offline video QA setting, where the full video is available. An exciting direction for future work is to explore how the same active evidence-seeking framework operates in broader scenarios, such as embodied or online streaming environments where an agent must perceive and act in real time. Second, AVP currently uses prompting to drive planning and observation, learning policies that optimize long horizon sensing efficiency under resource and latency constraints (e.g., via reinforcement learning or differentiable planners) would be a complementary direction that builds on the same architecture.

B. Additional Quantitative Results and Analysis

B.1. Reasoning Trace Analysis

Proposed by MINERVA [34], the MiRA (MINERVA Reasoning Assessment) score is a reference-based, LLM-as-a-judge metric for evaluating the quality of multimodal models' step-by-step reasoning traces for video question answering. It assesses a model's generated reasoning against a ground-truth trace using the four axes of the MINERVA rubric: Perceptual Correctness, Temporal Localization, Logical Reasoning, and Completeness. T his normalized score helps analyze why models succeed or fail beyond just the final answer's accuracy, specifically highlighting weaknesses in video-centric aspects like temporal grounding and perception.

As shown in Tab. 6, AVP achieves the highest overall MiRA score, outperforming all baselines across key reasoning dimensions. Compared to single-pass MLLMs, AVP delivers substantially stronger temporal localization, logical reasoning, and correctness. These improvements indicate that actively collecting structured, query-conditioned evidence leads to higher-quality reasoning traces besides higher final accuracy. In particular, AVP's gains in temporal and completeness highlight the benefit of iterative planning and reflection for complex multi-hop queries.

Method	Acc. %	MiRA Score ↑				
		P	T	L	C	Total
OpenAI o1	43.5	0.52	0.52	*0.86*	*0.88*	0.69
GPT-4o	45.5	0.57	0.67	0.77	0.79	0.70
Gemini 2.0 Flash	53.5	**0.62**	*0.75*	0.83	0.82	*0.75*
Gemini 2.5 Pro	*61.8*	*0.60*	0.62	**0.97**	0.78	0.74
AVP (Ours)	**65.6**	**0.62**	**0.82**	**0.97**	**0.93**	**0.84**

Table 6. Reasoning trace quality check on MINERVA. We report multiple-choice accuracy and MiRA scores normalized to be between 0 and 1. P: Perceptual Correctness, T: Temporal Localization: L: Logical Reasoning: C: Completeness. The best result is in **bold**, and the second best is in *italic*.

Methods	ER	EU	KIR	TG	Rea	Sum	Overall
GPT-4o	48.9	49.5	48.1	40.9	50.3	50.0	48.9
OpenAI o3	57.6	56.4	62.9	46.8	50.8	67.2	57.1
AdaReTAKe	53.0	50.7	62.2	45.5	54.7	37.9	53.3
VideoTree	30.3	25.1	26.5	27.7	31.9	25.5	28.8
VideoAgent	28.0	30.3	28.0	29.3	28.0	36.4	29.3
VCA	43.7	40.7	37.8	38.0	46.2	27.3	41.3
MR. Video	59.8	57.4	71.4	58.8	57.7	50.0	60.8
DVD	**73.4**	*73.3*	**80.4**	*72.3*	**70.7**	*74.1*	*74.2*
AVP (Ours)	*71.9*	**76.7**	*80.1*	**73.6**	*67.7*	**75.9**	**74.8**

Table 7. **Results by question type on LVBench.** We report performance across six official LVBench splits: *Entity Recognition (ER)*, *Event Understanding (EU)*, *Key Information Retrieval (KIR)*, *Temporal Grounding (TG)*, *Reasoning (Rea)*, and *Summarization (Sum)*. Accuracy (%) is computed as Correct / Total for each split.

B.2. Full Results for LVBench

As shown in Tab. 7, AVP achieves the best overall accuracy on LVBench, outperforming all prior systems including the strongest agentic baseline DVD [82]. The gains are most pronounced on splits that require integrating information over long temporal ranges: AVP delivers the highest scores on Event Understanding, Temporal Grounding, and Summarization, indicating that its plan–observe–reflect loop is effective at steering perception toward query-relevant moments and aggregating evidence across distant segments. On Key Information Retrieval, Entity Recognition, and Reasoning, AVP remains competitive with DVD, while still substantially outperforming powerful generic MLLMs across all question types. These results suggest that explicit active video perception is crucial for long video understanding.

B.3. Additional Ablation Study

Controlled Backbone Comparison. In Tab. 1, we follow prior works (VideoTree, DVD) and report the best performance of each method. To further ensure a fair comparison, we extend same-backbone evaluations (w/ OpenAI-o3) to additional agentic methods on the Video-MME long split. As shown in Tab. 8, AVP achieves the highest overall accuracy and the lowest inference time among all compared

methods, demonstrating improvements in both effectiveness and efficiency under a controlled backbone setting.

	VideoTree	SiLVR	DVD	**AVP**
Long Split Acc. (%) ↑	61.2	66.8	67.3	**76.8**
Inference Time (s) ↓	145.4	442.2	612.8	**102.3**

Table 8. Controlled backbone comparison (OpenAI-o3) with agentic methods on Video-MME long split. We report long split accuracy (%) and average inference time (s).

Different Backbone MLLM Selection within AVP. As shown in Tab. 9, the performance of AVP on MINERVA scales steadily with the strength of the backbone MLLM. Using the lightweight Qwen3-VL-8B yields 41.2% accuracy (2.0% improvements compared to the direct inference), while swapping in stronger general-purpose models such as Gemini-2.5-Flash and OpenAI-o3 improves accuracy to 56.9% and 59.0%, respectively. The best results are obtained with Gemini-2.5-Pro (65.6%), indicating that richer reasoning and instruction-following capabilities at the backbone level directly translate into better planning, evidence selection, and reflection for complex multi-hop queries. At the same time, AVP delivers consistent gains across a wide spectrum of MLLMs, suggesting that our AVP framework is broadly applicable and can flexibly exploit future backbone improvements.

Structured vs. Unstructured Evidence List. As shown in Tab. 10, replacing our structured, time-aligned evidence list with an unstructured flat list degrades performance on both benchmarks, indicating that temporally and semantically organized evidence is crucial for effective planning and reflection.

Confidence Threshold Sensitivity Analysis. As shown in Tab. 11, a moderate confidence threshold yields the strongest results on MINERVA and ties for best performance on LVBench. Lower thresholds lead to premature halting and reduced accuracy, while overly strict thresholds offer no additional gains. This suggests that AVP benefits from a balanced stopping criterion, confident enough to avoid early termination, yet flexible enough to prevent unnecessary observation rounds.

Extended Component Ablation. To verify that each component's contribution generalizes beyond the Gemini backbone, we repeat the component ablation (Tab. 3) using OpenAI-o3 on MINERVA. Using only the Observer achieves 54.2% accuracy; adding the Planner yields a 3.2% improvement, and incorporating the Reflector provides a further 1.6% gain.

Backbone MLLM	**MINERVA (Acc. %)**
Qwen3-VL-8B	41.2
Gemini-2.5-Flash	56.9
OpenAI-o3	59.0
Gemini-2.5-Pro	**65.6**

Table 9. **Backbone MLLM selection within AVP.** The performance of AVP on MINERVA scales steadily with the strength of the backbone MLLM.

Evidence Format	**MINERVA**	**LVBench**
Unstructured List	63.2	71.2
Structured Evidence List (Ours)	**65.6**	**74.8**

Table 10. **Ablation on structured evidence list.** Replacing our structured, time-aligned evidence list with an unstructured flat list hurts performance on both benchmarks, showing that organizing evidence by temporal and semantic grounding is important for effective planning and reflection.

Confidence Threshold	**MINERVA**	**LVBench**
0.5	64.2	73.2
0.7	**65.6**	**74.8**
0.9	65.4	**74.8**

Table 11. **Ablation on confidence threshold.** We vary the confidence threshold for halting, observing that different values trade off answer conservativeness and coverage on both benchmarks.

These additive improvements are consistent with the Gemini-based results in Tab. 3, confirming that each AVP component contributes meaningfully regardless of the backbone MLLM.

B.4. Amortized Multi-Query Efficiency

One potential advantage of caption-based methods is that the video-level database can be reused across multiple queries for the same video, amortizing the construction cost. We evaluate this on LVBench under a multi-query setting (avg. 15 queries/video). DeepVideoDiscovery spends 637.2 s on database construction and an additional 2329.5 s on per-query localization and answering across all queries. By skipping the captioning stage entirely, AVP answers all queries in 2179.5 s, achieving a **26.5% reduction** in total inference time even in this setting favorable to database-reuse methods. This result confirms that the efficiency gains of active, query-driven perception hold not only per-query but also in amortized multi-query scenarios.

B.5. Token Efficiency Across Backbones

AVP's token efficiency arises from how it adaptively plans for temporal and spatial token usage, rather than backbone-specific visual compression. Under the same OpenAI-o3

backbone, AVP averages 62.5K input tokens per query on LVBench, only 5.8% of DVD's usage and even lower than the Gemini-based AVP (132.5K). This result demonstrates that AVP's efficiency does not rely on a specific backbone.

B.6. Robustness Across Video Domains

To assess whether AVP's improvements hold across diverse video domains, we break down MINERVA results by domain category. As shown in Tab. 12, AVP consistently outperforms Gemini-2.5-Pro across sports, instructional videos, and films, indicating that the gains from active perception are not specific to a particular video type.

	Sports	Instructional	Films
Gemini-2.5-Pro	57.5	57.2	75.2
AVP (ours)	**60.8**	**59.6**	**77.1**

Table 12. Domain-level accuracy (%) on MINERVA. AVP improves over its Gemini-2.5-Pro backbone across all video domains.

C. Additional Qualitative Results

C.1. Additional Visualization

As illustrated in Fig. 4, this example showcases how AVP leverages iterative planning to solve compositional, numerically precise queries that cannot be answered from a single view of the video. In the first round, the agent executes a narrowly targeted observation around the specified timestamp to read off the millimeter totals from the paper, but the reflector explicitly flags that the evidence is incomplete. The planner then revises its strategy, broadening the search space to a coarse scan over the entire video to hunt for the missing semantic attribute (the average hatchling length), which the observer recovers from narration. Only after both local numeric measurements and global semantic context are available does the reflector combine them into the final answer. This visualization shows AVP could tackle complex, multi-hop video reasoning via its iterative design.

C.2. Failure Case

In Fig. 5, we analyze a representative failure mode of AVP on a fine-grained counting query. To save computation, the planner opts for a coarse 0.5 FPS scan of the entire video and the observer only records two three-point plays before the second Hawaii–UCSB clip. Since the missing shot at 00:20 is never observed, the reflector receives a logically consistent but incomplete evidence list and confidently outputs the wrong answer. This case illustrates that, while our active perception pipeline is effective for locating dispersed, high-level evidence, it might make mistakes on questions that hinge on short, local events and subtle broadcast cues (e.g., bar graphics and rapid scoring plays). We further analyze 50 randomly sampled LVBench failures, categorizing errors into *planner*, *observer*, and *reflector*. Only 8% and 16% of errors stem from planning and reflection, while most arise from observer perception errors (76%). This verifies the AVP's planner and reflector are relatively robust.

D. Additional Implementation Detail

D.1. Prompts

We provided the planner prompt, the observer prompt, and the reflector prompt as follow.

Planner prompt (initial planning)

Function. `get_planning_prompt(query, video_meta, options)`.
Goal. You are an expert video analysis planner. Create a concise, observation plan to answer the user's query.

Inputs.
- **User Query:** `{full_query}`
- **Video Information:** duration in seconds (e.g., `Duration: {duration} seconds`)
- **Options (optional):** multiple-choice options attached to the query

Planning framework. Produce observation with:
- **What (Reasoning Objective):** what the step tries to accomplish.
- **Where:** temporal span to examine, either *uniform* (entire video) or a specific time range.
- **How:** `fps` and `spatial_token_rate`.

Timestamp handling. First classify the query:
- **Factual questions:** e.g., "what", "how many", "who", "which", "count", "identify".
- **Reasoning / explanation questions:** e.g., "why", "how", "explain", "reason", "cause".

Then apply:
- **Rule 1 (Exact ranges).**
  - Factual: use the *exact* range, no padding (e.g., "07:15–07:18" → `[435.0, 438.0]`).
  - Reasoning: add 15–30 s padding before and after (e.g., "07:15–07:18" → `[420.0, 453.0]`).
- **Rule 2 (Single timestamp).**
  - Factual: 1 s forward window from timestamp (e.g., "at 02:15" → `[135.0, 136.0]`).
  - Reasoning: add 15–30 s context (e.g., "at 02:15" → `[120.0, 150.0]`).
- **Rule 3 (Approximate / vague timing).** Use a ±15 s window around the mentioned time (e.g., "around 1:23" → `[68.0, 98.0]`).

Heuristics for unknown timing.
- "opening / beginning" → `[0, 30]`.
- "end / ending" → `[max(0, duration - 30), duration]`.
- No timing mentioned: use a coarse uniform scan with `fps` in 0.25–1.0 and low/medium resolution.

Step configuration guidelines.
- **Uniform scan (timing unknown).** `load_mode = "uniform"`, `fps` in 0.25–1.0, `spatial_token_rate` ∈ {`"low"`, `"medium"`}, `regions = []`.
- **Region analysis (explicit timestamps).** `load_mode = "region"`, `fps` ≈ 2.0, `spatial_token_rate` ∈ {`"low"`, `"medium"`}, `regions = [[start, end]]`.

Few-shot examples. [Few_examples]

Output format. Return a single JSON object with:
- `reasoning`: natural language explanation of your planning.

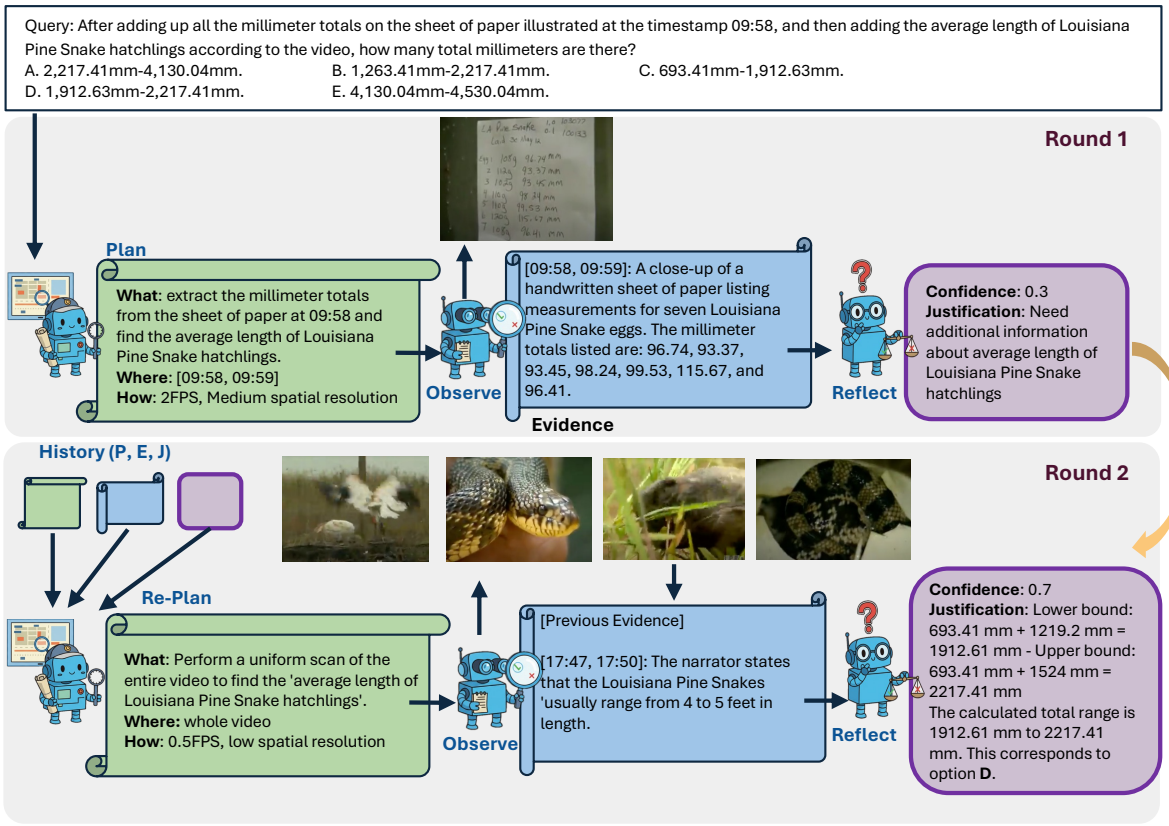

Figure 4. **Qualitative example of multi-round active perception in AVP (MINERVA sample).** Given the query, "After adding up all the millimeter totals on the sheet of paper illustrated at 09:58, and then adding the average length of Louisiana Pine Snake hatchlings according to the video, how many total millimeters are there?", AVP first plans to focus on the local timestamped frame at 09:58 and extracts the seven millimeter totals from the handwritten measurement sheet (Round 1). The reflector correctly judges that this evidence is insufficient because the average hatchling length is still unknown, and triggers a second round. In Round 2, the planner re-directs the observer to uniformly scan the full video at low FPS, locating a narrated segment that states hatchlings "usually range from 4 to 5 feet in length." By fusing the previous numeric evidence with this newly discovered range, the reflector computes the total millimeter interval and selects the correct option.

- `plans: what ∈ sub_query, where ∈ {"uniform", "region"}, how ∈ numeric fps (0.5–2.0), spatial_token_rate ∈ {"low", "medium"}`, and `regions` (list of `[start, end]` in seconds; empty for uniform).

Observer prompt (video inference / evidence extraction)

Goal. Analyze a specific video segment and extract precise, time-stamped evidence relevant to the user query.

Inputs.
- **sub_query:** the focused question for this round.
- **original_query:** the full user question (for multi-step agents).
- **context:** accumulated evidence from previous rounds.
- **start_sec, end_sec:** bounds of the segment to analyze.
- **video_duration_sec:** duration of the full video.
- **is_region:** whether this step analyzes a specified region or uniform scan.
- **regions:** list of `[start, end]` spans if multiple clips are provided.

Prompt structure.
- Primary task: describe visually relevant events in the analyzed video span.
- Provide:
  - **Detailed observations** tied to the query.
  - **Key timestamp ranges** (`timestamp_start, timestamp_end`) for each salient event.
  - **Reasoning** connecting observations to the sub-query.

Timestamp and evidence rules.
- Round timestamps to **integer seconds**: floor(start), ceil(end).
- List *all* relevant intervals for events that may match the query.
- Use context to avoid redundant descriptions.

Multiple-clip handling.
- When inputs include several regions, each corresponds to its absolute time span in the original video.
- You may reference clips descriptively (e.g., "Clip 1", "Clip 2").

Fallback rule (critical). If analyzing a *region* and no relevant information is present:
- Explicitly state: "No relevant information found in this time segment."
- Suggest expanding search to a uniform scan or additional regions.

Output format. Return a JSON object:

```
{
```

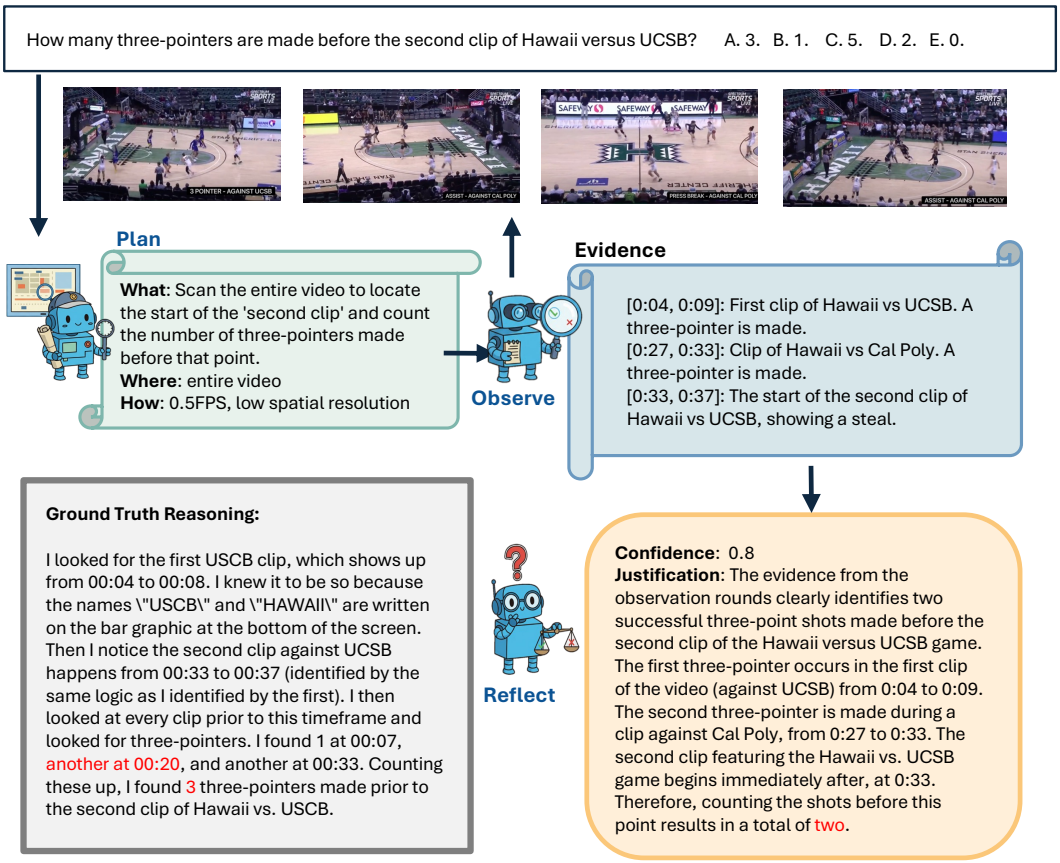

Figure 5. **Failure Case of AVP** (MINERVA sample). Given a long broadcast basketball video, AVP must answer: "How many three-pointers are made before the second clip of Hawaii versus UCSB?" The planner chooses to scan the entire video at 0.5 FPS with low spatial resolution, the observer summarizes the retrieved segments into a structured evidence list, and the reflector produces a confident answer of two. However, the ground-truth reasoning (yellow box) shows that a three-pointer at 00:20 is missed, so the correct count is three. Although the internal reasoning over the collected evidence is coherent, the initial coarse observation policy fails to capture a short, local event, leading to an overconfident but incorrect prediction.

```
  "detailed_response": "...",
  "key_evidence": [
    {
      "timestamp_start": <number>,
      "timestamp_end": <number>,
      "description": "..."
    }
  ],
  "reasoning": "..."
}
```

Example. [Few_examples]

Reflector prompt (evidence sufficiency checker)

Goal. Given the original query and cumulative evidence from all observation rounds, decide whether the current evidence is sufficient to answer the query, and produce a justification that either (i) contains the final answer, or (ii) explains what is missing.

Inputs.

- **query:** original user query (with options if MCQ).
- **evidence_summary:** aggregated evidence from all Observer steps.
- **video_duration:** total duration in seconds.
- **options:** optional list of MCQ options.

Your task.

- Decide a boolean `sufficient` indicating whether the evidence is enough to answer the query.
- **If sufficient (true):** the justification must give the *direct answer*.
  - MCQ: state the option letter (A/B/C/...) and a brief reason.
  - Open-ended: clearly state the answer in natural language.
- **If not sufficient (false):** the justification must explain what information is missing or uncertain (e.g., which regions, entities, or temporal spans require additional observation).
- Always provide a short `reasoning` paragraph that summarizes why the evidence is (not) sufficient.

Required JSON output (LLM response).

```
{
  "sufficient": <true | false>,
```

```
  "justification": "...",
  "reasoning": "..."
}
```

Few-shot examples. [Few_examples]