# VisualClaw: A Real-Time, Personalized Agent for the Physical World

**Anonymous authors**
Paper under double-blind review

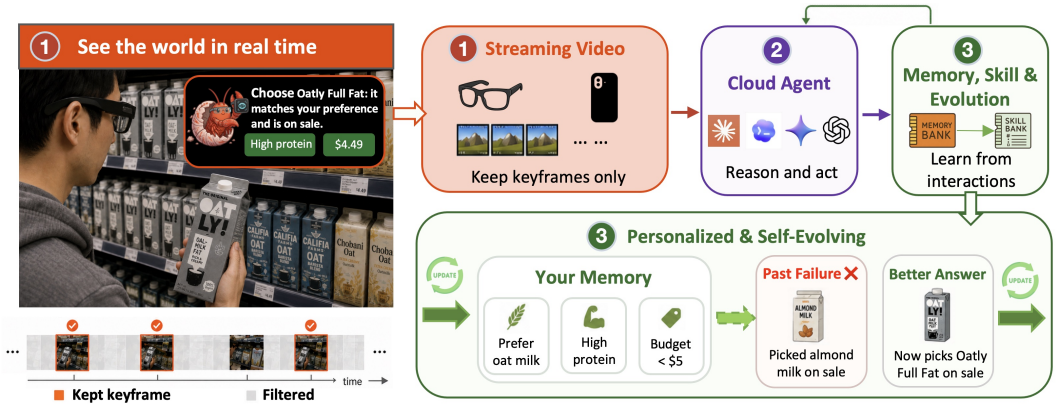


Figure 1: VisualClaw efficiently encode streaming video in real scenarios with a cascaded edge gate and produces personalized answers and actions. Correctness feedback and retrieved memories guide skill-bank evolution, allowing later rounds to reuse experience across multimodal agent tasks.

## Abstract

Vision language models (VLMs) are serving as general-purpose interfaces for complex multimodal tasks. However, deployment still faces three gaps: VLMs typically incur high latency and cost when processing dense video frames and long prompts, the agent scaffold remains static after deployment, and standard video-QA benchmarks do not test whether agents can use visual evidence inside tool-using workspaces. We present **VisualClaw**, a self-evolving multimodal agent built around two principles. First, *hybrid encoding* reduces deployment cost by filtering less informative streaming frames with a cascaded gate and compressing the text skill bank through hot/cold top-$k$ injection. Second, *skill evolution* lets the agent learn from failures: retrieved memories condition an offline evolver either as direct concatenated context or as guided evidence, producing skill-bank updates that help future questions. Across 4 video-QA benchmarks with 2 VLMs (Gemini 3 Flash and GPT-5.2), VisualClaw reduces API cost by $98.1\%$ versus full-frame upload and by $25.9\%$ versus offline Uniform-8 under the same skill bank. Under the streaming settings, our VisualClaw improves over the baseline by $+3.80\%$ on average, while reaches a peak $+15.80\%$ boost on EgoSchema with Gemini 3 Flash. To address the benchmark gap, we further curate **VisualClawArena**, a 200-scenario multimodal agentic benchmark built through a strict five-stage pipeline; models must use video evidence, documents, dynamic updates, and executable checks inside a workspace. On VisualClawArena, the same self-evolution framework with computer-use agent backends improves macro accuracy by $+2.9\%$ for Codex (GPT-5.5) and $+3.2\%$ for Claude Code (Sonnet 4.6) over no-evolution baselines, with a $-8.7\%$ cost reduction compared to the uniform-sampled baseline. These properties make VisualClaw a natural fit for streaming edge applications such as AI glasses, where the cascade filters continuous video locally and transmits only selected keyframes at interaction, while self-evolution supports adaptive assistance.



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.

# 1 INTRODUCTION

Multimodal models are becoming the default interface for agents that combine perception, language, memory, and tool use. Vision-language models (VLMs) are a concrete subset: with full visual information and long prompts, frontier VLMs already perform well on standard video question-answering (video-QA) multiple-choice (MC) benchmarks (Mangalam et al., 2023; Fu et al., 2024). However, this offline setup hides three deployment gaps: the full multimodal context is assumed to be available at query time, the model scaffold is assumed to stay fixed, and the evaluation is usually limited to static video-QA rather than workspace-style multimodal agents. Real video-centric agents, from cloud assistants to wearable AI glasses, break all three. Visual observations arrive as streams of unknown length; the agent should improve from its own failures; and practical tasks require inspecting files, reconciling video evidence with text records, editing workspace artifacts, and passing executable checks. VisionClaw (Liu et al., 2026) makes this setting concrete by combining Meta's AI glasses, live multimodal perception, and OpenClaw execution, but it leaves self-evolving skill/memory banks, prompt-cost growth, edge-side frame gating, and benchmarked agentic evaluation largely untested.

These gaps have so far been addressed mostly in isolation. On the *efficiency* side, frame-selection systems (Zhang et al., 2024; Song et al., 2024; He et al., 2024; Wang et al., 2024b) compress or subsample the input video, but they usually assume offline access to the full clip and do not address prompt-side cost from a growing skill bank. On the *adaptation* side, skill libraries for large language model (LLM) agents (Wang et al., 2024a; Zhao et al., 2024; Tang et al., 2025; Xia et al., 2026) distill reusable behavioural rules from past failures and inject them at inference, demonstrating that frontier-model accuracy can be lifted *without weight updates*. However, these systems remain text-only; in multimodal settings, a growing bank of skills and memories can become a new token bottleneck. On the *evaluation* side, standard video-QA benchmarks test one-shot answering, so they barely measure whether an agent can use visual evidence while conducting real actions and satisfying automatic checks.

We present VisualClaw, a self-evolving multimodal agent built around two principles. (i) *Hybrid encoding* for efficient edge deployment. The video stream is filtered by a lightweight cascade: perceptual hash, 128-dim CPU encoder, and adaptive change gate, so only salient video frames are sent to the VLM API. The skill bank is also encoded efficiently: a small hot top-$k$ set is injected with full skill text, while the remaining skills are exposed as a compact cold catalogue. (ii) *Self-evolving* skills and memories for adaptive agents. Correctly-solved examples are stored in the memory bank, and failures trigger an offline LLM as the evolver to use relevant memories and failures to update the skill bank. This failure-driven loop supports session-level adaptation without updating VLM weights. We show a running example of what VisualClaw is capable of in Figure 1.

To address the third gap, we curate VisualClawArena, a 200-scenario multimodal agentic benchmark for evaluating VisualClaw under tool-using execution with multiple-choice questions and executable-check problems. Starting from various existing video data (Yang et al., 2025; Mangalam et al., 2023; Lei et al., 2021), we propose data pipeline to build documents, chat/audio traces, dynamic updates, and executable checkers, and filter scenarios with a five-stage curation workflow: candidate generation, timestamp-grounded workspace construction, paired text-only/with-clip leakage filtering, multimodal criteria-based selection, and health checks. The final suite contains an average of $24.4$ steps and $18.1$ visual-required steps per scenario.

For static video-QA, we evaluate $4$ benchmarks and 2 VLM families: two egocentric benchmarks (EgoSchema, EgoPlan-Bench) and two general-video benchmarks (Video-MME long, NextQA), tested on Gemini 3 Flash and GPT-5.2. Our proposed streaming evolution configuration (`FullEvo`) improves over the baseline by $+3.80\%$ on average and by $+15.40\%$ on EgoSchema with Gemini 3 Flash; the no-memory evolving variant reaches $+15.80\%$ on the same benchmark. When the same evolution strategy is added to an offline Uniform-8 baseline, it also contributes an average $7.8\%$ accuracy boost across two benchmarks. On efficiency, hybrid encoding reduces aggregate API cost by $98.1\%$ against full-frame upload and by $25.9\%$ against offline Uniform-8 with the same evolved bank. The natural cascade exchanges some offline accuracy for causal operation and fewer uploads; at matched $K{=}8$, cascade-fill isolates selection quality and beats Uniform-8 on both short-clip benchmarks. We further evaluate VisualClaw on VisualClawArena with two tool-using agent backends, Codex (GPT-5.5) and Claude Code (Sonnet 4.6), over 200 VisualClawArena scenarios. VisualClaw with simple memory-to-evolver concatenation reaches $54.3\%$ macro accuracy with Codex and $52.2\%$

Table 1: Selected-frame VLMs. Selector is the frame selection method. Online indicates if the method keep/skip decision frame-by-frame as frames arrive. Edge-CPU: selector runs on-device with no GPU. And existing multimodal agents are unable to evolve natively.

| Method | Selector | Online | Edge-CPU | Evolve |
|---|---|---|---|---|
| LLoVi (Zhang et al., 2024) | uniform chunks | ✗ | ✗ | ✗ |
| MovieChat (Song et al., 2024) | sliding window | ✗ | ✗ | ✗ |
| MA-LMM (He et al., 2024) | trained mem-Q | ✗ | ✗ | ✗ |
| VideoAgent (Wang et al., 2024b) | LLM planner | ✗ | ✗ | ✗ |
| Frame-Voyager (Yu et al., 2025) | LLM planner | ✗ | ✗ | ✗ |
| SeViLA (Yu et al., 2023) | BLIP-2 locator | ✗ | ✗ | ✗ |
| VideoTree (Wang et al., 2025) | LLM tree-descent | ✗ | ✗ | ✗ |
| TimeChat (Ren et al., 2024) | uniform sampling | ✗ | ✗ | ✗ |
| VTimeLLM (Huang et al., 2024) | uniform sampling | ✗ | ✗ | ✗ |
| Video-LLaMA (Zhang et al., 2023) | uniform sampling | ✗ | ✗ | ✗ |
| LLaVA-Video (Zhang et al., 2025b) | uniform sampling | ✗ | ✗ | ✗ |
| VisionClaw (Liu et al., 2026) | n/a | ✓ | ✓ | ✗ |
| **VisualClaw** | **cascade (heuristic)** | ✓ | ✓ | ✓ |

with Claude Code, improving over matched no-evolution Cascade-8 baselines by $+2.9\%$ and $+3.2\%$ and over Uniform-8 by $+4.0\%$ and $+8.2\%$. The lift is strongest on empirically hard scenarios ($+5.4\%$ Codex, $+5.3\%$ Claude Code), and the Claude Code Cascade-8 setting costs $-8.7\%$ less than the Uniform-8 method. Together, these results show that the evolving skill/memory scaffold improves same-selector controls across both task types, while the causal frame gate reduces uploads and remains compatible with frozen VLM backbones.

## 2 Related Work

**Skill-based and memory-augmented agents.** A line of work augments agents with reusable skill libraries or external memory to improve performance without modifying model weights. Reflexion (Shinn et al., 2023) stores verbal self-reflections in the agent memory buffer; Voyager (Wang et al., 2024a) incrementally builds a library of executable code skills from successful episodes; ExpeL (Zhao et al., 2024) and Agent-KB (Tang et al., 2025) distill cross-task experience into natural-language rules. Memory systems include MemGPT (Packer et al., 2023), Generative Agents (Park et al., 2023), Mem0 (Chhikara et al., 2025), and MemEvolve (Zhang et al., 2025a). A shared limitation is that the skill library is treated as a static artifact, not coordinated with weight optimisation. MetaClaw (Xia et al., 2026) addresses this by coupling skill evolution with RL training; There are also recent computer use agents that leverage memory systems for GUI tasks (Han et al., 2026; Pointer, 2026). Our contribution is their multimodal system design: causal edge selection, feedback-driven skill evolution, and agentic evaluation, while the model weights remain untouched.

**Selected-frame and efficient video VLMs.** A growing line of work selects a small subset of frames before invoking the VLM, but the prevailing approach puts an LLM *in the per-frame selection loop* or assumes offline access to the full clip. LM-planner / locator selectors (Wang et al., 2024b; Yu et al., 2025; 2023; Wang et al., 2025) score candidate frames against the question via an inner LLM call; memory-augmented compressors (Song et al., 2024; He et al., 2024) use sliding windows or learned memory queries to fold long video into a fixed budget; uniform / chunked stacks (Zhang et al., 2024; Ren et al., 2024; Huang et al., 2024; Yang et al., 2023; Zhang et al., 2023; 2025b) use fixed frame budgets after clip access, optionally prefixing time tokens for temporal grounding. VisionClaw (Liu et al., 2026) shares our always-on wearable motivation with OpenClaw execution. Table 1 records properties demonstrated by each cited system; within this comparison, VisualClaw explicitly demonstrates causal frame-by-frame selection on CPU without LLM calls. Orthogonally, VisualClaw also evolves the skills and memories across the task stream.

## 3 VisualClaw: An Efficient Multimodal Agent that Evolves Itself

VisualClaw combines three modules at different timescales: an edge cascade $G$ selects frames as they arrive, a hot/cold injector retrieves from skill bank $S$ for each question, and a memory-augmented evolver updates $S$ from failures and the memory $M_v$ per session. The cascade reduces visual cost, while retrieval and evolution adapt the language scaffold without updating model weights $\theta$ (Figure 2).

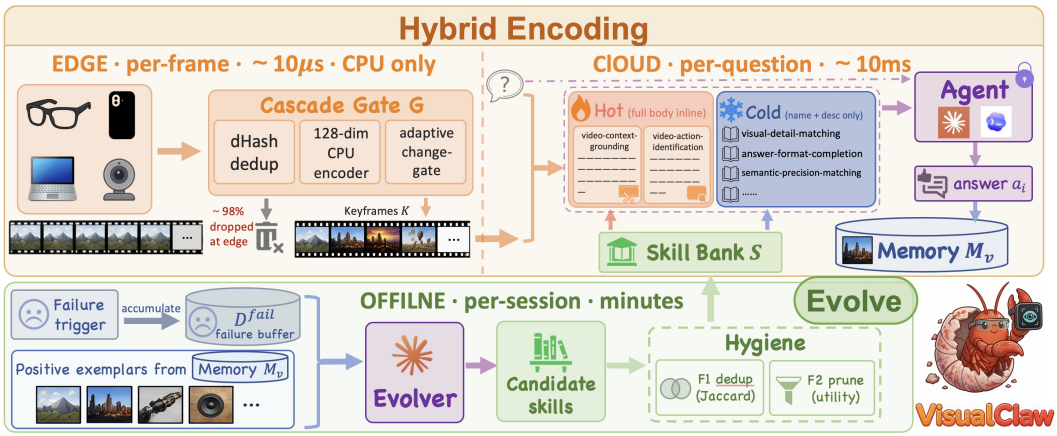


Figure 2: **VisualClaw pipeline.** A causal edge gate selects frames, hot/cold retrieval bounds question-time skill context, and memory-augmented evolution updates the skill bank after failure batches.

### 3.1 Preliminaries and Notation

We write the agent as $M = (\theta, S, M_v, G)$, where $\theta$ denotes frozen cloud-VLM weights, $S = \{s_1, \ldots, s_K\}$ the language-skill bank, $M_v$ an episodic agent memory indexed by sentence embeddings, and $G$ the edge-side visual gate. The LLM evolver $\mathcal{E}$ takes in $S, M_v$ and failure experience $D^{\text{fail}}$ to evolve the skill bank. For a question $q$ over a the frame stream $\mathcal{F}$, the answer $a$ is generated:

$$a \sim \pi_\theta\Big(\cdot \mid q,\ G(\mathcal{F}),\ S^{\text{hot}}(q),\ S^{\text{cold}}(q)\Big), \qquad S^{\text{hot}}(q) = \text{Ret}_S(q,k), \qquad S^{\text{cold}}(q) = \text{Cat}\Big(S \setminus S^{\text{hot}}(q)\Big). \tag{1}$$

Here $\text{Ret}_S(q,k)$ returns the hot top-$k$ skills, while the $\text{Cat}(\cdot)$ produces the name-and-description cold skill catalogues for agent use. Their operating timescales are:

**Per-frame ($\sim 10\,\mu$s gate time, edge).** CPU-only $G$ emits a MAJOR/MINOR/SKIP verdict.
**Per-question ($\sim 10$ ms retrieval time).** Retrieval inlines the hot top-$k$ skills and catalogues the rest.
**Per-session (minutes, offline).** After $N_{\text{evo}}$ failures, evolver $\mathcal{E}$ proposes $S \leftarrow S \cup \mathcal{E}(S, D^{\text{fail}}, M_v)$. These component times exclude VLM inference and network latency.

We further provide the full algorithm in Appendix A.4.

### 3.2 Per-frame: Cascaded Encoding Gate

Streaming video is redundant, while cloud cost grows with every forwarded frame. Our causal gate maps each arriving frame $f_t$ to $g_t \in \{\text{MAJOR}, \text{MINOR}, \text{SKIP}\}$ using only $f_{1:t}$. A MAJOR frame is uploaded as a keyframe; a MINOR frame updates the rolling reference without upload; and a SKIP frame triggers no action. Thus, the uploaded set is $K = \{f_t : g_t = \text{MAJOR}\}$. The visual gate $G$ applies three CPU processes: dHash removes exact and near-duplicate frames against a rolling buffer; a 128-dimensional descriptor combines HSV histogram, luminance, and edge density; and an adaptive gate compares this descriptor with a rolling reference under temporally decaying thresholds. Only $K$ video frames reaches the VLM, allowing online streams without full video buffering or future access.

### 3.3 Per-question: Skill Bank with Hybrid Hot/Cold Injection

Injecting every skill at each query makes prompt cost grow with $|S|$. In detail, each $s_j \in S$ is a reusable reasoning rule stored as a short markdown with a name, description, numbered guidance, and anti-pattern. $S$ starts from a seed skill bank $K_{\text{seed}}$ containing initial visual-reasoning skills, then grows during deployment.

For each question $q$, a template-based ranker matches its task type and orders candidates by lexical overlap with each skill's name and description. The top-$k$ set $S^{\text{hot}} = \text{Ret}_S(q,k)$ is inlined in full; the

remainder forms a name-and-description catalogue $S^{\text{cold}} = \text{Cat}(S \setminus S^{\text{hot}})$ with the main skill content available on demand. Full skill content injection therefore scales with $k$, not $|S|$.

### 3.4 Per-session: Memory-Augmented Self-Evolution

A static $S$ cannot cover unseen failures. We store correctly answered examples in $M_v$ as dense embeddings. After each answer, the benchmark evaluator returns correctness; $D^{\text{fail}}$ stores the failed question, response, and post-answer reference or checker feedback. After $N_{\text{evo}}$ failures accumulate, the evolver $\mathcal{E}$ retrieves a memory context $R = \text{Ret}_{M_v}(D^{\text{fail}})$ based on embedding similarity. For retrieved memory, we support two conditioning modes: `Cat.` directly appends $R$ to the skill evolution prompt ($S_{\texttt{Cat.}}$), while `Guide` asks the evolver to abstract reusable skills from $R$ ($S_{\texttt{Guide}}$):

$$\Delta S_{\texttt{Cat.}} = \mathcal{E}\left(\mathcal{P}_0(S, D^{\text{fail}}) \oplus R\right), \quad \Delta S_{\texttt{Guide}} = \mathcal{E}\left(\mathcal{P}_{\text{guide}}(R) \oplus \mathcal{P}_0(S, D^{\text{fail}})\right), \tag{2}$$

where $\mathcal{P}_{\text{guide}}$ marks $R$ as failure-related evidence and asks the evolver to avoid scenario-specific details. Both modes update the skill bank for future questions. The canonical `FullEvo` setting, shown without a suffix, instead supplies a bounded set of successful answer excerpts as evolver context. For comparison, the `+SkillMemCat` variant directly concatenates retrieved memories into the answer-time context without memory-conditioned skill evolution.

## 4 VisualClawArena: A Multimodal Agentic Benchmark

Static video-QA tests one-shot answers but not whether an agent can ground computer operations in visual evidence. We therefore introduce VisualClawArena, where tool-using agents edit files, maintain workspace state, and recover from prior mistakes.

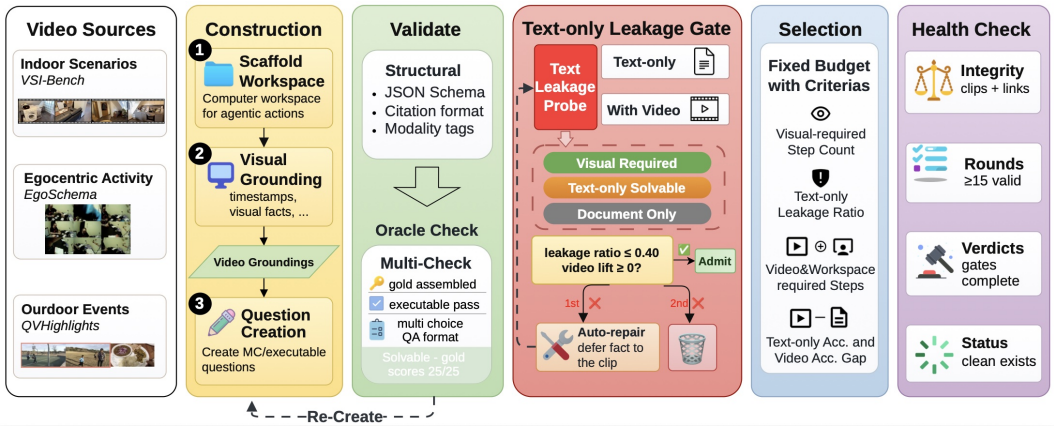


Figure 3: **VisualClawArena curation pipeline.** Videos from three sources pass through construction, validation, text-only leakage filtering, selection, and final health checks.

### 4.1 Building VisualClawArena

We build VisualClawArena from 200 videos: 100 VSI (Yang et al., 2025), 50 EgoSchema (Mangalam et al., 2023), and 50 QVHighlights (Lei et al., 2021). Each scenario combines video with documents, chat/audio traces, dynamic updates, and executable checks. Agents must ground file edits and final workspace state in both visual and textual evidence. Figure 3 summarizes the construction pipeline.

**Input and Construction.** Each candidate separates agent-visible data from evaluator-only files:

- `data/clip/` and `data/workspace/`: video evidence and persistent task files.
- `AGENTS.md`: task rules, output format, citations, and constraints.
- `IDENTITY.md`: the agent role and fixed background assumptions.
- `USER.md`: the user request and intent for the task candidate.

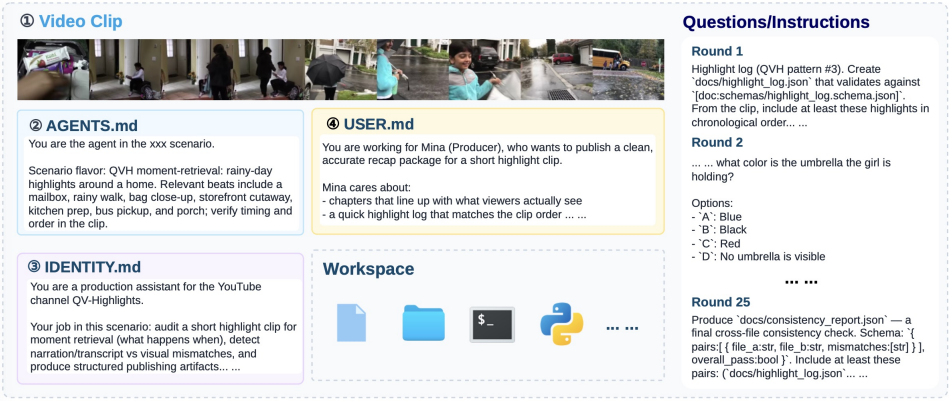


Figure 4: A task example in VisualClawArena with a video, workspace files, multiple-choice rounds, and computer-use requests.

- `data/updates/`: environment changes applied at their scheduled rounds.
- `spec/questions.json`, `scripts/`, and `gold/`: ordered instructions and evaluator-only scoring artifacts. Only the current instruction is exposed to the agent.

Gemini 3.1 Pro first proposes timestamped visual facts; we then construct candidate rounds, files, updates, reference answers, and checkers. These candidates are admitted only after the separate validation stages below. Tasks mix multiple-choice visual questions with executable operations such as writing evidence tables, updating JSON/CSV files, reconciling stale documents with the clip. Figure 4 shows one complete example.

Table 2: VisualClawArena statistics. MC/Exec and VR are average multiple-choice/executable and video-required round counts per scenario. Difficulty uses mean Cascade-8 w/o `FullEvo` accuracy across Codex and Claude Code.

| Source | Scen. | Video Len. | Avg. Rounds | Avg. MC/Exec | Avg. VR Rounds | Easy/Med./Hard |
|---|---|---|---|---|---|---|
| VSI (Yang et al., 2025) | 100 | 24–434 s | 23.9 | 6.7/17.2 | 15.4 | 2/15/83 |
| EgoSchema (Mangalam et al., 2023) | 50 | 180 s | 25.0 | 7.3/17.7 | 21.2 | 39/10/1 |
| QVHighlights (Lei et al., 2021) | 50 | 121–150 s | 24.9 | 7.1/17.9 | 20.4 | 4/21/25 |
| Total | 200 | 24–434 s | 24.4 | 6.9/17.5 | 18.1 | 45/46/109 |

**Validation.** We verify the JSON schema, answer keys, timestamp citations, and declared evidence modalities, and require at least $40\%$ of rounds to use video. In a fresh workspace, the evaluator then applies the gold solution and requires every multiple-choice answer and executable checker to pass. Failed candidates are rewritten and rerun through the same checks.

**Text-only Leakage Gate.** The same fixed VLM (Gemini 3.1 Pro) answers each candidate with the clip hidden and available. We label rounds as `visual_required`, `text_only_solvable`, or `doc_only`. A scenario is kept when its text-only-solvable ratio $L \leq 0.4$ and clip gain $\Delta = \mathrm{Acc}_{\mathrm{clip}} - \mathrm{Acc}_{\mathrm{text}} \geq 0$; otherwise, we repair it by moving the decisive textual clue back into the video. And the task round will be dropped if it failed twice in this check.

**Selection.** We rank surviving candidates by $s = v + 8\max(\Delta, 0) - 6L + 0.5m$, where $v$ counts visual-required rounds and $m$ counts rounds requiring both video and another workspace source. The weights prioritize visual coverage and video information gain, penalize text leakage, and lightly reward cross-modal rounds.

**Health Check.** We finally require every clip, at least 15 valid rounds per scenario, and a leakage label for every round. Table 2 summarizes the source-level scenario construction. For direct comparison, all rows in Table 5 use the same 3,106 scored rounds across the 200 scenarios: the intersection of round IDs completed by every reported configuration.

### 4.2 Bridge to Multimodal Agentic Benchmarks

For each scenario, the runner initializes a clean workspace, seed skill bank, and memory store, then saves selected keyframes locally. Workspace, skill, and memory state persist across ordered rounds

Table 3: Video-QA accuracy (%) with Gemini 3 Flash and GPT-5.2. Cascade columns are streaming; Uniform-8 Plain is offline. (`Cat.`) and (`Guide`) are the `FullEvo` variants from Sec. 3.4; best streaming result per row is **bold**.

| | | Cascade Encoding | | | | | | Uniform-8 |
|---|---|---|---|---|---|---|---|---|
| Benchmark | Model | Plain | Seed | +Evolve | +SkillMemCat | (Cat.) | (Guide) | Plain |
| EgoSchema | Gemini | 52.60 | 67.20 | **68.40** | 65.20 | 64.60 | 68.00 | 60.60 |
| | GPT-5.2 | 64.00 | 66.60 | 66.20 | 67.20 | 65.60 | **68.00** | 70.60 |
| V-MME long | Gemini | 60.33 | 61.56 | 61.33 | 62.78 | 62.56 | **64.22** | 61.44 |
| | GPT-5.2 | 55.89 | 54.00 | 52.22 | 54.67 | 52.78 | **55.89** | 58.78 |
| EgoPlan-Bench | Gemini | 24.62 | **30.80** | 29.93 | 28.31 | 30.04 | 28.85 | 37.96 |
| | GPT-5.2 | 28.42 | 28.74 | 28.31 | 28.09 | 28.85 | **29.39** | 43.06 |
| NextQA | Gemini | 72.70 | 75.10 | 73.90 | 75.50 | **75.70** | 74.50 | 77.70 |
| | GPT-5.2 | 73.20 | 72.00 | 72.30 | 70.90 | 72.50 | **73.30** | 78.90 |

within that scenario and reset before the next scenario. We deploy Codex or Claude Code into our VisualClaw as the backend to execute tasks using their native tool-using infrastructure, which enables bridging our VisualClaw paradigm seamlessly into the computer-used multimodal agent scenarios.

# 5 Experiments

**Evaluation Setup.** We evaluate VisualClaw on static video-QA and VisualClawArena. Video-QA uses frozen Gemini 3 Flash (Google DeepMind, 2025) and GPT-5.2 (OpenAI, 2025); VisualClawArena uses Codex and Claude Code through the workspace bridge in Sec. 4.2. Across both tasks, videos are sampled at 1 fps with at most 8 keyframes per query. Each video-QA configuration starts from the same seed bank with empty run-specific memory and follows a fixed dataset order; VisualClawArena resets state per scenario and preserves its authored round order. Video-QA covers EgoSchema (Mangalam et al., 2023), EgoPlan-Bench (Chen et al., 2023), Video-MME long (Fu et al., 2024), and NextQA (Xiao et al., 2021). Static video-QA entries are full-dataset, fixed-order point estimates; VisualClawArena comparisons use identical scenario and round IDs, with macro accuracy weighting scenarios equally. Dataset-specific and implementation details appear in Appendix A.1.

**Baselines.** For video-QA, `Plain` has no skill or memory scaffold; `Seed` adds the fixed initial skill bank; `+Evolve` updates skills from failures without memory; `+SkillMemCat` concatenates retrieved memories beside the fixed seed bank at answer time; and `FullEvo` uses memory-conditioned skill evolution (*i.e.*, `Cat.`/`Guide`). For VisualClawArena, `Cat.` concatenates failure-retrieved memory into the evolution prompt, canonical `FullEvo` uses only context with correct answers, and w/o `FullEvo` retains the selector and seed bank but disables evolution. Uniform-8 changes only the frame selector, with evolution enabled or disabled as marked.

## 5.1 Accuracy Results and Analysis on video-QA Tasks

**Static video-QA Gains Across Backbones.** In Table 3, `FullEvo` (`Guide`) improves over Plain by $+3.8\%$ on average and peaks at $+15.4\%$ on EgoSchema with Gemini 3 Flash. The `Seed` bank already improves Gemini by $+6.10\%$ on average; `+Evolve`, `+SkillMemCat`, and (`Cat.`) yield $+5.73\%$, $+5.39\%$, and $+5.67\%$. The guided variant adds a smaller but more stable gain: it exceeds `Seed` by $+0.33\%$ on Gemini and $+1.31\%$ on GPT-5.2. Thus, static skills address the dominant video-QA failures, while guided evolution improves consistency across backbones.

**Cascade-fill Beats Uniform-8 at Matched Frame Budget.** Table 4 isolates selection quality at $K{=}8$. Because the natural cascade uses only 1.13–1.51 frames on these short clips, cascade-fill preserves its keyframes and pads the remaining slots uniformly. It improves over Uniform-8 without skill scaffolding by $+1.80\%$ on NextQA and $+3.58\%$ on EgoPlan-Bench, and with `FullEvo` (`Guide`) by $+1.50\%$ and $+1.95\%$.

Table 4: Parity-budget comparison at matched frame budget ($K{=}8$, Gemini 3 Flash).

| Method | NextQA | EgoPlan-Bench |
|---|---|---|
| U-8 | 77.70 | 37.96 |
| Cascade-fill | 79.50 $_{+1.80}$ | 41.54 $_{+3.58}$ |
| U-8 + `FullEvo` | 81.20 | 50.11 |
| Cascade-fill + `FullEvo` | **82.70** $_{+1.50}$ | **52.06** $_{+1.95}$ |

Table 5: VisualClawArena results across all 200 scenarios. VisualClaw (`Cat.`) and `FullEvo` are compared with same-selector w/o `FullEvo` and Uniform-8 controls; best results are in **bold**.

| Backend | Setting | Early | Mid | Late | Micro | Macro |
|---|---|---|---|---|---|---|
| Codex | VisualClaw (`Cat.`) | 49.69 | **56.19** | **57.38** | **59.88** | **54.27** |
| | VisualClaw (`FullEvo`) | **50.66** | 54.66 | 56.88 | 59.50 | 53.89 |
| | VisualClaw w/o `FullEvo` | 48.10 | 52.93 | 53.47 | 57.95 | 51.35 |
| | Uniform-8 | 46.74 | 50.49 | 53.90 | 56.08 | 50.25 |
| | Uniform-8 w/o `FullEvo` | 44.35 | 49.04 | 52.58 | 55.15 | 48.51 |
| Claude Code | VisualClaw (`Cat.`) | **52.03** | **50.91** | **53.50** | **57.79** | **52.16** |
| | VisualClaw (`FullEvo`) | 50.87 | 49.98 | 51.40 | 56.54 | 50.77 |
| | VisualClaw w/o `FullEvo` | 48.80 | 48.51 | 49.68 | 55.34 | 49.00 |
| | Uniform-8 | 40.25 | 44.49 | 47.52 | 49.10 | 43.99 |
| | Uniform-8 w/o `FullEvo` | 40.20 | 41.57 | 45.87 | 47.49 | 42.37 |

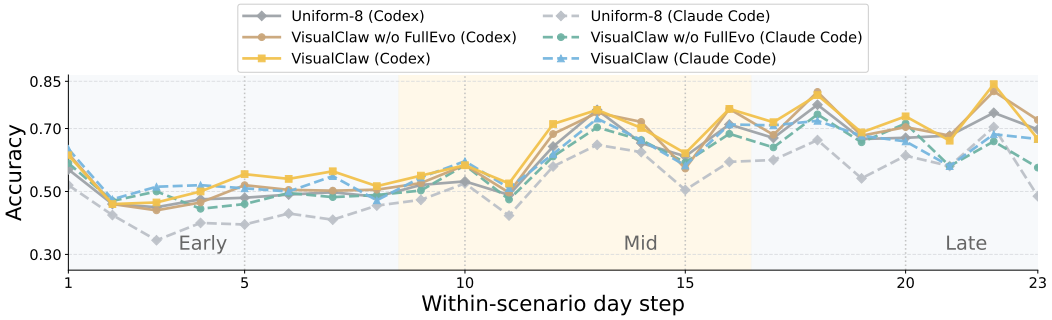


Figure 5: Per-day accuracy on 200 scenarios in VisualClawArena. We present VisualClaw (`Cat.`), VisualClaw w/o `FullEvo`, and Uniform-8. The $x$-axis is within-scenario step; the evolver fires after every five within-scenario failures, and only steps with at least 20 scenarios are displayed.

The combined setting reaches 52.06% on EgoPlan-Bench, +14.10% over plain Uniform-8, showing that frame selection and evolution are complementary.

**Guided Memory Evolution Fits Static video-QA.** In Table 3, (`Guide`) beats (`Cat.`) in six of eight configurations, averaging +1.19% and peaking at +3.40% accuracy on EgoSchema with Gemini 3 Flash; it also wins all four GPT-5.2 settings. Static video-QA gives the model one MC decision, so raw examples can act as narrow context, whereas the guided prefix abstracts them into reusable skills. This motivates testing whether tool-using agents, which can structure longer context through reads, edits, and verification, prefer concrete memory traces. We present additional offline, skill-bank, memory evolution ablations in Appendix A.3.

### 5.2 Results and Analysis on VisualClawArena

Table 5 reports the macro accuracy on VisualClawArena, using macro accuracy to weight scenarios equally and early/mid/late columns for the three within-scenario thirds.

**Agentic `FullEvo` Transfers Beyond video-QA.** VisualClaw (`Cat.`) reaches 54.27% macro accuracy with Codex and 52.16% with Claude Code: +2.92 and +3.16 points over the same-selector w/o `FullEvo` controls, and +4.02 and +8.17 over Uniform-8. For Claude Code, the paired scenario-level difference has a 95% confidence interval of $[+1.23, +5.17]$ points ($p$=0.002). Holding the backend, selector, frame budget, initial skill bank, scenarios, and round IDs fixed, the same-selector contrast isolates feedback-driven evolution in workspace execution.

**Concatenated Memory Fits Agentic Workflows.** VisualClaw (`Cat.`) exceeds the `FullEvo` by 0.38% with Codex and 1.39% with Claude Code. Unlike one-shot video-QA, tool-using backends can structure longer raw memories through file reads, edits, and verification, turning concrete traces into intermediate decisions rather than only prompt burden.

**Temporal Trends on VisualClawArena.** Figure 5 and Table 5 show adaptation across a scenario. Codex (`Cat.`) rises from 49.69% early to 56.19% mid and 57.38% late; Claude Code changes from 52.03% to 50.91% and then 53.50%. Early rounds establish the task contract, while later rounds

Table 6: Per-benchmark API-cost comparison on Gemini 3 Flash. KF/Q = frames sent to the API; tok/Q = input tokens per question; $/run = Gemini 3 Flash spend across the benchmark. Savings are our VisualClaw *vs*. the two baselines.

| Dataset (average duration) | Configuration | KF/Q | tok/Q | $/run | vs Full-frame | vs U-8+FullEvo |
|---|---|---|---|---|---|---|
| EgoSchema (3 min) | Full-frame @1 fps | ∼180 | ∼192,841 | $28.93 | — | — |
| | Uniform-8 + FullEvo | 8.00 | 13,419 | $2.01 | — | — |
| | **VisualClaw** | 2.95 | 9,524 | **$1.44** | −95.0% | −28.4% |
| V-MME long (∼30 min) | Full-frame @1 fps | ∼1,800 | ∼1,926,361 | $520.12 | — | — |
| | Uniform-8 + FullEvo | 8.00 | 15,818 | $4.28 | — | — |
| | **VisualClaw** | 5.41 | 13,420 | **$3.63** | −99.3% | −15.2% |
| EgoPlan-Bench (∼23 s) | Full-frame @1 fps | ∼23 | ∼16,363 | $4.53 | — | — |
| | Uniform-8 + FullEvo | 8.00 | 13,348 | $3.69 | — | — |
| | **VisualClaw** | 1.13 | 10,728 | **$2.97** | −34.4% | −19.5% |
| NextQA (∼30 s) | Full-frame @1 fps | ∼30 | ∼32,429 | $9.73 | — | — |
| | Uniform-8 + FullEvo | 8.00 | 14,025 | $4.21 | — | — |
| | **VisualClaw** | 1.51 | 8,207 | **$2.47** | −74.6% | −41.3% |
| **All experiments total** | | — | — | $563.31 / $14.19 / **$10.51** | −98.1% | −25.9% |

Table 7: Agent/evolver call counts and locally available cost on VisualClawArena.

| Backend | Setting | Agent Calls | Evolver Calls | Agent Cost | Evolver Cost | Total Cost |
|---|---|---|---|---|---|---|
| Claude Code | Uniform-8 | 3,123 | 308 | $665.68 | $21.20 | $686.88 |
| | Cascade-8 | 3,114 | 233 | $611.51 | $15.71 | $627.21 |
| | Δ | −0.3% | −32% | −8.9% | −35.0% | −8.7% |

combine video, text artifacts, and persistent workspace state, where accumulated context and evolved skills become more useful.

Difficulty and source results appear in Appendix A.6. Averaged across backends, (`Cat.`) changes accuracy relative to w/o `FullEvo` by −0.56 points on Easy, +1.08 on Medium, and +5.36 on Hard scenarios, so the benefit concentrates on tasks with more cross-modal and workspace constraints.

### 5.3 Efficiency Results and Analysis

**Cascade Cuts API Cost by an Order of Magnitude.** Table 6 uses full-frame as a cost upper-bound and Uniform-8 + `FullEvo` as the offline baseline for comparison. Against the full-frame upload, the API cost falls by 98.1%, with 99.3% on V-MME long; against Uniform-8 under the same evolved skill bank, VisualClaw sends 1.13–5.41 frames per question and saves 25.9%. This is an explicit accuracy–bandwidth trade-off: the natural cascade trails offline Uniform-8 + `FullEvo` by 2.56–21.26 points across the four Gemini benchmarks, while the matched-$K$=8 results in Table 4 separately show the selector's advantage at equal frame count. Appendix A.7 reports more details.

**Cascade also Wins w.r.t. Costs on VisualClawArena.** Claude Code Cascade-8 costs $627.21 versus $686.88 for Uniform-8 (−8.7%) in Table 7, while agent calls remain nearly unchanged (3,114 vs. 3,123). The $59.67 saving comprises $54.17 in agent calls and $5.50 from 75 fewer evolver calls. Codex follows the same evolver-call direction (120 vs. 146), but its provider cost is unavailable. Moreover, we present cases about skill- and memory-driven evolution in Appendix A.8.

## 6 Conclusion

We presented VisualClaw, a self-evolving multimodal agent for adaptive deployment, and VisualClawArena, a curated multimodal agentic benchmark for tool-using workspaces. VisualClaw jointly designs causal frame filtering, bounded skill injection, and feedback-driven skill evolution. Across four video-QA benchmarks and two VLM families, it reduces API cost and improves same-selector Plain in most settings, while exposing the accuracy–bandwidth trade-off against offline Uniform-8. On VisualClawArena, it improves macro accuracy over no-evolution controls, extending the evidence beyond standard video-QA.

## 7 Declaration of AI Tool Usage

During the preparation of this manuscript, we used OpenAI's GPT-5.5 model for minor language refinement and smoothing of the writing. The AI tool was not used for generating original content, conducting data analysis, or formulating core scientific ideas. All conceptual development, experimentation, and interpretation were conducted independently without reliance on AI tools.

## References


Anthropic. Introducing Claude Haiku 4.5. https://www.anthropic.com/news/claude-haiku-4-5, 2025. System card: https://anthropic.com/claude-haiku-4-5-system-card.

Yi Chen, Yuying Ge, Yixiao Ge, Mingyu Ding, Bohao Li, Rui Wang, Ruifeng Xu, Ying Shan, and Xihui Liu. EgoPlan-Bench: Benchmarking multimodal large language models for human-level planning, 2023.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready AI agents with scalable long-term memory, 2025.

Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, Peixian Chen, Yanwei Li, Shaohui Lin, Sirui Zhao, Ke Li, Tong Xu, Xiawu Zheng, Enhong Chen, Caifeng Shan, Ran He, and Xing Sun. Video-MME: The first-ever comprehensive evaluation benchmark of multi-modal LLMs in video analysis, 2024.

Gemini Team, Google. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context, 2024.

Google DeepMind. Gemini 3 flash: Frontier intelligence built for speed. https://deepmind.google/blog/gemini-3-flash-frontier-intelligence-built-for-speed/, 2025. Model release blog post; see also https://blog.google/technology/developers/build-with-gemini-3-flash/.

Qijun Han, Haoqin Tu, Zijun Wang, Haoyue Dai, Yiyang Zhou, Nancy Lau, Alvaro A Cardenas, Yuhui Xu, Ran Xu, Caiming Xiong, et al. Vlaa-gui: Knowing when to stop, recover, and search, a modular framework for gui automation. *arXiv preprint arXiv:2604.21375*, 2026.

Bo He, Hengduo Li, Young Kyun Jang, Menglin Jia, Xuefei Cao, Ashish Shah, Abhinav Shrivastava, and Ser-Nam Lim. MA-LMM: Memory-augmented large multimodal model for long-term video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. arXiv:2404.05726.

Bin Huang, Xin Wang, Hong Chen, Zihan Song, and Wenwu Zhu. VTimeLLM: Empower LLM to grasp video moments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. arXiv:2311.18445.

Jie Lei, Tamara L Berg, and Mohit Bansal. Detecting moments and highlights in videos via natural language queries. *Advances in Neural Information Processing Systems*, 34:11846–11858, 2021.

Xiaoan Liu, DaeHo Lee, Eric J Gonzalez, Mar Gonzalez-Franco, and Ryo Suzuki. Visionclaw: Always-on ai agents through smart glasses. *arXiv preprint arXiv:2604.03486*, 2026.

Karttikeya Mangalam, Raiymbek Akshulakov, and Jitendra Malik. EgoSchema: A diagnostic benchmark for very long-form video language understanding. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2023. arXiv:2308.09126.

OpenAI. Update to GPT-5 system card: GPT-5.2. https://openai.com/index/gpt-5-system-card-update-gpt-5-2/, 2025. Model card update, December 11, 2025.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems, 2023.

Pinelopi Papalampidi, Skanda Koppula, Shreya Pathak, Justin Chiu, Joe Heyward, Viorica Patraucean, Jiajun Shen, Antoine Miech, Andrew Zisserman, and Aida Nematzadeh. A simple recipe for contrastively pre-training video-first encoders beyond 16 frames. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. arXiv:2312.07395; introduces LongViViT.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST)*, 2023. arXiv:2304.03442.

Pointer. Pointer Agent: A new state of the art for computer use. `https://www.pointer.ai/blog/sota?utm_source=osworld&utm_medium=leaderboard&utm_campaign=sota`, 2026.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3982–3992, 2019. arXiv:1908.10084; underlying paper for the all-MiniLM-L6-v2 sentence-transformers model.

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

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2303.11366.

Enxin Song, Wenhao Chai, Guanhong Wang, Yucheng Zhang, Haoyang Zhou, Feiyang Wu, Haozhe Chi, Xun Guo, Tian Ye, Yanting Zhang, Yan Lu, Jenq-Neng Hwang, and Gaoang Wang. MovieChat: From dense token to sparse memory for long video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. arXiv:2307.16449.

Xiangru Tang, Tianrui Qin, Tianhao Peng, Ziyang Zhou, Daniel Shao, Tingting Du, Xinming Wei, Peng Xia, Fang Wu, He Zhu, Ge Zhang, Jiaheng Liu, Xingyao Wang, Sirui Hong, Chenglin Wu, Hao Cheng, Chi Wang, and Wangchunshu Zhou. Agent KB: Leveraging cross-domain experience for agentic problem solving, 2025.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*, 2024a. arXiv:2305.16291.

Xiaohan Wang, Yuhui Zhang, Orr Zohar, and Serena Yeung-Levy. VideoAgent: Long-form video understanding with large language model as agent. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024b. arXiv:2403.10517.

Ziyang Wang, Shoubin Yu, Elias Stengel-Eskin, Jaehong Yoon, Feng Cheng, Gedas Bertasius, and Mohit Bansal. VideoTree: Adaptive tree-based video representation for LLM reasoning on long videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. arXiv:2405.19209.

Peng Xia, Jianwen Chen, Xinyu Yang, Haoqin Tu, Jiaqi Liu, Kaiwen Xiong, Siwei Han, Shi Qiu, Haonian Ji, Yuyin Zhou, Zeyu Zheng, Cihang Xie, and Huaxiu Yao. MetaClaw: Just talk – an agent that meta-learns and evolves in the wild. *arXiv preprint arXiv:2603.17187*, 2026.

Junbin Xiao, Xindi Shang, Angela Yao, and Tat-Seng Chua. NExT-QA: Next phase of question-answering to explaining temporal actions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9777–9786, 2021. arXiv:2105.08276.

Antoine Yang, Arsha Nagrani, Paul Hongsuck Seo, Antoine Miech, Jordi Pont-Tuset, Ivan Laptev, Josef Sivic, and Cordelia Schmid. Vid2Seq: Large-scale pretraining of a visual language model for dense video captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. arXiv:2302.14115.

Jihan Yang, Shusheng Yang, Anjali W Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. Thinking in space: How multimodal large language models see, remember, and recall spaces. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 10632–10643, 2025.

Shoubin Yu, Jaemin Cho, Prateek Yadav, and Mohit Bansal. Self-chained image-language model for video localization and question answering. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2305.06988.

Sicheng Yu, Chengkai Jin, Huanyu Wang, Zhenghao Chen, Sheng Jin, Zhongrong Zuo, Xiaolei Xu, Zhenbang Sun, Bingni Zhang, Jiawei Wu, Hao Zhang, and Qianru Sun. Frame-Voyager: Learning to query frames for video large language models. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025. arXiv:2410.03226.

Ce Zhang, Taixi Lu, Md Mohaiminul Islam, Ziyang Wang, Shoubin Yu, Mohit Bansal, and Gedas Bertasius. A simple LLM framework for long-range video question-answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024. arXiv:2312.17235.

Guibin Zhang, Haotian Ren, Chong Zhan, Zhenhong Zhou, Junhao Wang, He Zhu, Wangchunshu Zhou, and Shuicheng Yan. MemEvolve: Meta-evolution of agent memory systems, 2025a.

Hang Zhang, Xin Li, and Lidong Bing. Video-LLaMA: An instruction-tuned audio-visual language model for video understanding. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations (EMNLP Demo)*, 2023. arXiv:2306.02858.

Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. Video instruction tuning with synthetic data. *Transactions on Machine Learning Research*, 2025b. arXiv:2410.02713; introduces LLaVA-Video.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2024. arXiv:2308.10144.

Table 8: Uniform-8 + `FullEvo (Guide)` delivers a positive lift across all four benchmarks using Gemini 3 Flash.

| Bench | U-8 plain | U-8 + `FullEvo (Guide)` | Δ |
|---|---|---|---|
| EgoSchema | 60.60 | **73.60** | +13.00 |
| V-MME long | 61.44 | **66.78** | +5.34 |
| EgoPlan-Bench | 37.96 | **50.11** | +12.15 |
| NextQA | 77.70 | **81.20** | +3.50 |

Table 9: EgoSchema leaderboard results. All baselines are reproduced from cited works.

| Method | Acc. (%) | Backbone |
|---|---|---|
| Frozen Bilinear (Mangalam et al., 2023) | 17.6 | ResNet+BERT |
| LongViViT (Papalampidi et al., 2024) | 56.8 | ViViT |
| LLoVi (GPT-4o) (Zhang et al., 2024) | 67.6 | GPT-4o |
| VideoAgent (Wang et al., 2024b) | 60.2 | GPT-4 + LLM planner |
| Gemini 1.5 Pro (Gemini Team, Google, 2024) | 72.2 | Gemini 1.5 Pro |
| VisualClaw (streaming `FullEvo (Guide)`) | 68.0 | Gemini 3 Flash |
| VisualClaw (U-8+`FullEvo (Guide)`) | **73.6** | Gemini 3 Flash |

# A Appendix

## A.1 Detailed Evaluation Setup

**Implementation.** We use Claude Haiku 4.5 (Anthropic, 2025) as the offline video-QA evolver and `all-MiniLM-L6-v2` (Reimers & Gurevych, 2019) as the memory encoder. Videos are sampled at 1 fps with at most 8 keyframes per query. The cascade uses $\tau_{\text{major}}$=0.30 and a 10 s silence ceiling, and $K_{\text{seed}}$=12 skills initialize the bank. Unless otherwise specified, $N_{\text{evo}}$=15 failures trigger evolution on video-QA, while VisualClawArena evolves after every five within-scenario failures.

**Benchmarks.** EgoSchema uses 500 instances with 3-min clips for long-horizon egocentric activity recognition. EgoPlan-Bench uses 923 instances with task-progress videos averaging $\sim$ 23 s, each paired with a current observation and task goal for next-action planning. Video-MME long uses 900 instances over videos longer than 30 minutes, while NextQA uses 1,000 instances over roughly 30 s clips for short-clip causal and temporal reasoning. VisualClawArena contains 200 scenarios and 3,106 task rounds in total. We report macro scenario accuracy so each scenario receives equal weight.

## A.2 Broader Impact

**Potentials for Edge Device Deployment.** Streaming wearables motivate local frame rejection before cloud inference. Our timing claim concerns the CPU gate ($\sim$10 $\mu$s per sampled frame), while measured per-question VLM latency is 2.95–14.56 seconds (Appendix A.7). We therefore claim streaming-compatible frame filtering and lower API traffic, not end-to-end real-time agent response; device decoding, radio energy, and sustained-stream behavior remain hardware-dependent.

**Responsible Deployment.** Because VisualClaw processes continuous visual streams and reuses accumulated experience, deployments should obtain consent for recorded environments, protect stored observations and memories, and audit skill-bank updates before reuse. Access controls and retention limits are especially important for wearable and other always-on settings.

## A.3 Further Analysis

We report complementary static video-QA, skill-bank, memory, and hot/cold injection analyses.

**Offline and Leaderboard Results.** Uniform-8 + `FullEvo (Guide)` improves over Uniform-8 Plain on all four Gemini 3 Flash benchmarks by +3.50 to +13.00 points (Table 8), including +13.00 on EgoSchema and +12.15 on EgoPlan-Bench. The streaming variant trails this offline configuration by 5.60 points on EgoSchema and 2.56 on V-MME long. On EgoSchema (Table 9), streaming VisualClaw reaches 68.00% versus VideoAgent's 60.20%; offline Uniform-8 + `FullEvo` reaches 73.60%, compared with 72.20% for Gemini 1.5 Pro and 67.60% for LLoVi. These are reference points with different backbones and frame-access protocols, rather than a controlled ranking.

**Bank Composition and Memory Dynamics.** Starting from 12 seed skills, `FullEvo` expands the bank to 40–67 skills across the four benchmarks (Table 10). On V-MME long, the top-3 evolved skills accumulate 314 activations at 71% accuracy, showing that newly synthesized skills are actively reused. Memory is retrieved only when evolution fires rather than for every question (Table 11), reducing repeated context while still conditioning future skill updates.

Table 10: Skill bank composition at end-of-run, Gemini 3 Flash with `FullEvo`. Total $= K_{\text{seed}}$ initial seed (kept) + evolved (added by the evolver).

| Bench | Seed | Evolved | Total |
|---|---|---|---|
| EgoSchema | 12 | 28 | 40 |
| Video-MME long | 12 | 45 | 57 |
| EgoPlan-Bench | 12 | 55 | 67 |
| NextQA | 12 | 32 | 44 |

Table 11: Memory ingest and retrieval rates per benchmark using `FullEvo` and Gemini 3 Flash. Retrievals are skill evolver-fusion fetches, where memory was never injected into VLMs.

| Bench | Ingests | Retrievals | Retrieval/Q |
|---|---|---|---|
| EgoSchema | 340 | 11 | 0.022 |
| Video-MME long | 578 | 22 | 0.024 |
| EgoPlan-Bench | 266 | 44 | 0.048 |
| NextQA | 745 | 17 | 0.017 |

Table 12: Model-wise average hybrid hot/cold top-$k$ ablation with `FullEvo` across the four video-QA benchmarks. The cascade frame budget is fixed at $K{=}8$ for every column; ALL injects the whole skill bank rather than changing the frame budget. Best value in each row is bolded.

| Model | $k{=}1$ | $k{=}3$ | $k{=}5$ | ALL |
|---|---|---|---|---|
| GPT-5.2 | 50.48 | **51.90** | 51.47 | 50.12 |
| Gemini 3 Flash | 57.27 | 57.73 | 57.49 | **59.43** |

**Hybrid Injection Top-$k$ Trade-off.** Table 12 summarizes the hot/cold injection grid as model-wise averages over the four video-QA benchmarks while holding the cascade frame budget and `FullEvo` stack fixed. Here $K{=}8$ is the fixed frame budget; top-$k$ controls how many hot skills are injected, and ALL injects the whole skill bank. GPT-5.2 performs best at $k{=}3$ ($51.90\%$), while Gemini 3 Flash performs best with ALL ($59.43\%$). We therefore use $k{=}3$–$5$ as the default for large banks and reserve full-bank injection for compact banks.

### A.4 Detailed Algorithm

Algorithm 1 summarizes VisualClaw. Here, $\mathrm{Ctx}_z$ forms the bounded evolver context: canonical `FullEvo` uses successful examples, while `Cat.` and `Guide` retrieve confidence-gated memories against the failure batch.

**Algorithm 1** VisualClaw agentic streaming inference loop.

**Require:** Agent $M = (\theta, S, M_v, G)$; question stream $\{q_i\}$; frame stream $\{f_t\}$; evolve threshold $N_{\text{evo}}$; hybrid top-$k$; memory mode $z$.

$K \leftarrow \emptyset$; $H \leftarrow \emptyset$; $D^{\text{fail}} \leftarrow \emptyset$
**for** each frame $f_t$ from edge **do** ▷ per-frame, sub-ms
  $h \leftarrow \mathrm{dHash}(f_t)$; **continue** if $\min_{h' \in H} \mathrm{Hamming}(h, h') \leq 6$ ▷ near-dup
  $H \leftarrow H \cup \{h\}$; $e \leftarrow \mathrm{LightweightEncoder}(f_t)$; $g \leftarrow \mathrm{ChangeGate}(e; \tau_{\text{major}}, \text{decay})$
  $K \leftarrow K \cup \{f_t\}$ if $g =$ MAJOR
**end for**
**for** each question $q_i$ **do** ▷ per-question, $\sim$10 ms
  $S_i^{\text{hot}} \leftarrow \mathrm{Ret}_S(q_i, k)$; $S_i^{\text{cold}} \leftarrow \mathrm{Cat}(S \setminus S_i^{\text{hot}})$; $K_i \leftarrow \mathrm{Budget}_8(K)$
  $a_i \leftarrow \pi_\theta(\cdot \mid q_i, K_i, S_i^{\text{hot}}, S_i^{\text{cold}})$; $(r_i, e_i) \leftarrow \mathrm{Evaluate}(a_i, q_i)$ ▷ post-answer only
  **if** $r_i = 1$ **then**
    $M_v \leftarrow \mathrm{Store}(M_v, (q_i, a_i))$
  **else**
    $D^{\text{fail}} \leftarrow D^{\text{fail}} \cup \{(q_i, a_i, e_i)\}$
  **end if**
  **if** $|D^{\text{fail}}| \geq N_{\text{evo}}$ **then** ▷ per-session, minutes
    $R \leftarrow \mathrm{Ctx}_z(M_v, D^{\text{fail}})$
    $\Delta S \leftarrow \mathcal{E}_z(S, D^{\text{fail}}, R)$; $S \leftarrow S \cup \Delta S$; $D^{\text{fail}} \leftarrow \emptyset$
  **end if**
**end for**

### A.5 COMPUTE COST

Total experiments reported in this paper: $\sim$\$190 across Gemini 3 Flash and GPT-5.2 (Azure proxy). Reasoning-token billing on the proxy is not separately exposed and may inflate the GPT-5.2 portion by 1.5–3$\times$ depending on Azure deployment configuration.

Table 13: Approximate compute costs by stage. "Wall" = approximate wall-clock with 6–12 concurrent runs. All Bedrock Haiku evolver calls are bundled into the Gemini-side accounting.

| Stage | Runs | Approx. cost (USD) | Approx. wall (h) |
|---|---|---|---|
| Tier-1 ablation ladder (EgoSchema/V-MME short/TeleEgo) | 16 | 46 | 18 |
| Tier-2 cascade per-stage breakdown | 5 | 2 | 1 |
| Cross-VLM full benchmark (GPT-5.2 6 runs) | 6 | 36 | 8 |
| Headline 4-benchmark $\times$ 2-VLM experiments | 8 | $\sim$50 | $\sim$12 |
| NextQA + EgoPlan-Bench ablation ladders | 14 | $\sim$50 | $\sim$10 |
| **Total** | $\sim$50 | $\sim$**190** | $\sim$50 |

### A.6 ADDITIONAL VISUALCLAWARENA DETAILS

We report complementary difficulty, source, and memory-to-evolver analyses on the matched core.

**Difficulty Trends on VisualClawArena.** Table 14 shows that `FullEvo`'s benefit is difficulty-dependent. Averaged over Codex and Claude Code, VisualClaw (`Cat.`) is slightly lower than w/o `FullEvo` on Easy scenarios (85.05% vs. 85.61%), but improves Medium scenarios (64.92% vs. 63.84%) and strongly improves Hard scenarios (35.14% vs. 29.78%). The same pattern appears for Uniform-8: adding `FullEvo` reduces the Easy average (80.45% vs. 81.80%) but improves Medium (56.63% vs. 54.60%) and Hard (29.44% vs. 26.81%). This matches our interpretation: easy tasks are already near saturation, so extra skills and retrieved failures can add prompt noise or over-correct simple answers; medium and hard tasks have more cross-modal conflict, longer dependencies, and file-level constraints, where evolved skills provide useful procedural bias. Across sources, Indoor/VSI remains the hardest split ($\sim$36–38% macro under `Cat.`), EgoSchema is the easiest ($\sim$81–84%), and QVHighlights sits in between ($\sim$55–57%), so the benchmark is not solved by one source family.

**Memory-to-evolver Ablations.** `Cat.` improves over canonical `FullEvo` by $+0.38$ macro points for Codex and $+1.39$ for Claude Code, suggesting that longer raw memory context is more useful in agentic workflows than in one-shot answer selection.

### A.7 FULL PER-EXPERIMENT TOKEN / COST / LATENCY PROFILE

The headline saving in Sec. 5.3 (Table 6) compares Gemini Cascade + `FullEvo` against the Uniform-8 + `FullEvo` offline ceiling. Table 16 reports the full per-experiment profile for both VLM families across the four benchmarks plus the cross-VLM ablation runs, drawn from the `vlm_usage` field of each results dump.

### A.8 CASE STUDIES

**How Evolved Skills and Memory Correct Errors.** Figure 6 traces two Gemini 3 Flash wins under the same visual evidence: each clip is reduced to one MAJOR keyframe ($1/180$ and $1/36$), so the correction comes from the language scaffold. In Case 1, the baseline follows the surface phrase "picks paint," while retrieved evolved skills enforce the purpose clause and recover GT (D), "loosens color intensity." In Case 2, two confidence-gated memories of the "hold-tight $\rightarrow$ stabilise" pattern guide evolution from (C), "moving themselves," to GT (B), "hold for balance."

**Additional Cases.** We provide two more examples of Gemini 3 Flash as well as GPT-5.2 on the EgoPlan-Bench and the EgoSchema benchmark in Figure 7.

Table 14: VisualClawArena macro accuracy by empirical scenario difficulty. Tiers are assigned on the same 200 scenarios using the average Cascade-8 w/o `FullEvo` accuracy of Codex and Claude Code: Easy $> 75\%$, Medium 55–75%, Hard $\leq 55\%$.

| Backend | Setting | Easy (45) | Medium (46) | Hard (109) |
|---|---|---|---|---|
| Codex | VisualClaw (`Cat.`) | **86.65** | 65.09 | **36.33** |
| | VisualClaw w/o `FullEvo` | 86.26 | **65.53** | 30.95 |
| | Uniform-8 | 84.29 | 60.42 | 31.90 |
| | Uniform-8 w/o `FullEvo` | 85.83 | 58.19 | 29.03 |
| Claude Code | VisualClaw (`Cat.`) | 83.45 | **64.74** | **33.94** |
| | VisualClaw w/o `FullEvo` | **84.96** | 62.14 | 28.61 |
| | Uniform-8 | 76.60 | 52.41 | 26.97 |
| | Uniform-8 w/o `FullEvo` | 77.73 | 49.92 | 24.58 |

Table 15: Memory/evolver-context ablations on the VisualClawArena. `Cat.` appends failure-retrieved memory, `Guide` inserts the same memory as a guided prefix, and `FullEvo` is the canonical success-context setting.

| Backend | Setting | Passed / Total | Micro Acc. | Macro Acc. |
|---|---|---|---|---|
| Codex | `Cat.` | 1,860 / 3,106 | 59.88 | **54.27** |
| Codex | `Guide` | 1,851 / 3,106 | 59.59 | 54.02 |
| Codex | `FullEvo` | 1,848 / 3,106 | 59.50 | 53.89 |
| Claude Code | `Cat.` | 1,795 / 3,106 | 57.79 | **52.16** |
| Claude Code | `FullEvo` | 1,756 / 3,106 | 56.54 | 50.77 |

Table 16: Per-experiment token / cost / latency profile 4 benchmarks $\times$ 2-VLM. \$/Q is the average money (USD) spent per question. lat./Q(s) is the average latency (second) per question on each dataset. KF/Q = avg keyframes per question after the cascade.

| Bench | Variant | Acc. (%) | in_tok/Q | out_tok/Q | \$/run | \$/Q | lat./Q (s) | KF/Q |
|---|---|---|---|---|---|---|---|---|
| EgoSchema | Gemini plain | 52.60 | 3,430 | 15.1 | \$0.53 | \$0.0011 | 9.05 | 2.96 |
| | Gemini `FullEvo` | 68.00 | 9,524 | 5.9 | \$1.44 | \$0.0029 | 9.80 | 2.95 |
| | GPT-5.2 plain | 64.00 | 905 | 5.0 | \$0.14 | \$0.0003 | 4.39 | 2.95 |
| | GPT-5.2 `FullEvo` | 68.00 | 6,369 | 4.9 | \$0.96 | \$0.0019 | 5.43 | 2.95 |
| V-MME long | Gemini plain | 60.33 | 6,111 | 10.8 | \$1.67 | \$0.0019 | 14.56 | 5.43 |
| | Gemini `FullEvo` | 64.22 | 13,420 | 2.3 | \$3.63 | \$0.0040 | 13.66 | 5.41 |
| | GPT-5.2 plain | 55.89 | 5,461 | 4.9 | \$6.18 | \$0.0069 | 9.82 | 5.40 |
| | GPT-5.2 `FullEvo` | 55.89 | 6,307 | 3.8 | \$7.13 | \$0.0079 | 12.58 | 5.40 |
| EgoPlan-Bench | Gemini plain | 24.62 | 1,352 | 10.5 | \$0.40 | \$0.0004 | 8.55 | 1.13 |
| | Gemini `FullEvo` | 28.85 | 10,728 | 1.5 | \$2.97 | \$0.0032 | 8.99 | 1.13 |
| | GPT-5.2 plain | 28.42 | 1,198 | 5.1 | \$0.34 | \$0.0004 | 5.43 | 1.13 |
| | GPT-5.2 `FullEvo` | 28.85 | 10,207 | 5.2 | \$2.83 | \$0.0031 | 4.11 | 1.13 |
| NextQA | Gemini plain | 72.70 | 1,750 | 4.0 | \$0.53 | \$0.0005 | 5.70 | 1.51 |
| | Gemini `FullEvo` | 74.50 | 8,207 | 1.3 | \$2.47 | \$0.0025 | 5.84 | 1.51 |
| | GPT-5.2 plain | 73.20 | 576 | 5.1 | \$0.18 | \$0.0002 | 2.95 | 1.51 |
| | GPT-5.2 hybrid-3 | 68.10 | 2,244 | 4.8 | \$0.68 | \$0.0007 | 9.24 | 1.51 |

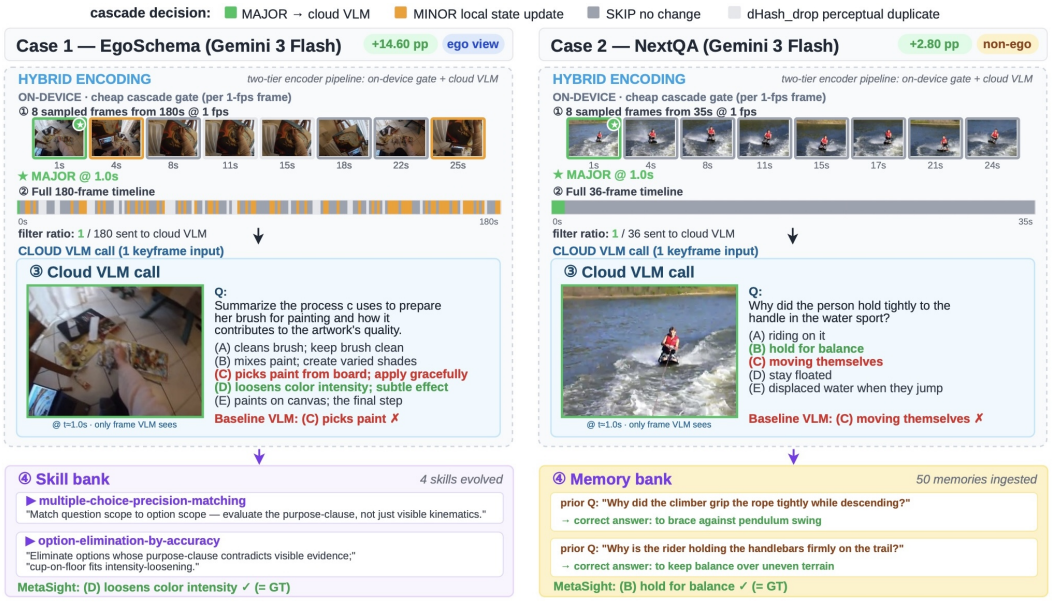


Figure 6: Case studies of VisualClaw. Case 1 (EgoSchema, $+\mathbf{14.60}\%$): single MAJOR keyframe $(1/180)$; evolved skills flip baseline **(C)** $\rightarrow$ GT (D). Case 2 (NextQA, $+\mathbf{2.80}\%$): single keyframe $(1/36)$; two memory entries on the "hold-tight $\rightarrow$ stabilise" pattern flip (C) $\rightarrow$ GT (B).

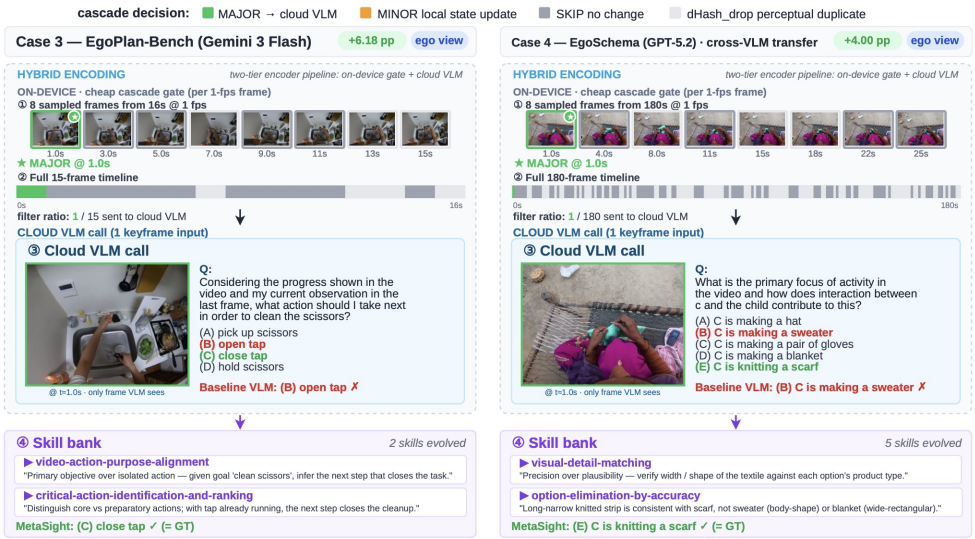


Figure 7: Two further VisualClaw wins, including a cross-VLM transfer. **Case 3** (EgoPlan-Bench, Gemini 3 Flash, $+\mathbf{6.18}\%$): single keyframe $(1/15)$; evolved skills flip **(B)** "open tap" $\rightarrow$ GT **(C)** "close tap." **Case 4** (EgoSchema, GPT-5.2, $+\mathbf{4.00}\%$, cross-VLM): a Gemini-evolved bank applied unmodified to GPT-5.2 corrects **(B)** "making a sweater" $\rightarrow$ GT **(E)** "knitting a scarf."