VIDEOAGENTBOOST: EVOLVING VIDEO AGENTS THROUGH ADDITIVE CAPABILITY IMPROVEMENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Changing video sources and question demands create a recurring need for video agents to evolve. Multimodal perception, tool use, and long-horizon reasoning make execution feedback costly to obtain, motivating targeted capability improvements from limited execution evidence. We introduce VIDEOAGENTBOOST, a framework for additive video-agent evolution guided by capability residuals: reusable descriptions of missing or insufficient behaviors identified by comparing execution trajectories with reference solutions grounded in annotated answers and supporting video evidence. Each round develops an increment for one residual, expanding or refining the relevant capabilities within the current agent. Targeted research and design guide code generation to implement the increment in the affected modules, including the coordination needed to use the improved capability. Before integration, targeted probing examines whether the increment contributes to resolving the selected deficiency through improved evidence acquisition or reasoning. Subsequent executions guide further improvements. We independently evolve DVD, VideoARM, and VideoAgent for five rounds on a shared evolution set. All three agents improve over their starting points on LVBench, LongVideoBench, and Video-MME. DVD gains 6.59 percentage points on LVBench over five evolution rounds, and component ablations further support the effectiveness of the proposed method. Our code and evolution-set annotations will be made publicly available.

1 INTRODUCTION

Video agents offer a flexible approach to video understanding by actively seeking and integrating evidence to answer complex questions. By combining language-model reasoning with retrieval and perception tools, they can locate relevant moments, inspect multimodal evidence, and revise their answers as new observations become available (Wang et al., 2024; Fan et al., 2024; Zhi et al., 2025). This approach supports tasks ranging from reconstructing action sequences to interpreting brief events and spoken details in long recordings. In practical applications, video content spans diverse activities, temporal structures, and audiovisual signals, while sources and user requirements continue to change. Meeting these evolving requirements calls for video agents that can continually develop their evidence-acquisition and reasoning capabilities.

Existing research advances video agents through the design of searchable representations, memory structures, retrieval strategies, and active perception mechanisms (Fan et al., 2024; Zhang et al., 2026b; Yin et al., 2026; Yang et al., 2025). These advances highlight the importance of video-agent design and raise a complementary question: how can execution feedback drive the continued improvement of video agents? Research on automated agent design explores feedback-driven changes to agent configurations, workflows, and code (Hu et al., 2025; Shang et al., 2025; Zhang et al., 2025). For video agents, this suggests using execution traces to identify missing capabilities and guide their development.

Video-agent evolution must account for the cost of obtaining this feedback. Executing and evaluating a candidate involves video decoding, multimodal perception, tool calls, and long-horizon reasoning, limiting the breadth of candidate evaluation within a practical budget. A failed answer alone offers limited guidance about what to improve: different failures may share a capability deficiency, while one question may expose several deficiencies. The challenge is therefore to identify

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: Motivation for evidence-bounded additive evolution. Changing video and question demands expose capability gaps (left), while costly multimodal execution motivates targeted use of execution feedback (middle). VIDEOAGENTBOOST targets a diagnosed capability residual and accumulates the resulting capability increment (right).

reusable capability deficiencies from the available trajectories and assess improvements through targeted execution. This motivates an additive approach that builds on accumulated capabilities while addressing one diagnosed deficiency at a time (Figure 1).

Boosting methods in machine learning (Freund & Schapire, 1997; Friedman, 2001) provide an important inspiration: they improve a predictor through additive corrections that target its remaining errors. Intuitively, this resembles uncovering and filling recurring blind spots in an agent's understanding. We draw on this principle to organize agent evolution around *capability residuals*, reusable descriptions of missing or insufficient capabilities grounded in execution evidence. Comparing actual trajectories with reference solutions constructed from annotated answers and supporting video evidence reveals where information acquisition or reasoning falls short. For example, sparse observations may repeatedly miss brief events across different questions. Abstracting these failures into a capability residual yields a reusable improvement target.

We introduce VIDEOAGENTBOOST, a framework that progressively improves a video agent one capability residual at a time. Each round diagnoses a target from the current execution evidence and develops a corresponding increment through targeted research, design, and code generation. *Capability expansion* introduces a missing behavior; *capability refinement* strengthens an existing one. Each increment includes the coordinated changes needed to use the improved capability. A focused Probe checks whether the intended capability is exercised and contributes to resolving the selected deficiency. Once verified, the increment is integrated into the agent, whose new executions refresh the evidence for the next round.

We evaluate VIDEOAGENTBOOST by independently evolving DVD, VideoARM, and VideoAgent on a shared evolution set and testing each agent on LVBench, LongVideoBench, and Video-MME. After five rounds, all three agents improve over their respective starting points on all three benchmarks, demonstrating the effectiveness of the same evolution procedure across different agent designs. A detailed study of DVD traces how successive capability increments improve evidence acquisition and reasoning, yielding a 6.59-percentage-point gain on LVBench. Trajectory case studies illustrate how capability improvements emerge and are verified, while component ablations support the contributions of Review and targeted Research.

Our main contributions are:

- We formulate capability residuals as reusable evolution targets that connect task-level execution failures to shared deficiencies in video evidence acquisition and reasoning.
- We develop VIDEOAGENTBOOST, an additive evolution framework that translates these residuals into coordinated capability improvements and verifies their contribution through targeted execution and trajectory analysis.
- We demonstrate consistent improvements across three video agents and three video-understanding benchmarks, with detailed analyses of capability accumulation and the contributions of individual framework components.

2 Related Work

Video understanding and video agents. Video understanding requires identifying relevant evidence in temporally extended, multimodal content. The evidence relevant to a question may be a sparse event, a dialogue or subtitle detail, a relation between distant moments, or a fine-grained visual property. Video agents make this evidence accessible through structured video representations and textual documents that support retrieval (Fan et al., 2024; Ma et al., 2025). VideoARM combines dynamic evidence acquisition with hierarchical multimodal memory to guide subsequent observations (Yin et al., 2026). Complementary approaches use language-model control, tool-based search, and adaptive inspection to decide where to look and what additional evidence to acquire (Wang et al., 2024; Zhang et al., 2026b; Zhi et al., 2025; Yang et al., 2025; Li et al., 2026). These mechanisms connect evidence organization with question-directed acquisition and reasoning, providing the capabilities on which video-agent evolution can build.

Automated agent evolution. Automated agent evolution uses execution feedback to improve configurations, workflows, and implementations. Meta Agent Search iteratively programs agents using an archive of evaluated designs (Hu et al., 2025), AgentSquare searches modular agent configurations (Shang et al., 2025), and AFlow optimizes executable workflows through search and execution feedback (Zhang et al., 2025). Gödel Agent and the Darwin Gödel Machine further investigate feedback-driven modification of agent implementations (Yin et al., 2025; Zhang et al., 2026a).

Boosting and incremental capability acquisition. AdaBoost and gradient boosting build predictors stage by stage, adding components directed at the error left by the accumulated predictor (Freund & Schapire, 1997; Friedman, 2001). Related work on progressive networks and local module composition explores incremental capacity and reusable components (Rusu et al., 2016; Ostapenko et al., 2021). These ideas motivate our capability-level approach to agent evolution: execution deficiencies serve as residual targets for increments that build on the agent's accumulated capabilities.

While existing video-agent research primarily develops manually designed architectures and tools, VideoAgentBoost introduces a meta-level algorithm that evolves video agents through execution feedback. Compared with prior work on automated agent evolution, our approach emphasizes capability-residual-guided additive improvements and targeted validation to accommodate costly multimodal execution, aligning the scope of each capability increment with the available validation evidence.

3 Problem Formulation

We evolve a video agent on a set $\mathcal{D}_{\text{evo}} = \{(v_i, q_i, y_i, e_i)\}_{i=1}^{N}$, where v_i is a video, q_i a question, y_i its annotated answer, and e_i the annotated supporting evidence intervals. At round t, the agent executes the video–question inputs $\mathcal{U}_{\text{evo}} = \{(v_i, q_i)\}_{i=1}^{N}$:

$$(\hat{y}_i^t, \tau_i^t) = \mathcal{A}_t(v_i, q_i), \tag{1}$$

where τ_i^t records its actions, observations, and reasoning state. Annotated answers and evidence intervals support evaluator-side analysis after execution. The objective is to improve the agent's capabilities from these executions and evaluate the evolved agent on independent video questions.

A reference solution g_i describes evidence sufficient to answer q_i, using the annotated answer and supporting intervals. Comparing it with the actual trajectory exposes a task-level divergence:

$$d_i^t = \text{Divergence}(g_i, \tau_i^t). \tag{2}$$

A *capability residual* is a reusable description of a missing or insufficient information-processing behavior. It specifies the deficient behavior, the execution evidence supporting it, the conditions under which it is needed, and an observable indication of improvement. Several tasks can support the same residual, while one task can expose several deficiencies. Review collects these divergences and their supporting evidence into records Rev_t and refreshes the pool

$$\mathcal{R}_t = \text{UpdatePool}\big(\mathcal{R}_{t-1}, \mathsf{Rev}_t\big), \qquad \mathcal{R}_{-1} = \varnothing. \tag{3}$$

New trajectories can revise the evidence for a residual, resolve it, or expose another deficiency.

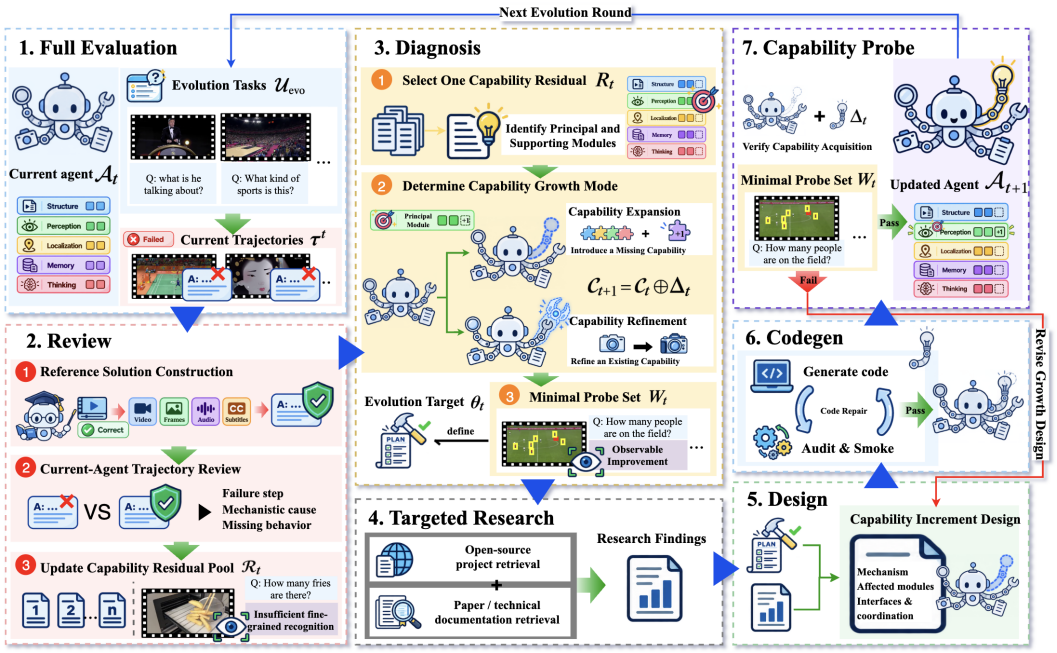

Figure 2: An evolution round in VIDEOAGENTBOOST. Review grounds task-level discrepancies in reference solutions built from answers and supporting evidence, and refreshes the residual pool. Diagnosis selects a capability target, its affected modules, and a small, targeted Probe set. Research and Design guide implementation, while source audit and Smoke support code repair. Probe evaluates the target capability: success admits the increment and leads to a new Full Evaluation; failure returns the same capability target to Design for revision.

The agent's cumulative capability state is organized through five semantic modules: Structure, Localization, Perception, Memory, and Thinking:

$$\mathcal{C}_t = \{\mathcal{C}_S^t, \mathcal{C}_L^t, \mathcal{C}_P^t, \mathcal{C}_M^t, \mathcal{C}_T^t\}, \qquad \mathcal{A}_t = \text{Compose}(\mathcal{C}_t). \tag{4}$$

Each $\mathcal{C}_m^t$ represents module m's current implementations of its accumulated capabilities. We use this decomposition to attribute a residual to a principal module and integrate the resulting capability increment into the current agent.

To make targeted use of costly video-agent execution feedback, we select one supported residual $R_t \in \mathcal{R}_t$ to focus the next increment Δ_t on behavior identified in the available trajectories. Capability expansion introduces a missing behavior; capability refinement strengthens an existing behavior under the diagnosed conditions. Each increment builds on the current capability implementations, extending their functionality or refining how they operate, together with the supporting module adjustments needed to use the improvement alongside accumulated capabilities. After validation, it is integrated through

$$\mathcal{C}_{t+1} = \mathcal{C}_t \oplus \Delta_t, \tag{5}$$

where $\oplus$ incorporates the verified capability change into the current agent. This stage-wise correction of remaining deficiencies is the boosting principle underlying VIDEOAGENTBOOST.

4 VIDEOAGENTBOOST

VIDEOAGENTBOOST turns task-level execution discrepancies into reusable capability improvements. Figure 2 shows the evolution process; Algorithm 1 summarizes one round. The central transformations are the abstraction of a task-level discrepancy into a capability residual, the design of a corresponding increment, and the verification of its contribution during execution. A multimodal teacher constructs reference solutions, while a coding agent performs trajectory analysis, capability diagnosis and development, and semantic Probe Audit.

Agent capability representation. We organize the initial agent into the five semantic modules in Equation 4, yielding $\mathcal{C}_0$. Structure prepares video-level representations and metadata for subsequent evidence acquisition; Localization identifies candidate evidence; Perception inspects the underlying audiovisual content; Memory maintains question-level context; and Thinking directs multi-step reasoning and tool use. These roles identify where a capability increment should be implemented within one executable agent. Thinking chooses Localization and Perception functions according to the question and accumulated observations. Appendix A.1 describes the module interactions.

From execution discrepancies to capability residuals. Full Evaluation executes all evolution tasks and records the current trajectories $\boldsymbol{\tau}^t = \{\tau_i^t\}_{i=1}^N$. A task-level discrepancy identifies a case for analysis, while its trajectory reveals the observations and decisions that produced it. To interpret the case, a multimodal teacher uses the annotated answer and supporting video intervals to construct a *reference solution*: a sufficient chain of audiovisual evidence and reasoning that explains the answer. This gives Review a reference for the information the agent needed to obtain and use.

Review compares the actual information flow with this reference. It examines where relevant evidence was missed, whether an observation captured the detail needed to answer the question, and whether the resulting information was retained and used in reasoning. For example, reaching a relevant interval but obtaining observations too sparse to resolve a brief event exposes a different deficiency from failing to locate that interval. The comparison concerns evidence sufficiency and use, allowing different effective action sequences. Review consolidates discrepancies that reflect a shared missing behavior across tasks, while distinguishing failures that require different capability changes. The resulting review records, denoted Rev_t, support the divergences and residual-pool update in Equations 2 and 3. As the agent changes, Review updates the pool using its new trajectories. Appendix B details reference solution construction and trajectory review.

Formally, the records passed to Diagnosis are

$$\mathsf{Rev}_t = \{\nu_i^t\}_{i=1}^N, \qquad \nu_i^t = \mathrm{Review}(g_i, \tau_i^t, e_i), \tag{6}$$

where each ν_i^t contains the observed divergence and the trajectory evidence supporting its interpretation.

Diagnosing an actionable evolution target. Diagnosis relates the review evidence to the current agent. It selects one supported residual from the refreshed pool and turns it into an actionable evolution target by determining its module attribution, growth mode, and Probe questions. It identifies where the agent's implementation limits the target behavior and specifies

$$\theta_t = (R_t, m_t^\star, \mathcal{M}_t^{\mathrm{assoc}}, z_t, W_t), \qquad z_t \in \{\mathrm{expansion}, \mathrm{refinement}\}, \tag{7}$$

where $m_t^\star$ is the principal module, $\mathcal{M}_t^{\mathrm{assoc}}$ contains supporting modules, and $W_t \subseteq \mathcal{U}_{\mathrm{evo}}$ is a small set of Probe questions.

The target connects a behavioral deficiency to an implementable change. In the case of a correctly located interval with insufficient visual detail, Perception may be the principal module; Thinking may also need to request the new inspection and use its observations. Diagnosis selects questions that expose this missing behavior and specifies what observable improvement would support its acquisition. A question can expose several deficiencies, so the Probe focuses on the selected residual and the contribution of its correction to solving the relevant questions.

Developing the capability increment. Expansion adds a capability whose behavior is absent from the current agent, such as adding the ability to acquire and use speech evidence when visual observations are insufficient. Refinement improves an existing capability under conditions identified by Review, such as adapting the temporal density of an available inspection routine to brief events. In both cases, the increment targets the diagnosed capability in the principal module and includes the supporting changes needed for other modules to request, retain, or use its output. The resulting behavior executes within the current agent.

The two forms of capability increment are summarized by

$$\Delta_t = \begin{cases} \mathrm{Expand}(m_t^\star, R_t), & z_t = \mathrm{expansion}, \\ \mathrm{Refine}(m_t^\star, \mathcal{C}_{m_t^\star}^t, R_t), & z_t = \mathrm{refinement}. \end{cases} \tag{8}$$

Algorithm 1 One round of capability-residual boosting

Require: Capability state $\mathcal{C}_t$, residual pool $\mathcal{R}_{t-1}$, evolution set $\mathcal{D}_{\text{evo}}$, and current trajectories $\boldsymbol{\tau}^t$
Ensure: Updated capability state, residual pool, and full-set trajectories

$\mathsf{Rev}_t \leftarrow \textsc{ReviewAll}(\mathcal{D}_{\text{evo}}, \boldsymbol{\tau}^t)$
$\mathcal{R}_t \leftarrow \textsc{UpdatePool}(\mathcal{R}_{t-1}, \mathsf{Rev}_t)$
$\theta_t \leftarrow \textsc{DiagnoseAndSelect}(\mathcal{R}_t, \mathsf{Rev}_t, \mathcal{C}_t)$
if $\theta_t \neq \varnothing$ **then**
  $\mathsf{Know}_t \leftarrow \textsc{TargetedResearch}(\theta_t, \mathcal{C}_t)$
  $\mathsf{Spec}_t \leftarrow \textsc{GrowthDesign}(\theta_t, \mathsf{Know}_t, \mathcal{C}_t)$
  $\Delta_t \leftarrow \textsc{DevelopAndProbe}(\mathcal{C}_t, \theta_t, \mathsf{Spec}_t)$
  if $\Delta_t \neq \varnothing$ **then**
    $\mathcal{C}_{t+1} \leftarrow \mathcal{C}_t \oplus \Delta_t$
    $\boldsymbol{\tau}^{t+1} \leftarrow \textsc{FullEval}(\text{Compose}(\mathcal{C}_{t+1}), \mathcal{U}_{\text{evo}})$
    return $(\mathcal{C}_{t+1}, \mathcal{R}_t, \boldsymbol{\tau}^{t+1})$
  end if
end if
return $(\mathcal{C}_t, \mathcal{R}_t, \boldsymbol{\tau}^t)$

The operators denote the implementation change specified by Design; supporting module adjustments are included in the same Δ_t.

Targeted Research examines relevant papers, open-source implementations, and technical documentation to identify mechanisms suited to the residual and the current implementation. Design combines these findings with θ_t to specify how the increment will work: the change in the principal module, the coordination with supporting modules, and the evidence or state that should appear in the resulting trajectory. The expected behavior links the implementation to the improvement criterion evaluated by Probe. Codegen implements the design; source audit and Smoke execution guide an internal repair loop. Appendix A.2 gives additional development details.

Verifying capability acquisition. Probe Run executes the candidate on W_t as ordinary video question answering. Semantic Probe Audit relates its answers and trajectories to the diagnosed deficiency. It checks whether the questions exercise the target condition, whether the intended mechanism operates through the affected module interactions, what relevant evidence or state it produces, and how that information contributes to the answer. This connects improvements on the selected questions to the behavior the increment was designed to provide. Appendix A.4 describes the evidence used in this assessment.

Let $\widetilde{\boldsymbol{\tau}}_{P,t}$ denote the candidate trajectories on the Probe questions. Probe Audit returns an admission judgment and revision feedback:

$$\begin{aligned} (p_t, f_t) &= \text{ProbeAudit}(\theta_t, \widetilde{\boldsymbol{\tau}}_{P,t}, \boldsymbol{\tau}^t, \mathsf{Rev}_t), \\ p_t = 1 &\implies \mathcal{C}_{t+1} = \mathcal{C}_t \oplus \Delta_t. \end{aligned} \tag{9}$$

Here p_t denotes the semantic judgment that the target behavior is present and usable in the affected interactions, and f_t provides feedback for further revision.

Probe feedback directs further Design and implementation for the same target when the intended capability remains insufficient. A successful Probe admits Δ_t through Equation 5. Full Evaluation then executes the updated agent on the entire evolution set, providing the new trajectories from which Review refreshes the residual pool. Appendix A.3 expands this multi-round procedure.

Here DEVELOPANDPROBE denotes the internal Codegen–Audit/Smoke–Probe revision loop for the selected target. An unsuccessful Probe returns the same target to Design and Codegen for revision; a successful Probe returns the verified increment for integration.

5 EXPERIMENTAL SETUP

We organize the experiments around three questions: (i) whether VIDEOAGENTBOOST improves different starting video agents across three video-understanding benchmarks; (ii) how capability increments accumulate over five rounds for one agent; (iii) how the major components of VIDEOAGENTBOOST contribute to evolution.

5.1 MODELS AND BENCHMARKS

We start from runnable implementations of DVD (Zhang et al., 2026b), VideoARM (Yin et al., 2026), and VideoAgent (Wang et al., 2024) and independently apply the same VIDEOAGENTBOOST procedure to each. All agent-evolution runs and component ablations use the common model and API assignments in Appendix D.1 whenever the corresponding capability is invoked, and all three runs use the same evolution set. The main comparison pairs every starting agent with its evolved counterpart. Appendix D.2 describes the initial agent implementations.

We evaluate the resulting agents on Video-MME (Fu et al., 2025), LongVideoBench (Wu et al., 2024), and LVBench (Wang et al., 2025). We follow the official evaluation protocol of each benchmark and evaluate each initial–evolved pair on the same questions with the same scoring procedure. We report accuracy for LVBench, LongVideoBench validation, and Video-MME.

5.2 EVOLUTION PROTOCOL AND EVALUATION SEPARATION

All three agents use a shared evolution set of 139 questions over 12 long videos, combining questions from CG-Bench (Chen et al., 2025) with manually constructed questions. The set covers diverse temporal, perceptual, and cross-segment reasoning demands and is separate from the evaluation instances of all three benchmarks. Each run contains five rounds. Full Evaluation covers the entire evolution set, and the resulting trajectories guide selection and validation of one capability increment per round. Annotated answers and supporting intervals are used for evaluator-side analysis. Appendix C describes the evolution set.

The held-out benchmark splits are not used to select residuals, develop candidates, or decide candidate admission. Final benchmark evaluation is performed after evolution. For the DVD round-wise analysis, we additionally evaluate its saved checkpoints on LVBench post hoc; these scores are used only for analysis and are not returned to the evolution loop.

5.3 COMPONENT ABLATIONS

We examine the individual and joint contributions of Review and Research. All variants start from DVD and use the same evolution set and model/tool configuration. We report the final agent's LVBench accuracy after five evolution rounds.

We compare the full method with three variants. In *w/o Review*, Diagnosis directly uses the preceding Full Evaluation results and execution records to identify residuals and select an evolution target, omitting reference-solution construction and per-task trajectory review. In *w/o Research*, Design proceeds directly from Diagnosis and the current agent, omitting targeted research into technical methods and implementation ideas for the selected capability. The third variant, *w/o Review & Research*, combines both changes.

A complementary case study in Appendix E.4 illustrates how Probe feedback guides candidate revision and verifies the resulting capability improvement.

6 RESULTS AND ANALYSIS

6.1 FINAL BENCHMARK PERFORMANCE

Table 1 compares three starting video agents with their independently evolved counterparts on LVBench, LongVideoBench, and Video-MME.

All three starting agents improve on all three benchmarks after evolution, including the long-video subsets of LongVideoBench and Video-MME. VideoAgent gains 12.46, 14.51, and 12.19 percentage points on LVBench, LongVideoBench, and Video-MME, respectively. These paired gains show that the same evolution procedure improves agents with different starting designs across the evaluated benchmarks. Appendix E.2 summarizes VideoAgent's five-round evolution. VideoARM's Video-MME gain spans short, medium, and long videos; Appendix E.1 provides the duration-wise results and its five-round evolution record. We next examine how the DVD result develops across rounds.

Table 1: Accuracy (%) on LVBench, LongVideoBench, and Video-MME. Each video-agent pair compares a starting agent with its counterpart independently evolved using VIDEOAGENTBOOST on the shared evolution set; "Long" denotes the long-video subset. Green arrows show percentage-point gains over the corresponding starting agent.

Method	LVBench	LongVideoBench (Val)		Video-MME	
	Overall	Overall	Long	Overall	Long
End-to-end video models					
GPT-4o (Hurst et al., 2024)	48.9	66.7	60.9	71.9	65.3
Gemini-2.5-Flash (Comanici et al., 2025)	56.7	66.2	61.8	74.2	69.1
Seed1.5-VL (Guo et al., 2025)	64.6	74.0	—	77.9	—
Qwen3-VL-235B-A22B (Bai et al., 2025)	67.7	—	—	79.2	—
Qwen3.5-Omni-Plus (Team, 2026)	71.2	—	—	81.9	—
Gemini-2.5-Pro (Comanici et al., 2025)	67.4	69.8	66.6	82.4	77.6
Video-agent evolution					
VideoAgent (Wang et al., 2024)	41.70	54.38	50.53	58.74	48.44
+ VIDEOAGENTBOOST	**54.16**$^{\uparrow 12.46}$	**68.89**$^{\uparrow 14.51}$	**66.67**$^{\uparrow 16.14}$	**70.93**$^{\uparrow 12.19}$	**63.56**$^{\uparrow 15.12}$
DVD (Zhang et al., 2026b)	74.56	73.30	71.63	79.19	72.44
+ VIDEOAGENTBOOST	**81.15**$^{\uparrow 6.59}$	**79.88**$^{\uparrow 6.58}$	**76.42**$^{\uparrow 4.79}$	**81.63**$^{\uparrow 2.44}$	**76.11**$^{\uparrow 3.67}$
VideoARM (Yin et al., 2026)	78.57	77.26	75.53	80.04	77.67
+ VIDEOAGENTBOOST	**84.25**$^{\uparrow 5.68}$	**81.82**$^{\uparrow 4.56}$	**78.55**$^{\uparrow 3.02}$	**83.59**$^{\uparrow 3.55}$	**80.67**$^{\uparrow 3.00}$

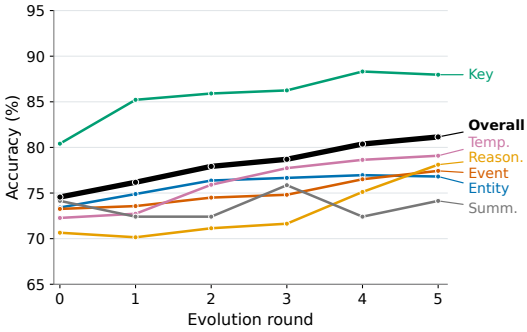

Figure 3: DVD checkpoint accuracy on LVBench across five capability increments. Six colored lines show category-wise accuracy and the thicker black line shows overall accuracy. Appendix Table 8 gives the exact accuracy values.

Table 2: LVBench accuracy (%) of the initial DVD and agents after five evolution rounds. Full denotes DVD evolved with VIDEOAGENTBOOST. The green arrow shows the gain over initial DVD; red arrows show drops from Full (percentage points).

Variant	Accuracy (%)
Full	**81.15**$^{\uparrow 6.59}$
w/o Research	78.37$^{\downarrow 2.78}$
w/o Review	76.89$^{\downarrow 4.26}$
w/o Review & Research	75.98$^{\downarrow 5.17}$
Initial DVD	74.56

6.2 CAPABILITY ACCUMULATION ACROSS EVOLUTION ROUNDS

Figure 3 shows DVD's overall and category-wise LVBench accuracy at the initial checkpoint and after each of five evolution rounds, with all saved checkpoints evaluated after completion of the five-round evolution. Appendix E provides the exact results, capability increments, and representative cases.

Across five rounds, DVD's LVBench score rises by 6.59 percentage points with a monotonic overall trajectory. The increments progressively address how the agent seeks and resolves evidence: Thinking first gains evidence-gap-aware control; focused Perception then supplies finer observations, followed by adaptive temporal Localization. The final two rounds add on-demand audio transcription and OCR to Perception and refine inspection of short intervals. This sequence develops complementary functions within the same agent. Improvements are uneven across question categories, with the largest gains in key information retrieval, reasoning, and temporal grounding. The improvements on LVBench follow evolution on the shared evolution tasks, supporting the usefulness of the accumulated capabilities on held-out questions.

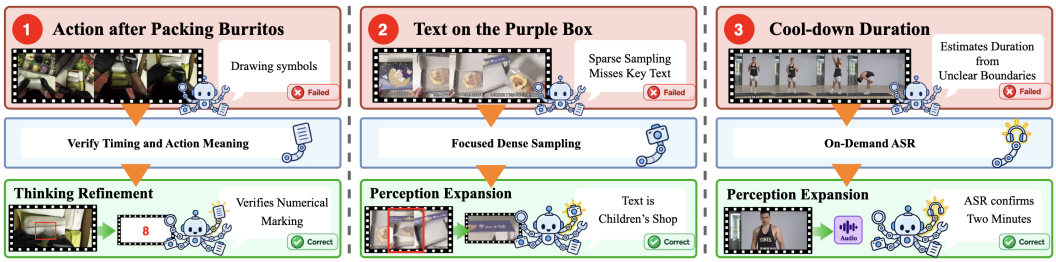

Figure 4: Examples of capability improvements during DVD evolution on the shared evolution set. Columns show cases from rounds 1, 2, and 4, respectively; rows show the preceding failure, the capability increment, and the resulting successful behavior.

Figure 4 illustrates how three increments change DVD's use of evidence. Thinking refinement helps distinguish numerical marking from drawing symbols by verifying the action and its temporal context. Focused inspection recovers text missed by sparse sampling, while on-demand ASR provides an explicit exercise duration when visual boundaries are ambiguous. These cases connect capability improvements to the evidence acquired and used in answering. Appendix E.4 provides detailed trajectory analyses.

6.3 Component Ablation

Table 2 compares the final LVBench accuracy of DVD evolved with the full method and the three ablation variants, assessing the individual and joint contributions of Review and Research.

Removing Review reduces final LVBench accuracy by 4.26 percentage points. This result supports the value of comparing reference solutions with the agent's execution trajectories when identifying capability deficiencies. Such comparisons provide evidence about where the agent's observations or reasoning fall short, helping Diagnosis select actionable evolution targets. Removing Research reduces accuracy by 2.78 points, indicating that targeted retrieval of technical methods and implementation ideas helps translate these targets into useful capability increments.

Removing both stages yields a 5.17-point decrease relative to the full method, yet still improves over the initial DVD agent. A plausible explanation is that Codex's existing knowledge and reasoning ability allow the remaining stages to infer useful changes directly from execution feedback. Explicit trajectory review and targeted research nevertheless yield stronger final performance.

7 Conclusion and Limitations

We presented VideoAgentBoost, a framework for improving video agents through capability-residual-guided additive evolution. It turns execution-grounded deficiencies into reusable capability increments and validates their contribution through targeted probing. Experiments with three different video agents show consistent improvements across three video-understanding benchmarks, while round-wise analysis and ablations illustrate the contributions of capability accumulation, trajectory review, and targeted research. These findings support residual-directed additive evolution as an effective way to develop video-agent capabilities from costly multimodal execution feedback.

Limitations. Our evaluation covers five evolution rounds per starting agent, leaving the longer-term gains and eventual limits of capability accumulation to be explored. With a larger capability repertoire, effective evolution also depends on the agent's ability to distinguish related tools and coordinate their use; how this ability scales with continued growth remains an open question. Multimodal execution remains costly, and residual diagnosis depends on the quality of reference solutions and observed trajectories. Although VideoAgentBoost focuses development and probing on a selected deficiency, further work could investigate how to obtain equally informative feedback with fewer video executions and use it to guide more effective capability improvements.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the VIDEOAGENTBOOST procedure, including module organization, capability-residual diagnosis, candidate development, and capability probing, in the main text and Appendix A. Reference-solution construction and trajectory review are described in Appendix B, the composition and annotation of the shared evolution set are documented in Appendix C, and the model and experimental configurations are provided in Appendix D. Representative evolution trajectories and capability-growth cases are reported in Appendix E. As supplementary material, we provide the shared evolution-set questions, annotated answers, and supporting evidence intervals. The supplement also includes presentation snapshots of the initial and evolved DVD agents and three recorded executions on LVBench test questions. The original videos are not redistributed. We plan to release the implementation of VIDEOAGENTBOOST following the review process.

AI USE STATEMENT

Except for the generative AI models that constitute the studied VIDEOAGENTBOOST system and are explicitly described as part of the methodology and experimental pipeline, generative AI was not used by the authors to design the research methodology or experiments, process the data, analyze or interpret the experimental results, conduct the literature review, or derive mathematical results.

ChatGPT was used only for language assistance in preparing the manuscript, including grammar, style, readability, and, where applicable, polishing or translating complete sentences provided by the authors. It was not used to determine the scientific content or claims of the paper. All AI-assisted language edits were reviewed and verified by the authors, who take full responsibility for the final manuscript.

Generative AI models used as components of VIDEOAGENTBOOST, including for trajectory review, diagnosis, capability development, code generation and audit, and Probe Audit, are part of the method under study and are explicitly documented in the paper.

REFERENCES

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.

Guo Chen, Yicheng Liu, Yifei Huang, Baoqi Pei, Jilan Xu, Yuping He, Tong Lu, Yali Wang, and Limin Wang. Cg-bench: Clue-grounded question answering benchmark for long video understanding. In *International Conference on Learning Representations*, volume 2025, pp. 45647–45682, 2025.

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.

Yue Fan, Xiaojian Ma, Rujie Wu, Yuntao Du, Jiaqi Li, Zhi Gao, and Qing Li. Videoagent: A memory-augmented multimodal agent for video understanding. In *European Conference on Computer Vision*, pp. 75–92. Springer, 2024.

Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pp. 1189–1232, 2001.

Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, et al. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24108–24118. IEEE, 2025.

Dong Guo, Faming Wu, Feida Zhu, Fuxing Leng, Guang Shi, Haobin Chen, Haoqi Fan, Jian Wang, Jianyu Jiang, Jiawei Wang, et al. Seed1. 5-vl technical report. *arXiv preprint arXiv:2505.07062*, 2025.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In *International Conference on Learning Representations*, volume 2025, pp. 21344–21377, 2025.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Keliang Li, Yansong Li, Hongze Shen, Mengdi Liu, Hong Chang, and Shiguang Shan. Lenswalk: Agentic video understanding by planning how you see in videos. *arXiv preprint arXiv:2603.24558*, 2026.

Ziyu Ma, Chenhui Gou, Hengcan Shi, Bin Sun, Shutao Li, Hamid Rezatofighi, and Jianfei Cai. Drvideo: Document retrieval based long video understanding. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18936–18946. IEEE, 2025.

OpenAI. Openai codex. `https://github.com/openai/codex`, 2025.

OpenAI. GPT-5.6 Sol Model. `https://developers.openai.com/api/docs/models/gpt-5.6-sol`, 2026.

Oleksiy Ostapenko, Pau Rodriguez, Massimo Caccia, and Laurent Charlin. Continual learning via local module composition. *Advances in Neural Information Processing Systems*, 34:30298–30312, 2021.

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

Yu Shang, Yu Li, Keyu Zhao, Likai Ma, Jiahe Liu, Fengli Xu, and Yong Li. Agentsquare: Automatic llm agent search in modular design space. In *International Conference on Learning Representations*, volume 2025, pp. 3841–3865, 2025.

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

Shihao Wang, Guo Chen, De-an Huang, Zhiqi Li, Minghan Li, Guilin Liu, Jan Kautz, Jose M Alvarez, Lei Zhang, and Zhiding Yu. Videoitg: Multimodal video understanding with instructed temporal grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24640–24650, 2026a.

Weihan Wang, Zehai He, Wenyi Hong, Yean Cheng, Xiaohan Zhang, Ji Qi, Ming Ding, Xiaotao Gu, Shiyu Huang, Bin Xu, et al. Lvbench: An extreme long video understanding benchmark. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22958–22967. IEEE, 2025.

Xiaohan Wang, Yuhui Zhang, Orr Zohar, and Serena Yeung-Levy. Videoagent: Long-form video understanding with large language model as agent. In *European Conference on Computer Vision*, pp. 58–76. Springer, 2024.

Ziyang Wang, Honglu Zhou, Shijie Wang, Junnan Li, Caiming Xiong, Silvio Savarese, Mohit Bansal, Michael S Ryoo, and Juan Carlos Niebles. Active video perception: Iterative evidence seeking for agentic long video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9088–9099, 2026b.

Haoning Wu, Dongxu Li, Bei Chen, and Junnan Li. Longvideobench: A benchmark for long-context interleaved video-language understanding. *Advances in Neural Information Processing Systems*, 37:28828–28857, 2024.

Zeyuan Yang, Delin Chen, Xueyang Yu, Maohao Shen, and Chuang Gan. Vca: Video curious agent for long video understanding. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 20168–20179. IEEE, 2025.

Xunjian Yin, Xinyi Wang, Liangming Pan, Li Lin, Xiaojun Wan, and William Yang Wang. Gödel agent: A self-referential agent framework for recursively self-improvement. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27890–27913, 2025.

Yufei Yin, Qianke Meng, Minghao Chen, Jiajun Ding, Zhenwei Shao, and Zhou Yu. Videoarm: Agentic reasoning over hierarchical memory for long-form video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24042–24051, 2026.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin gödel machine: open-ended evolution of self-improving agents. In *International Conference on Learning Representations*, volume 2026, pp. 104223–104294, 2026a.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, et al. Aflow: Automating agentic workflow generation. In *International Conference on Learning Representations*, volume 2025, pp. 34040–34077, 2025.

Xiaoyi Zhang, Zhaoyang Jia, Zongyu Guo, Jiahao Li, Bin Li, Houqiang Li, and Yan Lu. Deep video discovery: Agentic search with tool use for long-form video understanding. *Advances in Neural Information Processing Systems*, 38:89863–89895, 2026b.

Zhuo Zhi, Qiangqiang Wu, Wenbo Li, Yinchuan Li, Kun Shao, Kaiwen Zhou, et al. Videoagent2: Enhancing the llm-based agent system for long-form video understanding by uncertainty-aware cot. *arXiv preprint arXiv:2504.04471*, 2025.

A Additional Method Details

A.1 Module organization and execution

Initialization organizes a runnable agent into Structure, Localization, Perception, Memory, and Thinking under shared interfaces. Structure prepares reusable video-level state before question reasoning. Depending on the agent, this state can include video metadata, sampled frames, timestamped descriptions, audio transcripts, or retrieval indices linked to the source video and time. Localization uses the available representation to guide evidence acquisition, through indexed retrieval or exploration of the video.

During question answering, Thinking uses question-level Memory to choose Localization and Perception functions and interpret their outputs. Perception can inspect raw frames or audio within a selected interval; Memory retains the observations and reasoning context needed for subsequent decisions. An execution trajectory records these tool actions, observations, and state changes together with the final answer. This record allows Review and Probe Audit to follow how information reaches the answer.

A.2 Research and candidate development

Research retrieves relevant open-source implementations, papers, and technical documentation to identify mechanisms suitable for the selected residual. Design combines this material with the current implementation to specify the principal capability change, supporting module coordination, and expected execution behavior.

Codegen implements the specification. Source audit checks the implementation against the design, while Smoke executes representative inputs and exposes operational defects through their trajectories. The coding agent uses these observations to revise the candidate. Once runnable, the candidate undergoes Probe on the questions selected by Diagnosis. Probe feedback that identifies an insufficient capability informs another design and implementation attempt for the same residual.

Algorithm 2 Complete capability-residual evolution

Require: Initial capability state $\mathcal{C}_0$ and evolution set $\mathcal{D}_{\text{evo}}$
Ensure: Evolved agent $\mathcal{A}_t$ and its full-set trajectories $\boldsymbol{\tau}^t$

$t \leftarrow 0$
$\mathcal{R}_{-1} \leftarrow \varnothing$
$\mathcal{U}_{\text{evo}} \leftarrow \textsc{StripGold}(\mathcal{D}_{\text{evo}})$
$\boldsymbol{\tau}^0 \leftarrow \textsc{FullEval}(\text{Compose}(\mathcal{C}_0), \mathcal{U}_{\text{evo}})$
while evolution compute remains **do**
  $\mathsf{Rev}_t \leftarrow \textsc{ReviewAll}(\mathcal{D}_{\text{evo}}, \boldsymbol{\tau}^t)$
  $\mathcal{R}_t \leftarrow \textsc{UpdatePool}(\mathcal{R}_{t-1}, \mathsf{Rev}_t)$
  $\theta_t \leftarrow \textsc{DiagnoseAndSelect}(\mathcal{R}_t, \mathsf{Rev}_t, \mathcal{C}_t)$
  if $\theta_t = \varnothing$ **then**
    break
  end if
  $\mathsf{Know}_t \leftarrow \textsc{TargetedResearch}(\theta_t, \mathcal{C}_t)$
  $\mathsf{Spec}_t \leftarrow \textsc{GrowthDesign}(\theta_t, \mathsf{Know}_t, \mathcal{C}_t)$
  $\Delta_t \leftarrow \varnothing$; $p_t \leftarrow 0$
  while round development compute remains **do**
    $\Delta_t \leftarrow \textsc{CodegenAndAudit}(\mathcal{C}_t, \mathsf{Spec}_t, \Delta_t)$
    $\widetilde{\mathcal{A}}_t \leftarrow \textsc{AssembleCandidate}(\mathcal{C}_t, \Delta_t)$
    $\widetilde{\boldsymbol{\tau}}_{P,t} \leftarrow \textsc{ProbeRun}(\widetilde{\mathcal{A}}_t, W_t)$
    $(p_t, f_t) \leftarrow \textsc{ProbeAudit}(\theta_t, \widetilde{\boldsymbol{\tau}}_{P,t}, \boldsymbol{\tau}^t, \mathsf{Rev}_t)$
    if $p_t = 1$ **then**
      break
    end if
    $\mathsf{Spec}_t \leftarrow \textsc{ReviseDesign}(\mathsf{Spec}_t, \Delta_t, f_t)$
  end while
  if $p_t = 0$ **then**
    break
  end if
  $\mathcal{C}_{t+1} \leftarrow \mathcal{C}_t \oplus \Delta_t$
  $\boldsymbol{\tau}^{t+1} \leftarrow \textsc{FullEval}(\text{Compose}(\mathcal{C}_{t+1}), \mathcal{U}_{\text{evo}})$
  $t \leftarrow t + 1$
end while
return $(\text{Compose}(\mathcal{C}_t), \boldsymbol{\tau}^t)$

A.3 Complete evolution procedure

Algorithm 2 expands the multi-round process in Figure 2. Review refreshes the capability-residual pool from the current trajectories, and Diagnosis selects one target θ_t with its affected modules, growth mode, and Probe set W_t. The inner development loop below summarizes DevelopAndProbe. CodegenAndAudit implements or revises the candidate increment through code inspection, Smoke execution, and trajectory-guided repair. Probe then evaluates the target capability; its feedback guides further design revisions for the same target. In addition to the judgment p_t, Probe Audit returns feedback f_t for these revisions.

A.4 Evidence for Probe Audit

Probe Run records complete candidate trajectories on the selected questions without access to annotated answers or supporting intervals. After execution, Codex performs Probe Audit by comparing the candidate trajectory with the current-agent trajectory, the review records, the selected target, and the affected capability changes. The Probe set contains questions that expose the selected deficiency. The audit examines whether the intended behavior is exercised, produces relevant evidence or state, and contributes to answering the questions through the intended module interactions. Final-answer correctness is considered together with this mechanistic evidence. Feedback from an unsuccessful Probe returns to Design for the same target.

Table 3: Composition of the shared evolution set. Video durations are reported in minutes and seconds.

Statistic	Value
Unique videos	12
Questions	139
Selected CG-Bench questions	125
Manually constructed questions	14
Minimum / median / maximum duration	13:51 / 30:28 / 60:30
Supporting evidence intervals	288
Questions with multiple intervals	54

B Reference Solution Construction and Review

After the current agent answers the question, the multimodal teacher `qwen3.8-max` receives the question, annotated answer, and all supporting evidence intervals. When the evidence consists of disjoint intervals, the intervals are composed in their original temporal order while retaining a mapping between the composite evidence view and source timestamps.

The teacher observes the evidence media and constructs a reference solution that identifies the audiovisual facts sufficient to support the annotated answer and explains their relation.

Codex performs Review by comparing this reference solution with the current-agent trajectory, including the positions it inspected relative to the annotated intervals. Each review record contains the task-level divergence and its supporting execution evidence; together these records form Rev_t. Review compares the sufficiency and use of the acquired evidence, allowing alternative effective solution paths.

Reference solution construction during the five-round DVD evolution used approximately 47M teacher-model tokens on the shared evolution set.

C Shared Evolution Set

The shared evolution set combines questions selected from CG-Bench (Chen et al., 2025) with manually constructed questions on the same videos. Table 3 summarizes its composition. The additional questions use seven of the twelve videos and broaden coverage of repeated-event disambiguation, ordinal event localization, non-speech sound retrieval, narrative structure, event-boundary localization, long-range entity matching, and cross-segment reasoning. All three starting agents use this set independently; its videos and questions are disjoint from the three evaluation benchmarks.

Video content and question demands. The videos cover everyday activities, sports and fitness, software operation, instructional material, film and television, practical demonstrations, and health and safety content. Questions and answer options are in English; speech and on-screen text are predominantly Chinese and English, with one primarily Korean program. Questions require fine-grained recognition of brief events, repeated-event counting, temporal relations, cross-segment identity matching and multi-hop reasoning, speech and on-screen text interpretation, and global narrative understanding. For example, a software operation question counts four clicks supported by four separate intervals; another question matches a person across three distant intervals; and a gardening-video question draws on six intervals to summarize the storyline. These demands combine localized evidence acquisition with reasoning over temporally dispersed observations.

Answers and supporting evidence. The selected CG-Bench questions retain their original questions, options, answers, and evidence annotations. For the additional questions, answers, options, and supporting intervals were manually specified during construction. Every question has at least one supporting interval: 85 use a single interval and 54 use multiple intervals. Multi-interval evidence supports tasks such as event counting, temporal relations, identity matching, and summarization. Annotation checks verified that answers matched the available options and that evidence

Table 4: Common API and evolution-model assignments used in the experiments.

Role	Model	Function
Thinking	`qwen3.7-max`	Question analysis, tool selection, evidence synthesis, and answering.
Video structuring and visual perception	`qwen3.7-plus`	Video representation construction and localized visual inspection.
ASR	`qwen3-asr-flash`	Audio transcription.
OCR	`qwen-vl-ocr-2025-11-20`	On-screen text recognition.
Embedding	`text-embedding-v4`	Text-embedding retrieval over video descriptions.
Reference-solution teacher	`qwen3.8-max`	Reference-solution construction from annotated answers, supporting video evidence, and ASR transcripts.
Trajectory review	Codex (GPT-5.6 Sol)	Comparison of agent trajectories with reference solutions to produce Review findings.
Diagnosis	Codex (GPT-5.6 Sol)	Analysis of Review findings and selection of a capability residual.
Evolution development	Codex (GPT-5.6 Sol)	Research, Design, Codegen, and source audit.
Probe Audit	Codex (GPT-5.6 Sol)	Assessment of target-capability acquisition from candidate answers and trajectories.

intervals fell within video duration. The annotated answers and intervals support evaluator-side reference solution construction and trajectory review.

D Experimental Configurations

Sources of external model results. The GPT-4o and Gemini-2.5 results in Table 1 are taken from the comparison in Active Video Perception (Wang et al., 2026b). Seed1.5-VL, Qwen3-VL, and Qwen3.5-Omni results are taken from their respective technical reports (Guo et al., 2025; Bai et al., 2025; Team, 2026); the Qwen3-VL results correspond to the 235B-A22B-Instruct variant.

D.1 Model configurations

All agent-evolution runs and component ablations use the common assignments in Table 4 whenever the corresponding capability is invoked. Codex (OpenAI, 2025) uses GPT-5.6 Sol (OpenAI, 2026) with high reasoning effort for evolution development, Diagnosis, trajectory review, and Probe Audit. The reference-solution teacher receives the supporting video evidence together with audio transcripts produced by the ASR model.

D.2 Initial agent implementations

The three starting agents retain their core evidence-acquisition and reasoning mechanisms within the five-module organization.

VideoAgent. In the VideoAgent-based implementation (Wang et al., 2024), Structure samples video frames at 1 FPS and indexes their visual features for retrieval. Question answering begins with observations of five uniformly sampled frames, an initial answer, and a three-level confidence assessment. When more evidence is needed, Thinking describes the desired visual content and selects a local segment bounded by adjacent sampled frames. Localization retrieves the most relevant frame within that segment using visual features, and Perception obtains a new visual observation. Memory retains frame positions, observations, the current answer, and confidence assessments. Thinking updates its answer and confidence after each observation, with at most two retrieval–observation rounds before the final answer.

DVD. The DVD-based implementation (Zhang et al., 2026b) organizes video evidence at multiple granularities. Structure samples frames at 2 FPS and constructs approximately ten-second segments with precomputed transcripts, timestamped visual descriptions, entity information, and a semantic

Table 5: VideoARM evolution: accuracy on the shared evolution set, capabilities, and primary modules. The Module column lists the primary module improved in each round; coordinated changes to other modules are omitted.

Round	Acc.	Capability	Module
0	52.52%	Initial VideoARM with sparse scene observation, interval-level visual analysis, audio transcription, question memory, and an iterative controller.	-
1	55.40%	Failure-aware evidence recovery that records unsuccessful calls and adjusts intervals, parameters, modalities, or tools.	Memory
2	57.55%	Timestamped coarse localization that converts sparse observations into local summaries and candidate intervals for focused inspection.	Localization
3	61.15%	Question-conditioned evidence grounding that checks candidate evidence against the events, entities, and premises required by the question.	Thinking
4	61.15%	Event disambiguation over temporal summaries to separate repeated or adjacent events and rank candidate positions.	Localization
5	65.47%	Comparative multi-clip perception that jointly examines localized intervals while preserving candidate identity and temporal order.	Perception

Table 6: VideoARM accuracy on the Video-MME test set by video duration.

Duration	Initial agent	Final evolved agent
Short	83.11%	87.33%
Medium	79.33%	82.78%
Long	77.67%	80.67%
Overall	80.04%	83.59%

index. Segment-level entity information is consolidated into a video-wide entity registry. During question answering, Thinking selects among a text-based global overview, semantic segment retrieval, and visual inspection of identified intervals. Localization provides relevant segments and overview evidence from the structured representation, while Perception examines frames from one or more selected intervals. Memory retains the question-level reasoning, tool calls, and observations, allowing Thinking to refine its search and combine evidence across segments before answering.

VideoARM. The VideoARM-based implementation (Yin et al., 2026) combines dynamic evidence acquisition with hierarchical multimodal memory. Structure records video metadata, including duration, frame rate, and audio availability. Thinking selects observations according to the question and accumulated evidence: Localization supports broad exploration through sparse, temporally indexed frames, and Perception supplies detailed visual analysis or audio transcription for selected intervals. Memory organizes the resulting evidence into scene summaries, audio transcripts, and local visual analyses within each question. Thinking uses the updated memory to select further observations or produce the final answer.

E Evolution Trajectories and Case Studies

E.1 VideoARM evolution and benchmark evaluation

Starting from VideoARM, VideoAgentBoost develops five capability increments on the shared evolution set. Table 5 reports this run's evolution-set trajectory.

The evolved VideoARM improves over its initial state in every Video-MME duration group (Table 6). Overall benchmark accuracy rises from 80.04% to 83.59% after evolution on the shared evolution set.

E.2 VideoAgent evolution and benchmark evaluation

Starting from the VideoAgent implementation described in Appendix D.2, the first evolution round introduces ten-second clip descriptions and text embeddings for description-based semantic re-

Table 7: VideoAgent evolution: accuracy on the shared evolution set, capabilities, and primary modules. The Module column lists the primary module improved in each round; coordinated changes to other modules are omitted.

Round	Acc.	Capability	Module
0	44.60%	Five-frame initial observation, answer-confidence assessment, visual-feature-based frame retrieval within a temporal segment, and visual re-inspection.	-
1	48.20%	Ten-second clip descriptions and text embeddings introduce description-based semantic retrieval; clips retrieved within a segment are inspected through grouped frame sampling.	Structure
2	51.08%	Global top-k semantic clip retrieval adds candidates beyond the segments bounded by initially sampled frames and carries those candidates into Memory and Thinking.	Localization
3	54.68%	Timestamped OCR and interval-restricted ASR provide on-screen text and speech evidence to reasoning.	Perception
4	61.87%	Multi-query retrieval locates temporally dispersed candidate clips; Perception samples frames across the retrieved intervals and preserves their temporal order.	Localization
5	64.03%	Thinking and Memory retain prior actions, observations, and reasoning context across steps, allowing the model to choose the next tool action or answer from the evolving question-level state.	Thinking

Table 8: Round-wise LVBench results. The six category columns report accuracy for entity recognition (Entity), event understanding (Event), key information retrieval (Key), temporal grounding (Temp.), reasoning (Reason.), and summarization (Summ.).

Round	Entity	Event	Key	Temp.	Reason.	Summ.	Overall
0	73.41%	73.26%	80.41%	72.27%	70.65%	74.14%	74.56%
1	74.89%	73.57%	85.22%	72.73%	70.15%	72.41%	76.18%
2	76.37%	74.50%	85.91%	75.91%	71.14%	72.41%	77.92%
3	76.66%	74.81%	86.25%	77.73%	71.64%	75.86%	78.70%
4	76.96%	76.51%	88.32%	78.64%	75.12%	72.41%	80.37%
5	76.81%	77.43%	87.97%	79.09%	78.11%	74.14%	81.15%

trieval. Table 7 summarizes the capabilities accumulated over five rounds on the shared evolution set.

The accumulated capabilities improve VideoAgent on all three benchmarks (Table 1). The long-video subsets show gains of 16.14 percentage points on LongVideoBench and 15.12 points on Video-MME.

E.3 DVD EVOLUTION AND LVBENCH EVALUATION

Table 8 gives the exact LVBench accuracies of DVD's saved checkpoints plotted in Figure 3.

Table 9 summarizes the capability increment admitted in each DVD evolution round, together with its accuracy on the shared evolution set and primary module.

DVD's evolution-set accuracy increases from 66.19% at the initial checkpoint to 82.73% after five rounds. Together, Tables 8 and 9 connect the benchmark checkpoint curves to the capabilities accumulated during evolution.

E.4 CAPABILITY-GROWTH CASES

The following cases illustrate capability development and candidate revision in the DVD run on the shared evolution set. A worked round connects diagnosis, research, implementation, and validation; a Probe-guided revision case shows how execution feedback improves a candidate before admission. Three task-level cases illustrate how the increments from the first, second, and fourth

Table 9: DVD evolution: accuracy on the shared evolution set, capabilities, and primary modules. The Module column lists the primary module improved in each round; coordinated changes to other modules are omitted.

Round	Acc.	Capability	Module
0	66.19%	Initial DVD agent with video structuring, semantic retrieval, global browsing, and frame inspection.	-
1	74.10%	Evidence-gap-aware Thinking control: call tools when supporting evidence is absent, reconcile conflicting observations, and seek time-aligned confirmation before deciding.	Thinking
2	75.54%	Focused frame inspection with richer evidence and retrieval metadata for fine-grained visual analysis.	Perception
3	76.98%	Adaptive temporal localization for event relations and short-lived evidence, coupled with focused visual verification inside candidate intervals.	Localization
4	80.58%	On-demand ASR and OCR perception with multimodal evidence carried into question-level Memory.	Perception
5	82.73%	Task-adaptive sampling within localized short intervals, with strengthened timestamp and interval-boundary handling.	Perception

rounds improve evidence verification, fine-grained visual observation, and speech-based evidence acquisition, respectively.

Worked evolution round. DVD's second evolution round addressed insufficient visual verification of retrieved evidence. Figure 5 summarizes the reasoning behind this increment and shows a condensed implementation of the resulting tool. Across tasks, semantically related clips could miss the relevant moment, while ordinary sparse inspection left transient actions, text, and counts unresolved. Diagnosis targeted Perception, with supporting changes to Localization and Thinking. Research identified question-conditioned fine-grained observation and interactive evidence acquisition as useful implementation ideas. VideoITG's instruction-conditioned grounding and frame selection (Wang et al., 2026a), together with the memory-augmented VideoAgent of Fan et al. (2024) and its interactive use of perception tools, informed a design that coupled candidate retrieval with targeted visual inspection and further search when the evidence remained insufficient.

The implementation added a focused-inspection tool alongside ordinary frame inspection, enabling denser sampling of the original video within retrieved intervals. Localization exposed retrieval scores, candidate ranks, and score gaps; Thinking used these signals and subsequent observations to decide whether to inspect a candidate more closely, reformulate the query, or seek another evidence source. The increment thus strengthened visual verification while providing retrieval-relevance signals to guide further reasoning.

Probe executed five selected questions, with three answered correctly. The trajectories showed how the new behavior contributed to evidence acquisition. On an exercise-counting question, an inconsistent count from ordinary inspection prompted focused observation, which described alternating knee raises against successive countdown readings and led to comparison with another exercise interval. On a software-operation question, the agent inspected several red warning dialogs and compared their Chinese text to identify the one referring to the publishing tool, yielding the correct answer. Probe showed that the candidate changed its observation granularity or evidence source and used the resulting information to guide further reasoning, supporting acceptance of the increment. The focused-inspection capability became part of the accumulated agent and was further refined in the fifth round.

Probe-guided revision case. DVD's fifth evolution round refined an existing focused-inspection tool to support denser, task-dependent sampling within localized short intervals. The first Probe showed that the candidate instead used the tool to scan broad temporal ranges, repeatedly splitting long requests into smaller windows while maintaining a largely fixed sampling rate. Probe Audit identified a mismatch between this behavior and the intended fine-grained observation capability. Revision coordinated Perception and Thinking: focused inspection operated on short localized intervals, while Thinking selected sampling density according to the temporal detail needed and used the returned sampling information to assess its observations.



Figure 5: Development of the Focused Frame Inspection tool in DVD's second evolution round. The upper panels summarize Review, Diagnosis, Research, and Design; the lower panel presents a condensed implementation for a single localized interval, using 4 FPS sampling to obtain fine-grained visual evidence and return the observation with sampling metadata.

The second Probe demonstrated the revised behavior on an exercise-interval question. An earlier broad observation had mistaken part of a jogging sequence for mountain climbers. Dense inspection of the candidate interval identified jogging throughout, correcting that interpretation. Thinking then combined the observation with countdown readings from retrieved clips to infer the full jogging interval and select the correct answer. Tool use also became more selective: a separate task requiring broader inspection used ordinary frame observation. Probe Audit accepted the refined capability based on its appropriate use, adaptive sampling, and contribution to subsequent reasoning. This case illustrates how Probe feedback can improve both an existing perceptual tool and the reasoning that directs its use.

Temporal and semantic verification through Thinking refinement. DVD's first evolution round strengthened evidence verification before answering. One question asks what the protagonist does first after packing six burritos. The initial agent had observed a pen being used to write on the wrapping paper, but selected the distractor describing drawing symbols rather than the correct option describing numerical marking. The observation alone had not led to sufficient verification of the action against the question and answer options.

The evolved agent first inspected an earlier candidate interval, then continued searching to verify the action after all six burritos had been packed. It checked the writing on the white wrappers and identified numerical marks, distinguishing the observed action from the distractor describing drawing symbols. This verification supported the correct choice of making numerical marks with a pen. The case illustrates how Thinking can improve an answer by checking temporal relevance and the precise meaning of the observed action.

Fine-grained visual verification through focused inspection. DVD's second evolution round added focused frame inspection to obtain visual details missed by sparse sampling. A question uses a subtitle about buying clothes for a baby to identify a scene and asks what is written on a purple box. The preceding agent inspected the relevant interval, but its sampled frames missed the view containing the answer text.

After ordinary inspection, the evolved agent invoked the new focused tool to observe the box more densely within the selected interval. The resulting visual observation identified "Children's Shop" on the box, supplying the text needed to answer the question. This case illustrates how denser local sampling can recover decisive evidence that is absent from an otherwise relevant set of observations.

Duration verification through on-demand speech perception. DVD's fourth evolution round expanded Perception with on-demand ASR and OCR. The speech capability is illustrated by a question asking how long the final stretching and cool-down exercise lasts. The preceding agent attempted to estimate the duration from the visual sequence, but ambiguous action boundaries prevented a precise determination.

The evolved agent browsed the video and inspected the ending segment, but visual inspection left the exercise boundaries and duration uncertain. Thinking then requested an ASR transcription of the relevant speech, which explicitly stated, "Next, we will do two minutes of stretching and cooling down." The final reasoning used this statement to select two minutes. The added speech-perception tool thus supplied direct duration evidence where visual boundary estimation was insufficient.

Together, these three task-level cases illustrate more precise evidence verification, denser visual observation, and access to complementary speech evidence. Each addresses a distinct deficiency within the accumulated five-module agent.