RTVA: A FRAMEWORK FOR REAL-TIME VISUAL AGENTS IN EVOLVING ENVIRONMENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Alongside task-oriented visual agents that operate through turn-based action loops, recent proactive visual agents aim to support more natural interaction by continuously perceiving evolving environments and responding in real time. Enabling such agents requires more than low latency: perception, response generation, and LLM reasoning proceed concurrently but at different timescales, making it difficult to keep ongoing interaction aligned with a changing environment. Different tasks also demand distinct patterns of intervention and adaptation. We characterize streaming visual interaction through three temporal boundaries: before response initiation, during response delivery, and when delayed reasoning returns. We introduce RTVA, a framework that decomposes real-time visual interaction along these boundaries into composable components within a dual-system architecture. These components separate content generation from whether, when, and how generated content may influence interaction. To adapt RTVA to taskspecific interaction requirements, AutoHarness searches component compositions across these boundaries, then optimizes their behavioral policies. RTVA achieves the strongest overall cross-domain performance in both temporal alignment and interaction quality for real-time visual interaction. We further introduce the Visual In-Flight (VIF) Benchmark, a complementary capability evaluation of whether ongoing responses remain grounded, current, and coherent as visual evidence changes.

1 INTRODUCTION

Humans interacting with a changing environment do not simply perceive once, form a response, and execute it unchanged. They continuously react to new events, adapt ongoing behavior, and revise earlier decisions when the situation changes. A sports commentator may redirect an unfinished description when an unexpected play occurs. Such adaptation is essential for effective real-time interaction, because information can become outdated even while a response is being generated. This motivates proactive visual agents that support continuous and adaptive interaction.

Recent agentic approaches to proactive visual interaction continuously monitor evolving visual streams and determine when emerging events warrant a response (Qian et al., 2025; Zhang et al., 2026; Jiang et al., 2026). In parallel, native real-time multimodal models are increasingly internalizing such interaction decisions, learning directly from streaming inputs when to remain silent or initiate a response (Yao et al., 2026; Wang et al.; Yan et al., 2026; Yang et al., 2026b). Yet proactive interaction does not end at response initiation: as the environment continues to evolve, generated, ongoing, or asynchronously reasoned content may become outdated before it affects the interaction. This raises a complementary question: how should an agent control generated content throughout its lifecycle so that it influences interaction only while it remains appropriate to the current environment?

We organize this problem around three temporal boundaries that expose explicit points for agent-level control. Before generation, *proactive interaction selection* determines whether the current situation warrants intervention. During generation, *response adaptation* determines how an ongoing response should proceed as new visual evidence arrives. When asynchronous reasoning completes, *thought commitment* determines whether the delayed result should still affect subsequent interaction. Together, these boundaries characterize real-time interaction as a set of complementary decisions that can be made by agent.

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.

We introduce **RTVA**, a general-purpose framework for real-time visual agents in evolving environments. RTVA realizes the interaction capabilities associated with these temporal boundaries as composable agent-level components that complement capabilities natively provided by the underlying model. Specifically, these components define both how interaction is controlled at the three temporal boundaries and how foreground behavior (System 1), asynchronous reasoning (System 2), and *Shared Task Memory* interact over time. Its central principle is *generation–execution separation*: producing content is distinct from deciding whether, when, and how that content should affect the interaction. Different real-time tasks, however, require different combinations of these capabilities and different policies for using them. For example, game commentary may benefit from aggressive event-triggered intervention and in-flight revision, whereas procedural guidance may require more conservative progress-conditioned responses. We therefore introduce **RTVA AutoHarness**, which automatically adapts an agent's interaction capabilities and behaviors to the demands of each task. AutoHarness searches over structural compositions, freezes the selected structure, and then tunes its behavioral policies using feedback from complete streaming interactions.

We evaluate RTVA across game commentary, egocentric procedural guidance, and open-ended GUI assistance, comparing against both native real-time models and proactive visual agent frameworks. Across these heterogeneous interaction settings, RTVA achieves the strongest overall cross-domain performance in temporal alignment and interaction quality. Existing task-level metrics, however, cannot directly measure whether an agent appropriately adapts an ongoing response as new visual evidence arrives. We therefore introduce the **Visual In-Flight (VIF) Benchmark**, a causal evaluation of whether a streaming visual agent keeps an ongoing response grounded, current, and temporally coherent as the observed scene evolves during generation.

The main contributions are summarized as follows:

- We introduce **RTVA**, a general-purpose framework that formulates real-time visual interaction around three temporal decision boundaries: proactive interaction selection, in-flight response adaptation, and asynchronous thought commitment. RTVA realizes these interaction capabilities as composable components under *generation–execution separation*.
- We develop **RTVA AutoHarness**, a task-specific optimizer that adapts real-time agents to different interaction patterns by optimizing both the composition of interaction capabilities and their behavioral policies.
- We propose the **Visual In-Flight (VIF) Benchmark**, a causal benchmark for evaluating whether ongoing responses remain grounded, current, and temporally coherent as visual observations evolve during generation.
- We conduct comprehensive experiments across three different real-time visual tasks. RTVA achieves the strongest overall performance in temporal alignment and interaction quality, while ablations and VIF evaluation further validate the contributions of asynchronous reasoning, thought commitment, and inflight response adaptation.

2 Related Work

2.1 Streaming and Real-Time Multimodal Models

Video multimodal models have progressed from offline video understanding of pre-recorded clips (Maaz et al., 2024; Lin et al., 2024), toward continuous processing of streaming visual inputs (Wang et al.; Yan et al., 2026). Early streaming models primarily improve how models process and retain information over time. VideoLLM-Online (Chen et al., 2024) interleaves visual chunks and text, while LiveCC (Chen et al., 2025) learns online video representations from large-scale streaming speech transcription. These approaches establish model-side foundations for processing visual streams without waiting for a complete video. Beyond streaming perception, recent models increasingly incorporate interaction decisions directly into the model. ProAct-VL (Yan et al., 2026) trains a response head to decide when to respond, while StreamMind (Ding et al., 2025) couples stateful streaming perception with event-gated LLM invocation. MiniCPM-o (Cui et al., 2026), ELLSA (Wang et al., 2026), and DuplexOmni (Huang et al., 2026) further support concurrent multimodal perception and generation, with some further incorporating action or asynchronous thinking. Together, these developments increasingly move continuous perception, response timing, and concurrent interaction into the multimodal model itself.

2.2 Proactive and Real-Time Visual Agents

Beyond model-native streaming capabilities, a complementary line of work organizes continuous perception, response decisions, memory, and reasoning at the agent or system level. StreamBridge(Wang et al., 2025a) uses compressed memory and a lightweight activation model for proactive responses, Dispider (Qian et al., 2025) separates continuous perception, response decisions, and asynchronous answer generation, while Mage-VL (Yang et al., 2026b) combines codec-native streaming perception with a lightweight event gate for proactive interaction. Recent agent architectures further separate latency-sensitive interaction from slower reasoning. GPT-Live (OpenAI, 2026) and JoyAI-VL-Interaction (Yao et al., 2026) combine real-time interaction with delegation to background processing, while ProAct (Zhu et al., 2026) runs a slower cognitive system alongside behavioral interaction to derive proactive intentions from accumulated context. Existing systems instantiate real-time interaction capabilities through model-native behaviors or task-specific system designs. RTVA instead exposes the interaction capabilities themselves as configurable agent-level components, allowing initiation, in-flight adaptation, asynchronous reasoning, memory, and their influence paths to be selectively composed into task-specific interaction patterns.

2.3 Automated Optimization of Agentic Systems

As agent systems become increasingly complex, automated optimization offers a way to reduce manual design effort and adapt agents to different tasks. Existing approaches optimize agent systems at different levels, including prompts and language-model programs (Khattab et al., 2024; Yuksekgonul et al., 2025), workflows and computational graphs (Zhuge et al., 2024; Hu et al., 2025; Zhang et al.), and modular agent architectures or multi-agent structures (Shang et al.; Zhang et al., 2025; Zhou et al., 2026; Hu et al., 2026). However, these methods mainly optimize reasoning workflows or system structures for task performance, while real-time interaction also requires adapting how the system behaves over time. RTVA AutoHarness addresses this gap by searching over temporal interaction components and optimizing their behavioral policies, such as when to respond, revise an ongoing response, or use asynchronous reasoning.

3 RTVA Framework

3.1 Architecture Overview

Let $\mathcal{E} = e_1, e_2, \ldots$ denote a timestamped stream of visual, audio, user, agent, tool, and control events, and let s_t denote the live interaction state available at time t. Unlike turn-based interaction, this state continues to evolve while an agent is generating a response, executing it, or performing longer-horizon reasoning. RTVA therefore organizes real-time interaction around three temporal decision boundaries: *response initiation*, which determines whether a response should begin; *in-flight response adaptation*, which determines how ongoing output should react to newly arriving evidence; and *asynchronous thought commitment*, which determines whether delayed reasoning remains applicable when it returns.

RTVA realizes the interaction capabilities around these boundaries as composable agent-level components, from which an agent can selectively instantiate the capabilities needed for a particular interaction pattern. RTVA adopts a dual-system organization. System 1 maintains the latency-sensitive foreground interaction loop, while System 2 performs longer-horizon reasoning asynchronously without blocking System 1. Shared Task Memory maintains evolving task state and coordination context across the two systems. A concrete RTVA configuration determines which capabilities are instantiated, how they exchange information and trigger subsequent processing, how memory is organized, and how each capability behaves. Together, these choices define the agent's *interaction pattern*.

Across different RTVA configurations, the three boundaries share a common principle: *generation–execution separation*: generated responses, revisions, and asynchronous thoughts are treated as proposals whose effects are determined against the interaction state when they would take effect. Let $\mathcal{B} = \{\textsc{initiation}, \textsc{adaptation}, \textsc{thought}\}$. For each $b \in \mathcal{B}$,

$$p_b = \mathrm{Gen}_b(c_{\le t}), \qquad a_b = C_b(p_b, s_\tau, W_b).$$

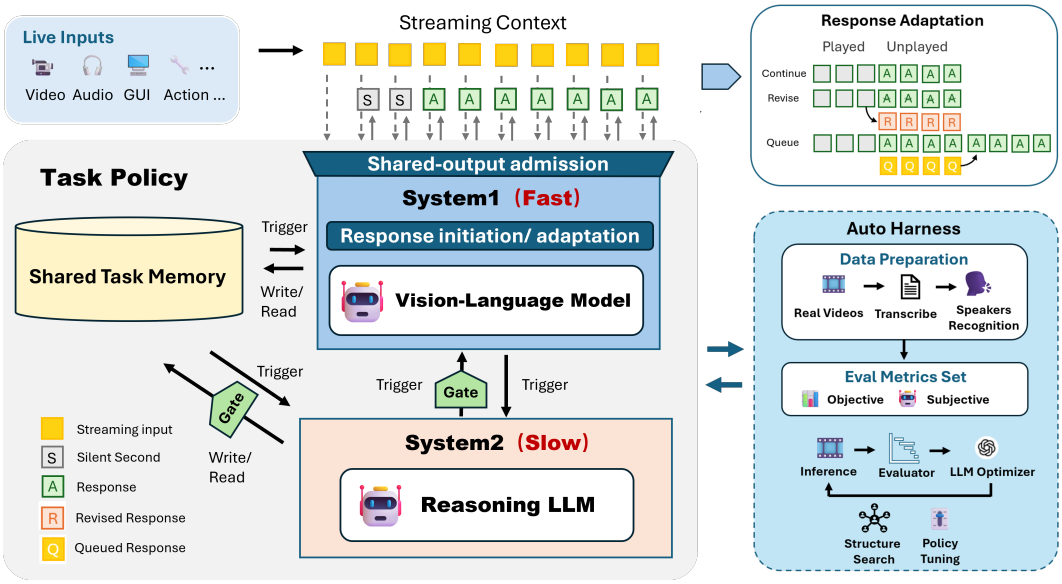


Figure 1: **RTVA framework and task-specific configuration.** (a) System 1, System 2, and Shared Task Memory form the RTVA runtime architecture. (b) Three temporal boundaries govern response initiation, in-flight response adaptation, and asynchronous thought commitment. (c) AutoHarness searches task-specific compositions of interaction components and then tunes their behavioral policies using complete-episode feedback.

, where $c_{\leq t}$ is the causal context used to generate p_b, s_τ is the live state when its effect is considered, and W_b contains intervening events. The corresponding capabilities may be instantiated through native model behaviors, RTVA components, or their composition.

Figure 1 summarizes the framework and task-specific configuration, while Algorithm 1 specifies the streaming runtime.

3.2 System 1: Interaction Selection and Response Adaptation

System 1 implements the latency-sensitive foreground interaction loop, where the first two temporal boundaries, response initiation and in-flight response adaptation, arise.

Response initiation. System 1 integrates the decision of whether to speak with response generation, with both operating on the same evolving interaction context. During streaming perception, System 1 evaluates response initiation at a configurable polling interval. The interval can be adjusted through an agent tool, allowing System 1 to control when it next re-evaluates whether to speak. Updates from memory or asynchronous reasoning can additionally invoke foreground processing through the same initiation policy. Given live state s_t and interaction history H_t,

$$\pi_i(s_t, H_t) \to \{\texttt{silent}, \texttt{speak}\},$$

Native proactive models use their built-in response-initiation behavior. For other backends, we expose the same `speak`/`silent` decision through prompting. When next-token log probabilities are available, synonymous first-token alternatives are grouped into $\mathcal{V}_{\text{speak}}$ and $\mathcal{V}_{\text{silent}}$, and we compute

$$p_{\text{speak}} = \sigma\big(\text{LSE}_{v\in\mathcal{V}_{\text{speak}}}\,\ell_v - \text{LSE}_{v\in\mathcal{V}_{\text{silent}}}\,\ell_v\big)\,,$$

where ℓ_v denotes the next-token log probability of token v. If $p_{\text{speak}} \geq \theta$, generation continues as a streaming response; otherwise, System 1 remains silent. The threshold θ controls the task-specific intervention tendency.

Response adaptation. While a response is being played, System 1 continues to process newly arriving evidence. Let $c_{\leq t}$ denote the current causal interaction context and o_t the speech queue.

Algorithm 1 RTVA Streaming Runtime

Initialize live state s, Shared Task Memory M, and output state o
while interaction is active **do**
  Incorporate newly arrived events into s and update M ▷ **Response Initiation**
  if foreground processing is triggered **then**
    $a_{\text{init}} \leftarrow \pi_i(s_t, H_t)$
    if $a_{\text{init}} =$ `speak` **then**
      Continue generation as public streaming output
    end if
  end if ▷ **In-Flight Response Adaptation**
  if new evidence arrives while o is active **then**
    $a_{\text{adapt}} \leftarrow C_{\text{adapt}}(o, s, W_{\text{adapt}})$
    Update o according to $a_{\text{adapt}} \in \{\text{CONTINUE}, \text{QUEUE}, \text{REVISE}\}$
  end if
  if asynchronous reasoning is triggered **then**
    Capture context c_j at t_j and launch System 2 asynchronously
  end if ▷ **Asynchronous Thought Commitment**
  for each thought σ_j returned at time τ_j **do**
    $W_j \leftarrow \mathcal{E}_{(t_j, \tau_j]}$
    $a_{\text{thought}} \leftarrow C_{\text{thought}}(\sigma_j, s_{\tau_j}, W_j)$
    Commit σ_j according to a_{thought}
  end for
end while

During active speech, the adaptation policy selects

$$a_t = \rho(c_{\le t}, o_t), \qquad a_t \in \mathcal{A}_{\text{adapt}} = \{\text{CONTINUE}, \text{QUEUE}, \text{REVISE}\}.$$

CONTINUE preserves the current output; QUEUE appends a new contribution to the speech queue; and REVISE replaces the targeted unplayed continuation while preserving already delivered content.

Native streaming models realize adaptation through their continuous perception–generation process. For other backends, RTVA re-invokes System 1 at each polling step during active speech with the updated interaction context and current speech queue to select CONTINUE, QUEUE, or REVISE, and updates the speech queue accordingly.

3.3 SYSTEM 2: ASYNCHRONOUS REASONING AND THOUGHT COMMITMENT

System 2 performs longer-horizon reasoning asynchronously alongside the foreground interaction loop.

Triggering. System 2 supports four forms of activation: explicit requests from System 1, periodic invocation, selected stream events, and changes to designated memory stores.

Asynchronous reasoning. When the j-th reasoning process is launched at time t_j, System 2 captures the current task context s_{t_j} while System 1 continues interacting with the live stream. The returned result forms a candidate thought σ_j. Let τ_j denote its return time. Events arriving during asynchronous reasoning form the intervening-event window

$$W_j = \{e_\ell \in \mathcal{E} \mid t_j < t(e_\ell) \le \tau_j\}.$$

Thought commitment. When σ_j returns, its influence is evaluated against the current interaction state and intervening evidence:

$$G(\sigma_j, s_{\tau_j}, W_j) \to \{\texttt{apply}, \texttt{update}, \texttt{discard}\}.$$

We instantiate G with Qwen2.5-VL-7B as a lightweight visual validity gate. `apply` commits the thought's valid influence, `update` requests refreshed reasoning under the current state, and `discard` suppresses its influence. A committed thought is routed to its designated destination: it may update shared memory, enter subsequent System 1 context, or invoke System 1 immediately.

3.4 SHARED TASK MEMORY

Shared Task Memory organizes evolving task state and coordination context into task-specific stores shared across System 1 and System 2 roles. Each store holds a designated type of information, such as task progress, observations, interaction strategies, or intermediate reasoning results, and can be selectively read or updated by different roles. Store updates can invoke subscribed System 1 or System 2 processing, enabling information produced by one role to trigger subsequent processing by another. Delayed System 2 updates pass through thought commitment before being written to shared state.

3.5 TASK-SPECIFIC COMPOSITION AND AUTOHARNESS

RTVA represents a task-specific configuration as a declarative `TaskPolicy` specifying its interaction components, composition, and behavioral controls. AutoHarness optimizes this policy using a two-stage procedure.

Optimizer. At each iteration, an LLM optimizer proposes policy modifications based on the current policy, optimization feedback, and prior search history. Structured experiment memory carries previous successes and failures across iterations to reduce redundant exploration and guide subsequent proposals.

Architecture search. AutoHarness first searches over the compositional structure of RTVA interaction components, including their instantiation, connectivity, information flow, and triggering mechanisms. Architecture candidates may include prompt or parameter changes required by a structural modification, allowing behavior to co-adapt with the proposed structure.

Behavior tuning. After architecture search, AutoHarness freezes the selected architecture and candidate code, then tunes only the behavioral controls of the instantiated RTVA components, including their prompts, polling cadence, thresholds, and other runtime parameters.

Across both stages, optimization improvements can be inherited by subsequent search iterations and are periodically evaluated on held-out episodes; failed validation rolls the search back to the last validated policy. Appendix C provides the candidate-generation protocol, mutation constraints, promotion criteria, validation protocol, and optimization budgets.

4 VISUAL IN-FLIGHT (VIF) BENCHMARK

Existing streaming-video benchmarks primarily evaluate when an agent should respond and what it should say, but do not isolate adaptation after a public response has already begun. Direct evaluation of this capability is confounded by the response trajectory chosen before new evidence arrives: different systems may encounter the same visual event while already producing different utterances. We introduce VIF to evaluate whether an agent should `continue`, `queue`, or `revise` its response as new visual evidence arrives.

4.1 BENCHMARK CONSTRUCTION

We construct cases from egocentric procedural guidance in HoloAssist (Wang et al., 2023) and game commentary from LiveGamingBenchmark (Yan et al., 2026). These domains provide natural streaming interactions in which evolving visual events can change the appropriate trajectory of ongoing speech. Source speech is segmented into one-second target-speaker intervals and annotated as `continue`, `queue`, or `revise` based on the corresponding visual and conversational context. For `queue` and `revise` cases, we additionally annotate the onset of the visual evidence that warrants adaptation. We further assess annotation quality through manual inspection and cross-checking of sampled cases. Detailed annotation procedures and statistics are provided in Appendix E.

4.2 Evaluation and Metrics

For each evaluation window, VIF fixes the preceding interaction context and visual history, after which the evaluated system processes the target window causally and produces its natural streaming output. A frozen LLM evaluator then maps system behavior to `continue`, `queue`, or `revise`. We report per-action success rate using a fixed 3-second evaluation window: `queue` and `revise` succeed if the corresponding action occurs within the window, while `continue` succeeds if neither adaptation action occurs. Detailed timing, matching, and evaluator protocols are provided in Appendix F.

5 Experiments

5.1 Experimental Setup

We evaluate RTVA on three real-time interaction settings: game commentary on LiveGamingBenchmark (Yan et al., 2026), procedural guidance on HoloAssist (Wang et al., 2023), and GUI assistance on GUIDE-Bench (Yang et al., 2026a). We use a unified continuous-interaction protocol and report F1, TimeDiff, LiveU, FinalQ (Yan et al., 2026), and PAUC (Wang et al., 2025b). Timing is measured from actual public output. LiveGamingBenchmark and HoloAssist use their original dialogue, whereas we construct dialogue references for GUIDE-Bench. For each domain, we manually specify a task-specific `TaskPolicy` based on its interaction pattern while using the same RTVA runtime semantics across domains. These human-designed policies serve as the default RTVA configurations and as the baselines for the AutoHarness study in Section 5.5. All comparisons use the same cases, causal observation budgets, task information, and interaction duration. Agentic systems retain their original backbones and controls, thus reflecting system-level performance. Implementation and evaluation details are provided in Appendix A, with additional AutoHarness settings in Appendix C.

5.2 Cross-Domain Baselines and Results

Baselines. We compare against general-purpose real-time models (GPT Realtime (OpenAI, 2025) and Gemini Live (Google)), specialized streaming models (LiveCC (Chen et al., 2025) and ProAct-VL (Yan et al., 2026)), and agentic interaction systems (Mage-VL (Yang et al., 2026b) and JoyAI-VL-Interaction (Yao et al., 2026)). GPT Realtime and Gemini Live use the same GPT-5.4 background-reasoning setup, providing a matched-backbone comparison of their runtime configurations. Backend configurations and evaluation details are provided in Appendix A.

	LiveGaming					HoloAssist					GUIDE-Bench				
Method	F1 (%)↑	TimeDiff↓	LiveU↑	FinalQ↑	PAUC (%)↑	F1 (%)↑	TimeDiff↓	LiveU↑	FinalQ↑	PAUC (%)↑	F1 (%)↑	TimeDiff↓	LiveU↑	FinalQ↑	PAUC (%)↑
A. Models															
ProAct-VL	<u>68.45</u>	<u>2.12</u>	**5.98**	<u>5.70</u>	<u>20.52</u>	7.14	170.22	2.58	1.80	11.14	11.30	8.87	3.91	2.81	14.10
LiveCC	45.58	10.18	4.19	4.74	18.38	7.33	220.18	1.48	1.43	13.51	<u>41.06</u>	7.07	3.99	3.72	12.28
GPT-Realtime + GPT-5.4	54.65	4.02	4.60	5.38	18.50	20.83	<u>6.07</u>	<u>3.77</u>	<u>4.00</u>	<u>20.21</u>	28.52	5.53	3.85	3.70	<u>19.25</u>
Gemini Live + GPT-5.4	15.87	15.00	3.26	4.15	12.35	2.60	211.85	2.29	1.38	10.63	0.11	<u>5.13</u>	2.07	1.21	14.08
B. Agentic Systems															
Mage-VL	58.80	3.78	2.95	4.16	6.19	**33.12**	**2.29**	2.72	3.08	2.30	35.60	7.23	1.98	1.01	0.53
JoyAI + GPT-5.4	28.78	3.33	4.16	3.94	6.62	11.68	8.97	3.62	3.14	4.00	3.13	7.54	<u>4.22</u>	<u>4.22</u>	10.28
RTVA (GPT-Realtime + GPT-5.4)	**70.63**	**1.84**	<u>5.62</u>	**5.81**	**21.05**	<u>30.74</u>	7.29	**4.26**	**4.33**	**20.63**	**51.06**	**4.87**	**5.17**	**4.93**	**22.67**

Table 1: Cross-domain performance under the unified continuous-interaction protocol. F1 and PAUC are reported as percentages. Bold and underlined values indicate the best and second-best results, respectively.

Results. RTVA provides the strongest overall cross-domain performance, with particularly large gains on LiveGaming and GUIDE-Bench. Under the matched GPT-Realtime + GPT-5.4 backbone, RTVA increases F1 from 54.65 to 70.63 on LiveGaming, from 20.83 to 30.74 on HoloAssist, and from 28.52 to 51.06 on GUIDE-Bench, showing that the gains persist without changing the underlying model pair. On HoloAssist, Mage-VL achieves higher F1 and lower TimeDiff, whereas RTVA leads LiveU, FinalQ, and PAUC.

5.3 COMPONENT ABLATIONS

We ablate asynchronous System 2 reasoning and delayed-thought commitment, while evaluating in-flight response adaptation separately with VIF. For thought commitment, the *Always Apply* variant bypasses validation and commits every delayed thought.

Ablation Results. Removing asynchronous System 2 reasoning degrades all evaluated metrics across the three domains (Table 2). The largest changes appear in PAUC on LiveGaming and F1 and TimeDiff on GUIDE-Bench, while FinalQ remains comparatively stable. This pattern is consistent with System 2 complementing the fast interaction loop with longer-horizon reasoning.

Always applying delayed thoughts performs worse than removing System 2 across most metrics, despite retaining asynchronous reasoning. Delayed reasoning is therefore not unconditionally beneficial: information generated from an earlier context may no longer be appropriate when it returns. Explicit commitment provides a temporal boundary between asynchronous thought generation and its effect on public behavior.

Removing in-flight adaptation consistently reduces F1, FinalQ, and PAUC across all three benchmarks, suggesting that when evidence changes during generation, adapting the ongoing response improves both final response quality and overall interaction quality.

	LiveGaming					HoloAssist					GUIDE-Bench				
Configuration	F1$^{\uparrow}$	TimeDiff$^{\downarrow}$	LiveU$^{\uparrow}$	FinalQ$^{\uparrow}$	PAUC$^{\uparrow}$	F1$^{\uparrow}$	TimeDiff$^{\downarrow}$	LiveU$^{\uparrow}$	FinalQ$^{\uparrow}$	PAUC$^{\uparrow}$	F1$^{\uparrow}$	TimeDiff$^{\downarrow}$	LiveU$^{\uparrow}$	FinalQ$^{\uparrow}$	PAUC$^{\uparrow}$
RTVA	**70.63**	**1.84**	**5.62**	**5.81**	**21.05**	**30.74**	**7.29**	**4.26**	**4.33**	**20.63**	**51.06**	**4.87**	**5.17**	**4.93**	**22.67**
w/o System 2	67.96	2.39	5.45	5.78	15.05	29.23	7.59	3.96	4.23	19.04	45.53	6.26	5.00	4.78	21.95
w/o Gate	66.83	2.40	5.10	5.56	13.73	29.07	7.93	3.99	4.17	16.66	45.06	6.40	4.84	4.57	21.31
w/o Adaptation	67.16	2.35	5.39	5.65	11.93	29.22	7.26	4.17	4.24	19.64	45.85	6.08	5.40	4.76	19.87

Table 2: Ablations of asynchronous reasoning, thought commitment, and in-flight response adaptation.

5.4 IN-FLIGHT ADAPTATION ANALYSIS

RTVA achieves the highest success rate on both adaptation actions (Table 3), reaching 39.17% on `revise` and 33.50% on `queue`. Importantly, compared with native GPT-Realtime, which uses the same foreground backbone without RTVA's explicit interaction control, RTVA improves `revise` from 34.00% to 39.17% and `queue` from 10.33% to 33.50%. The gains span two distinct forms of in-flight adaptation. RTVA more reliably revises an ongoing response when new visual evidence invalidates its current trajectory, while the larger gain on `queue` shows stronger ability to preserve the active response and defer newly warranted information for subsequent output. Importantly, these gains do not come at the expense of persistence: RTVA also improves `continue` success over native GPT-Realtime, from 70.42% to 86.33%.

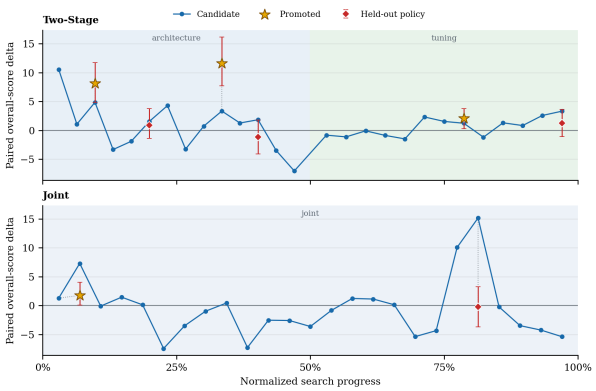
Model	Revise	Queue
ProAct-VL (Yan et al., 2026)	7.83	3.33
LiveCC (Chen et al., 2025)	5.33	3.00
StreamingVLM (Xu et al., 2026)	17.33	6.67
MMDuet2 (Wang et al.)	0.17	0.67
LiveStar (Yang et al., 2025)	22.00	8.67
GPT-Realtime (OpenAI, 2025)	34.00	10.33
RTVA	**39.17**	**33.50**

Table 3: VIF in-flight adaptation performance at $\delta = 3$ seconds.

5.5 AUTOHARNESS ANALYSIS

Finally, we evaluate whether AutoHarness can discover effective task-specific RTVA policies and how its heterogeneous `TaskPolicy` space should be optimized. We consider two factors: initialization, comparing generic and cross-task transferred policies, and optimization strategy, comparing *Joint* and *Two-Stage* search. The strategy comparison uses the same initial policies, data splits, and total search budgets. All reported final performance is measured on a sealed test partition. Detailed search spaces, budgets, promotion criteria, and split isolation are provided in Appendix C.

Optimization strategy. Under matched search budgets on LiveGaming, Behavior-only optimization reaches an Overall score of 46.24 on the sealed test partition, while Joint and Two-Stage reach 48.76 and 50.90, respectively. The improvement over Behavior-only indicates that structure provides



(a) Search dynamics

(b) AutoHarness results

Setting	Generic	Transfer	Human
Task-specific configuration			
LiveGaming	46.84	50.90	54.91
HoloAssist	39.98	38.75	38.38
GUIDE-Bench	38.13	35.49	45.67

Strategy	Overall
Behavior-only	46.24
Joint	48.76
Two-stage	**50.90**

Figure 2: **AutoHarness analysis. (a)** Search dynamics of two-stage and joint optimization on LiveGamingBenchmark under the same transferred HoloAssist initialization. **(b)** Cross-benchmark AutoHarness results and search-strategy ablation.

additional value beyond behavioral tuning. Figure 2(a) further shows that Two-Stage obtains two validated promotions during structural search and another after freezing the selected structure for behavioral tuning, whereas Joint obtains only one promotion despite producing several high-scoring candidates. We therefore use Two-Stage in the subsequent experiments.

Cross-task structural adaptation. Transferring a three-role HoloAssist policy to LiveGaming reveals substantial target-specific structural adaptation. AutoHarness first integrates priority-event detection into the game-state perception path with direct routing to System 1, then replaces the LLM-based priority monitor with deterministic history-aware deduplication. These promotions shift the policy toward faster event routing and suppression of redundant commentary, providing concrete evidence that different interaction modes benefit from different structures. Surprisingly, the transferred policy still provides a stronger initialization than the generic policy on LiveGaming (50.90 vs. 46.84), showing that task-specific structures can retain useful priors across interaction modes.

Task-specific policy search. We further evaluate task-specific policy search across the three interaction modes using Two-Stage optimization. Figure 2(b) compares Generic initialization with a policy transferred from another interaction mode (*Transfer*), alongside the human-designed task-specific policy. Cross-task initialization can provide substantial benefits in some settings: on LiveGaming, Transfer improves the discovered policy from 46.84 to 50.90. On HoloAssist and GUIDE-Bench, Generic initialization instead performs better, reaching 39.98 versus 38.75 and 38.13 versus 35.49, respectively. These results indicate that policies learned from other interaction modes can provide useful priors for automatic search, although their effectiveness depends on the source-target interaction pair. AutoHarness also discovers policies competitive with human design. On HoloAssist, the policy found from Generic initialization surpasses the human-designed policy (39.98 vs. 38.38), while the human-designed policies remain stronger on LiveGaming and GUIDE-Bench.

6 Conclusion

We introduced RTVA, a framework that formulates continuous visual interaction around three temporal boundaries: response initiation, in-flight adaptation, and asynchronous thought commitment. RTVA realizes control at these boundaries through composable agentic components under generation–execution separation, while AutoHarness adapts their composition and behavior to task-specific interaction patterns. Across three interaction domains, our experiments demonstrate strong real-time interaction performance, highlighting the value of extending interaction control beyond response initiation. These results establish temporal control across the full response lifecycle as a practical design principle for visual agents operating in evolving environments.

AI USE STATEMENT

In this work, we used generative AI tools to generate synthetic data, support qualitative annotation, and edit the manuscript to improve readability. We did not use generative AI tools to generate research ideas or propose or refine hypotheses, design or provide feedback on the research methodology or experiments, or interpret results and draw scientific conclusions. We did not use generative AI tools for any other tasks requiring disclosure. All AI-assisted text, synthetic data, and annotations were reviewed by the authors before use. The research ideas, methods, interpretation of results, and scientific conclusions are the authors' own. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses existing datasets under their respective licenses and constructs derived references as described in the appendix. VIF references are generated using the automated annotation procedure and are separately subjected to human verification. Real-time visual agents may produce incorrect or outdated guidance and may process privacy-sensitive visual observations. The systems evaluated in this work are intended for research rather than safety-critical deployment, and consequential applications should retain appropriate human oversight.

REPRODUCIBILITY STATEMENT

Appendix A specifies configurations and reference construction; Appendices F and C detail VIF evaluation and AutoHarness search.

REFERENCES

Joya Chen, Zhaoyang Lv, Shiwei Wu, Kevin Qinghong Lin, Chenan Song, Difei Gao, Jia-Wei Liu, Ziteng Gao, Dongxing Mao, and Mike Zheng Shou. Videollm-online: Online video large language model for streaming video. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18407–18418, 2024. doi: 10.1109/CVPR52733.2024.01742.

Joya Chen, Ziyun Zeng, Yiqi Lin, Wei Li, Zejun Ma, and Mike Zheng Shou. Live: Learning video llm with streaming speech transcription at scale. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 29083–29095, 2025. doi: 10.1109/CVPR52734.2025.02708.

Junbo Cui, Bokai Xu, Chongyi Wang, Tianyu Yu, Weiyue Sun, Yingjing Xu, Tianran Wang, Zhihui He, Wenshuo Ma, Tianchi Cai, Jiancheng Gui, Luoyuan Zhang, Xian Sun, Fuwei Huang, Moye Chen, Zhuo Lin, Hanyu Liu, Qingxin Gui, Qingzhe Han, Yuyang Wen, Huiping Liu, Rongkang Wang, Yaqi Zhang, Hongliang Wei, Chi Chen, You Li, Kechen Fang, Jie Zhou, Yuxuan Li, Guoyang Zeng, Chaojun Xiao, Yankai Lin, Xu Han, Maosong Sun, Zhiyuan Liu, and Yuan Yao. Minicpm-o 4.5: Towards real-time full-duplex omni-modal interaction, 2026. URL https://arxiv.org/abs/2604.27393.

Xin Ding, Hao Wu, Yifan Yang, Shiqi Jiang, Qianxi Zhang, Donglin Bai, Zhibo Chen, and Ting Cao. Streammind: Unlocking full frame rate streaming video dialogue through event-gated cognition. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 13448–13459, 2025. doi: 10.1109/ICCV51701.2025.01249.

Google. Gemini live. https://gemini.google/overview/gemini-live/.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 21344–21377, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/36b7acf6f6010652b3f2a433774a66fe-Paper-Conference.pdf.

Yuntong Hu, Yuting Zhang, Matthew Trager, Yi Zhang, Shuo Yang, Wei Xia, and Stefano Soatto. EvoMAS: Evolutionary generation of multi-agent systems. In *Forty-third International*

Conference on Machine Learning, 2026. URL https://openreview.net/forum?id=ic0AGRIkmY.

Muye Huang, Lingling Zhang, Xingyu Yu, Lei Shi, Zhanyu Ma, Jun Xu, Jiuchong Gao, Jinghua Hao, Renqing He, and Jun Liu. Duplexomni: Real-time listening, seeing, thinking, and speaking for full-duplex interaction, 2026. URL https://arxiv.org/abs/2606.09186.

Hongxiang Jiang, Zengrui Ge, guo chen, Qixiong Wang, Jile Jiao, Xuetao Feng, Yuan Wang, and Yan Wang. Memento: Toward an all-day proactive assistant for ultra-long streaming video. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=FtdbdoGbk3.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan A, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. DSPy: Compiling declarative language model calls into state-of-the-art pipelines. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=sY5N0zY5Od.

Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-LLaVA: Learning united visual representation by alignment before projection. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 5971–5984, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.342. URL https://aclanthology.org/2024.emnlp-main.342/.

Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Khan. Video-ChatGPT: Towards detailed video understanding via large vision and language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12585–12602, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.679. URL https://aclanthology.org/2024.acl-long.679/.

OpenAI. Introducing gpt-realtime and realtime api updates for production voice agents. https://openai.com/index/introducing-gpt-realtime/, August 2025.

OpenAI. Introducing gpt-live. https://openai.com/index/introducing-gpt-live/, July 2026. Accessed: 2026-09-24.

Rui Qian, Shuangrui Ding, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Yuhang Cao, Dahua Lin, and Jiaqi Wang. Dispider: Enabling video LLMs with active real-time interaction via disentangled perception, decision, and reaction. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pp. 24045–24055, June 2025.

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 Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 3841–3865. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/0ae94013da7cd459402fd77874e09ee3-Paper-Conference.pdf.

Haibo Wang, Bo Feng, Zhengfeng Lai, Mingze Xu, Shiyu Li, Weifeng Ge, Afshin Dehghan, Meng Cao, and Ping Huang. Streambridge: Turning your offline video large language model into a proactive streaming assistant. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 132332–132359. Curran Associates, Inc., 2025a. doi: 10.52202/085713-4406. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/bf6939f9058a391c47014731b2486e2a-Paper-Conference.pdf.

Siyin Wang, Wenyi Yu, Xianzhao Chen, Xiaohai Tian, Jun Zhang, Lu Lu, Yuxuan Wang, and Chao Zhang. End-to-end listen, look, speak and act. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=LYyoRqf0Ij.

Xin Wang, Taein Kwon, Mahdi Rad, Bowen Pan, Ishani Chakraborty, Sean Andrist, Dan Bohus, Ashley Feniello, Bugra Tekin, Felipe Vieira Frujeri, Neel Joshi, and Marc Pollefeys. Holoassist: an egocentric human interaction dataset for interactive ai assistants in the real world. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 20270–20281, October 2023.

Yueqian Wang, Songxiang Liu, Disong Wang, Nuo Xu, Guanglu Wan, Huishuai Zhang, and Dongyan Zhao. MMDuet2: Enhancing proactive interaction of video mllms with multi-turn reinforcement learning. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 23257–23270. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/281970ef377580ab9ec3c7d3781110d4-Paper-Conference.pdf.

Yueqian Wang, Xiaojun Meng, Yifan Wang, Huishuai Zhang, and Dongyan Zhao. Proactivevideoqa: A comprehensive benchmark evaluating proactive interactions in video large language models, 2025b. URL https://arxiv.org/abs/2507.09313.

Ruyi Xu, Guangxuan Xiao, Yukang Chen, Liuning He, Kelly Peng, Yao Lu, and Song Han. StreamingVLM: Real-time understanding for infinite video streams. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=gVbPWbA97s.

Weicai Yan, Yuhong Dai, Qi Ran, Haodong Li, Wang Lin, Tao Jin, Xing Xie, Hao Liao, and Jianxun Lian. Proact-VL: A proactive videoLLM for real-time AI companions. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=k9PKgV0L4C.

Saelyne Yang, Jaesang Yu, Yi-Hao Peng, Kevin Qinghong Lin, Jae Won Cho, Yale Song, and Juho Kim. Guide: A benchmark for understanding and assisting users in open-ended gui tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13017–13027, June 2026a.

Senqiao Yang, Kaichen Zhang, Zhaoyang Jia, Jinghao Guo, Yifei Shen, Xinjie Zhang, Xiaoyi Zhang, Haoqing Wang, Xiao Li, Peng Zhang, Xiang An, Yin Xie, Zhening Liu, Xun Guo, Jiahao Li, Shicheng Zheng, Jinglu Wang, Zongyu Guo, Wenxuan Xie, Zihan Zheng, Yuxuan Luo, Bin Li, and Yan Lu. Mage-vl: An efficient codec-native streaming multimodal foundation model, 2026b. URL https://arxiv.org/abs/2607.24904.

Zhenyu Yang, Kairui Zhang, Yuhang Hu, Bing Wang, Shengsheng Qian, Bin Wen, Fan Yang, Tingting Gao, Weiming Dong, and Changsheng Xu. Livestar: Live streaming assistant for real-world online video understanding. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=4n7IifN7yr.

Dingyu Yao, Junhao Zhou, Chenxu Yang, Chuanyu Qin, Haowen Hou, Zheming Liang, Congcong Wang, Yuhang Cao, Shenglong Ye, Shuai Xie, Shuhuan Gu, Haoyang Huang, Qingyi Si, Nan Duan, and Jiaqi Wang. Joyai-vl-interaction: Real-time vision-language interaction intelligence, 2026. URL https://arxiv.org/abs/2606.14777.

Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Pan Lu, Zhi Huang, Carlos Guestrin, and James Zou. Optimizing generative AI by backpropagating language model feedback. *Nature*, 639(8055):609–616, 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-08661-4.

Guibin Zhang, Luyang Niu, Junfeng Fang, Kun Wang, Lei Bai, and Xiang Wang. Multi-agent architecture search via agentic supernet. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 75834–75852. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/zhang25bi.html.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, XiongHui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. AFlow: Automating agentic workflow generation. In Y. Yue, A. Garg, N. Peng,

F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 34040–34077. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/5492ecbce4439401798dcd2c90be94cd-Paper-Conference.pdf.

Kecheng Zhang, Zongxin Yang, Mingfei Han, Haihong Hao, Yunzhi Zhuge, Changlin Li, junhan zhao, Zhihui Li, and Xiaojun Chang. Progressive online video understanding with evidence-aligned timing and transparent decisions. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=oKB0CacHaM.

Han Zhou, Xingchen Wan, Ruoxi Sun, Hamid Palangi, Shariq Iqbal, Ivan Vulić, Anna Korhonen, and Sercan Arik. Multi-agent design: Optimizing agents with better prompts and topologies. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 15844–15872, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/1ab4e0e8f35078109bc78d7b465d306f-Paper-Conference.pdf.

Xiaomeng Zhu, Fengming ZHU, Weijie Zhou, Ye Tian, Zhenlin Hu, Yufei Huang, Yuchun Guo, Xinyu Wu, Zhengyou Zhang, Fangzhen Lin, and Xuantang Xiong. Proact: A benchmark and multimodal framework for structure-aware proactive response. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=YUQPbg4sQz.

Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. GPTSwarm: Language agents as optimizable graphs. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 62743–62767. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/zhuge24a.html.

A Experimental Details and Cross-Domain Protocol

We provide additional details on the experimental setup for LiveGaming, HoloAssist, and GUIDE-Bench. Human-designed TaskPolicies were specified using the task definitions and development data only, without inspecting test examples or results.

A.1 Metrics and Evaluators

The cross-domain reports use five metrics: timestamp F1, TimeDiff, LiveU, FinalQ, and PAUC. The definitions below describe the implemented scoring protocol.

Timestamp F1 (↑). For case i, let P_i and G_i be the sets of integer-second keys in the exported prediction and reference timelines, respectively. A second is counted once regardless of the number of words assigned to it. With $G_i \neq \varnothing$,

$$\mathrm{Precision}_i = \frac{|P_i \cap G_i|}{|P_i|}, \qquad \mathrm{Recall}_i = \frac{|P_i \cap G_i|}{|G_i|},$$

where precision is zero for $P_i = \varnothing$. The case-level score is

$$\mathrm{F1}_i = \frac{2|P_i \cap G_i|}{|P_i| + |G_i|}.$$

Thus a silent prediction on a positive-reference case scores zero.

TimeDiff (↓). Let $g_i^- = \min G_i$, $g_i^+ = \max G_i$, and let r_i be the earliest prediction second at or after g_i^-, when such a second exists. The base delay and out-of-span count are

$$d_i = \begin{cases} r_i - g_i^-, & P_i \cap [g_i^-, \infty) \neq \varnothing, \\ g_i^+ - g_i^-, & \text{otherwise}, \end{cases}$$

$$n_i^{\mathrm{out}} = \left|\{t \in P_i^+ : t < g_i^- - k \text{ or } t > g_i^+ + k\}\right|,$$

$$\mathrm{TimeDiff}_i = d_i + \alpha n_i^{\mathrm{out}}, \qquad k = 2\,\mathrm{s}, \quad \alpha = 0.5\,\mathrm{s},$$

where P_i^+ contains prediction keys with nonempty text. When $G_i \neq \varnothing$ but $P_i^+ = \varnothing$, TimeDiff is set to the span of the reference interval, $g_i^+ - g_i^-$. Such missed responses are additionally reflected in F1. Cases with $G_i = \varnothing$ are unassessable.

LiveU (↑). The text judge receives the background context and timestamped reference and prediction timelines. It assigns integer scores from 1 to 10 for *Time* (synchrony, coverage, and overlap etiquette), *Rate* (cadence, density, and repetition), and *TextU* (clarity and usability as streaming speech). The case score is their unweighted mean:

$$\mathrm{LiveU}_i = \frac{\mathrm{Time}_i + \mathrm{Rate}_i + \mathrm{TextU}_i}{3}.$$

FinalQ (↑). The judge evaluates the chronologically concatenated public prediction for the entire case. It scores *Fidelity* (consistency and non-fabrication), *Continuity* (coherent progression with the context), and *Substance* (task-relevant information and useful coverage), each as an integer from 1 to 10:

$$\mathrm{FinalQ}_i = \frac{\mathrm{Fidelity}_i + \mathrm{Continuity}_i + \mathrm{Substance}_i}{3}.$$

The reference is one valid response track, not a requirement for verbatim agreement.

PAUC (↑). For a case window $[b_i, e_i]$ of length $D_i = e_i - b_i > 0$, nonempty predictions are sorted by timestamp. Each remaining cumulative answer is judged against the reference text using the task-specific coverage rubric. Raw scores $s_{ij} \in \{1, 2, 3\}$ are shifted to $q_{ij} = s_{ij} - 1 \in \{0, 1, 2\}$ before integration.

For the $m_i \geq 1$ retained judge points at times $t_{i1}, \ldots, t_{im_i}$, define

$$x_{i0} = 0, \qquad x_{ij} = (1 - \omega)(t_{ij} - b_i) \quad (1 \leq j \leq m_i),$$
$$x_{i,m_i+1} = D_i, \qquad q_{i0} = 0.5, \qquad \omega = 0.5, \qquad q_{\max} = 2.$$

The normalized left-held step-function area is

$$\mathrm{PAUC}_i = \frac{1}{D_i q_{\max}} \sum_{j=0}^{m_i} q_{ij}(x_{i,j+1} - x_{ij}).$$

The parameter ω compresses the scored time axis: zero preserves the original delays, while one retains only the final quality. The initial ordinate 0.5 corresponds to normalized quality 0.25 before the first judged point.

A.2 Baseline Configurations

Table 4 summarizes the systems used in the cross-domain evaluation. We preserve each baseline's native interaction and response-generation mechanisms whenever possible. All systems are evaluated causally, with access only to observations and interaction history available up to the current time. We use a 10-second visual context sampled at 1 fps for all systems except ProAct-VL, which retains its native 16,384-token KV-cache budget. Benchmark-specific task information is kept consistent across systems, while system-specific formatting follows each model's native interface.

A.3 Qualitative Interaction Example

Figure 3 illustrates how RTVA combines foreground interaction, asynchronous reasoning, shared task memory, and thought commitment during an evolving interaction.

B TaskPolicy Specifications

A `TaskPolicy` declaratively specifies how an RTVA agent is configured for a particular interaction domain. It defines both the structural composition of the agent and the behavioral controls under which its components operate. The runtime validates the policy before interaction. Table 5 summarizes the main policy categories.

System	Model family	Interaction interface	Benchmark adaptation
LiveCC	released checkpoint	streaming commentary with native cache and decoding	
ProAct-VL	released checkpoint	learned speak/silence over visual chunks	
Mage-VL	released checkpoint and framework	codec-video segments; visual speak/silence gate followed by text generation	1-second segments; explicit-silence directives removed
JoyAI-VL-Interaction	released checkpoint and framework	native full-duplex video interaction with asynchronous delegation	GPT-5.4 delegation bridge
GPT-Realtime + GPT-5.4	dated hosted deployments	native Realtime session with asynchronous function calling	client-side GPT-5.4 execution
Gemini Live + GPT-5.4	dated hosted deployments	native Gemini Live session with asynchronous function calling	client-side GPT-5.4 execution
RTVA	dated foreground and System 2 deployments	System 1 foreground interaction with asynchronous System 2	task-specific TaskPolicy

Table 4: Cross-domain baseline configurations.



Figure 3: Example RTVA execution in dual commentary. Foreground System 1 components adapt commentary to unfolding game events, while System 2 asynchronously retrieves task-relevant knowledge and proposes subsequent commentary guidance through shared task memory. Thought commitment rejects an outdated suggestion before it can affect the ongoing interaction.

Structural and behavioral controls. We distinguish TaskPolicy fields according to their effect on the instantiated agent. Structural controls determine the enabled System 1 and System 2 components, persistent stores, task actions, triggers, information routing, and influence paths. Changes to these fields can alter the instantiated agent structure and are authorized during AutoHarness structural search. Behavioral controls determine how the selected structure operates, including prompts, perception and history windows, cadence, confidence thresholds, token limits, silence behavior, and other runtime parameters. After structural search is completed, the selected structure is frozen and AutoHarness modifies only authorized behavioral fields.

B.1 Human-Designed TaskPolicies

Table 6 summarizes the human-designed policies used as task-specific configurations.

B.2 Evaluation Uncertainty

We additionally report 95% bootstrap confidence intervals over evaluation cases to characterize sampling uncertainty in the cross-domain evaluation. These intervals reflect uncertainty induced by the finite evaluation sets rather than run-to-run variation across independent model executions.

Table 5: TaskPolicy abstraction. A TaskPolicy specifies task semantics, agent structure, information flow, and runtime behavior without modifying the RTVA framework implementation.

Category	Representative fields	Semantics
Task contract	role, persona, objectives, rubric	Defines the agent role, objectives, behavioral constraints, and task-specific evaluation requirements.
Perception	modalities, sampling, observation windows, history	Specifies available modalities, causal observation windows, sampling rates, frame budgets, history limits, and input quality.
System 1	directives, inputs, topology, silence, cadence	Specifies foreground instructions, available information sources, runtime members, speaking behavior, and observation cadence.
System 2	prompts, triggers, inputs, outputs, mechanisms	Instantiates asynchronous roles and specifies their prompts, triggers, input contracts, output schemas, and execution mechanisms.
Shared task memory	stores, schemas, mutations	Declares typed persistent state, associated schemas, authorized mutations, and rendering behavior.
Information flow and triggers	routes, events, requests, store changes	Specifies how information moves among components and when asynchronous roles receive opportunities to run.
Actions and tools	task actions, tools, command translation	Declares task-specific actions and their authorized effects on memory or the external interaction.
Runtime controls	thresholds, windows, budgets, timeouts, staleness	Controls invocation cadence, resource budgets, evidence recency, and admission of delayed information.

Component	LiveGaming	HoloAssist	GUIDE-Bench
System 2 routines	one commentary director	one procedural-guidance director	task-progress reviewer plus safety guard
Persistent stores	none	observation state; procedure state	task progress; plan; observations; assistance state
Primary triggers	periodic director review	request, local-observation change, and periodic review	8-second progress review and 4-second safety review
Influence routing	commentary guidance to System 1	procedural guidance to System 1	strategy update or targeted interruption
Behavioral objective	relevant commentary without repetitive filler	grounded and actionable procedural guidance	intervene only on visible need; otherwise remain silent

Table 6: Human-designed TaskPolicy configurations across the three interaction domains.

C AutoHarness Protocol

C.1 Search Space

AutoHarness operates on authorized TaskPolicy fields rather than modifying the RTVA runtime itself. Table 8 summarizes the structural and behavioral search spaces used by the optimizer. Full policy scope does not imply unconstrained mutation: candidate changes remain restricted by stage-specific field classification, protected paths, TaskPolicy schema constraints, interface-preservation rules, and deterministic preflight validation.

C.2 Search Procedure

AutoHarness optimizes authorized `TaskPolicy` fields under two search strategies. *Two-Stage* allocates one search epoch to structural optimization and one to behavioral tuning, freezing the selected structure between the two stages. *Joint* searches structural and behavioral controls simultaneously for two epochs. The two strategies therefore use the same total search budget. All proposed policies are validated against the `TaskPolicy` schema before evaluation.

Method	F1 (%)	TimeDiff	LiveU	FinalQ	PAUC (%)
LiveGaming					
ProAct-VL	68.45 [67.48, 69.38]	2.12 [1.97, 2.28]	5.98 [5.90, 6.05]	5.70 [5.64, 5.77]	20.52 [19.52, 21.53]
LiveCC	45.58 [44.18, 46.95]	10.18 [9.70, 10.67]	4.19 [4.10, 4.28]	4.74 [4.67, 4.80]	18.38 [17.44, 19.33]
GPT-Realtime + GPT-5.4	54.65 [53.66, 55.63]	4.02 [3.81, 4.23]	4.60 [4.55, 4.65]	5.38 [5.32, 5.43]	18.50 [17.50, 19.55]
Gemini Live + GPT-5.4	15.87 [14.79, 16.96]	15.00 [14.39, 15.60]	3.26 [3.17, 3.35]	4.15 [4.06, 4.23]	12.35 [11.76, 12.94]
Mage-VL	58.80 [57.86, 59.71]	3.78 [3.60, 3.96]	2.95 [2.92, 2.98]	4.16 [4.12, 4.19]	6.19 [5.56, 6.83]
JoyAI + GPT-5.4	28.78 [28.00, 29.55]	3.33 [3.11, 3.55]	4.16 [4.11, 4.22]	3.94 [3.89, 3.99]	6.62 [6.03, 7.24]
RTVA (GPT-Realtime + GPT-5.4)	**70.63 [69.71, 71.53]**	**1.84 [1.71, 1.98]**	5.62 [5.57, 5.67]	**5.81 [5.76, 5.87]**	**21.05 [20.02, 22.10]**
HoloAssist					
ProAct-VL	7.14 [5.27, 9.14]	170.22 [148.45, 193.67]	2.58 [2.29, 2.88]	1.80 [1.65, 1.96]	11.14 [10.06, 12.32]
LiveCC	7.33 [5.11, 9.74]	220.18 [192.96, 250.25]	1.48 [1.35, 1.62]	1.43 [1.31, 1.55]	13.51 [12.54, 14.57]
GPT-Realtime + GPT-5.4	20.83 [19.22, 22.50]	6.07 [4.93, 7.72]	3.77 [3.68, 3.87]	4.00 [3.86, 4.16]	20.21 [17.26, 23.24]
Gemini Live + GPT-5.4	2.60 [1.73, 3.60]	211.85 [186.20, 240.26]	2.29 [2.00, 2.59]	1.38 [1.28, 1.50]	10.63 [9.75, 11.50]
Mage-VL	**33.12 [30.37, 35.89]**	**2.29 [1.05, 4.43]**	2.72 [2.67, 2.78]	3.08 [3.01, 3.16]	2.30 [1.19, 3.61]
JoyAI + GPT-5.4	11.68 [10.53, 12.86]	8.97 [6.15, 12.02]	3.62 [3.51, 3.74]	3.14 [3.01, 3.28]	4.00 [2.50, 5.64]
RTVA (GPT-Realtime + GPT-5.4)	30.74 [28.08, 33.41]	7.29 [6.04, 8.90]	**4.26 [4.13, 4.40]**	**4.33 [4.17, 4.50]**	**20.63 [17.63, 23.68]**
GUIDE-Bench					
ProAct-VL	11.30 [9.69, 13.02]	8.87 [8.37, 9.37]	3.91 [3.72, 4.11]	2.81 [2.66, 2.97]	14.10 [13.05, 15.18]
LiveCC	41.06 [40.01, 42.17]	7.07 [6.79, 7.37]	3.99 [3.91, 4.08]	3.72 [3.62, 3.83]	12.28 [10.45, 14.15]
GPT-Realtime + GPT-5.4	28.52 [27.14, 29.90]	5.53 [5.27, 5.79]	3.85 [3.78, 3.93]	3.70 [3.60, 3.81]	19.25 [17.22, 21.33]
Gemini Live + GPT-5.4	0.11 [0.00, 0.29]	5.13 [5.01, 5.24]	2.07 [1.88, 2.27]	1.21 [1.15, 1.28]	14.08 [13.60, 14.58]
Mage-VL	35.60 [34.32, 36.92]	7.23 [6.95, 7.52]	1.98 [1.93, 2.04]	1.01 [1.00, 1.01]	0.53 [0.41, 0.68]
JoyAI + GPT-5.4	3.13 [2.44, 3.89]	7.54 [7.11, 7.97]	4.22 [4.14, 4.31]	4.22 [4.09, 4.36]	10.28 [8.59, 12.01]
RTVA (GPT-Realtime + GPT-5.4)	**51.06 [49.61, 52.47]**	**4.87 [4.60, 5.16]**	**5.17 [5.06, 5.29]**	**4.93 [4.78, 5.08]**	**22.67 [20.32, 25.05]**

Table 7: Cross-domain performance with 95% bootstrap confidence intervals over evaluation cases. Intervals characterize sampling uncertainty from the finite evaluation sets rather than run-to-run variation across independent model executions.

AutoHarness first evaluates the initial policy. At each search iteration, the optimizer proposes candidate policies using only the current authorized policy, optimization-split feedback, prior search summaries, and the active search contract. Candidates are evaluated on the same case–repeat pairs, with two repeats per case. Candidate selection uses a held-out promotion split, while the sealed-test partition is evaluated only after the final policy is frozen.

C.3 Split Isolation and Promotion

Each domain uses disjoint optimization, held-out promotion, and sealed-test partitions. Candidate generation uses only optimization-split feedback and prior search summaries, without access to promotion or sealed-test examples. The held-out promotion split is used only for candidate selection, and the sealed-test partition is not accessed during search or policy selection.

AutoHarness uses an Overall score that combines the five evaluation dimensions after aligning their scales and directions. For F1 and PAUC reported as percentages, TimeDiff (d) in seconds, and LiveU and FinalQ on their native 1–10 scales, we define

$$\text{Overall} = \frac{10}{5}\left(\frac{\text{F1}}{10} + \max(0, T_d - d) + \text{LiveU} + \text{FinalQ} + \frac{\text{PAUC}}{10}\right).$$

Here T_d is a configurable delay scale used only to map TimeDiff to the common optimization range. We fix $T_d = 10\,\text{s}$ for all experiments reported in this paper, which covers the typical delay range in these tasks. Delays beyond T_d therefore receive zero contribution from the TimeDiff term. During search, each case–repeat receives an Overall score, and promotion compares candidate and incumbent mean Overall scores over the same paired case–repeat set. Candidate promotion requires an improvement of at least

$$\Delta\text{Overall} \geq \tau, \qquad \tau = 0.5.$$

Table 8: AutoHarness search space. Structural fields may be modified during structural search; authorized non-structural fields constitute the behavioral search space.

Stage	Searchable fields	Allowed interventions	Constraints
Structural	`/stores`	Add stores, schema fields, and legal mutations; replace authorized store implementations	New stores must expose a declared information path to System 1.
Structural	`/system1/input` `/runtime/system1/topology` `/runtime/system1/urgent_members`	Add information sources or modify foreground topology	Existing System 1 inputs cannot be removed.
Structural	`/system2_roles`, `/system2/roles`	Add or remove complete System 2 roles	New roles require complete input/output contracts and a declared path through which their outputs can influence System 1.
Structural	Role `type`, `factory`, `description`, `input`, `output`, `trigger`, `run_if`, and `mechanism`	Modify sensing, routing, triggering, orchestration, or component implementation	Existing roles are modified field-wise rather than replaced wholesale.
Structural	Candidate-local Python modules	Replace authorized role factories or custom store implementations	Candidate code is sandboxed and must pass interface and runtime validation before evaluation.
Behavioral	System 1 directives, System 2 prompts, native response directives, and speaking instructions	Modify task instructions, speaking criteria, response style, and role behavior	Task-contract and optimizer metadata remain protected.
Behavioral	Cadence and trigger parameters	Modify foreground polling and System 2 execution frequency	Values must satisfy field-local bounds declared by the policy schema or trigger specification.
Behavioral	Perception and history windows, frame limits, and item limits	Modify the amount and recency of evidence available to each invocation	Values must satisfy field-local policy-schema and positivity constraints; no single global numerical range is imposed.
Behavioral	Thresholds and silence controls	Modify speaking decisions and trigger sensitivity	Values follow their field-local Boolean, enumerated, or numerical domains; probability thresholds are restricted to $[0, 1]$.
Behavioral	Model budgets, token limits, retries, and timeouts	Modify inference capacity and runtime tolerance	Values remain subject to field-local policy constraints, deterministic preflight checks, and backend capabilities.

C.4 Search Budget

All search strategies use the same 60-case optimization split and 30-case held-out promotion split, with two repeats per case. Two-Stage runs one structural and one behavioral epoch, while Joint runs two joint epochs, giving both strategies the same total search budget. With a batch size of 8, each epoch contains eight iterations (seven 8-case batches and one 4-case batch). At two candidates per iteration, each strategy evaluates 32 candidate policies in total, excluding the initial policy. Candidates are evaluated only on their current optimization batch, yielding 480 candidate-side optimization case–repeat executions.

Promotion is performed every two iterations, for eight checkpoints in total. At each checkpoint, the selected working policy is evaluated on the 30-case held-out promotion split with two repeats per case and compared against the validated incumbent, yielding 480 candidate-side promotion executions. Thus, each strategy uses 960 candidate-side case–repeat executions, or 1,680 total executions when including the optimization baseline and promotion-incumbent evaluations required for paired comparisons. The selected policy is frozen before evaluation on the 120-case sealed-test partition, which is excluded from the search budget.

D AutoHarness Policy Analysis

Table 10 summarizes the validated promotions observed when initializing LiveGaming search from the transferred HoloAssist policy.

E VIF Partition Statistics

VIF covers procedural guidance from HoloAssist and both solo and multi-party commentary from LiveGamingBenchmark. Table 11 reports the frozen source statistics; each annotation unit corresponds to a one-second target-speaker interval.

Setting	Two-Stage	Joint
Search epochs	1 structural + 1 behavioral	2 joint
Iterations per epoch	8	8
Candidates per iteration	2	2
Total candidate policies	32	32
Optimization cases	60	60
Promotion cases	30	30
Repeats per case	2	2
Candidate-side executions	960	960
Full search executions	1,680	1,680

Table 9: AutoHarness search budget. Two-Stage and Joint use identical candidate and execution budgets. Sealed-test evaluation is excluded from the search budget.

Strategy	Promotion	Policy change	Held-out Gain
Two-Stage	P1	Integrates priority-event detection into game-state perception and routes detected events directly to System 1, while retaining the dedicated priority monitor as a fallback	+8.127
Two-Stage	P2	Replaces the LLM-based priority monitor with deterministic history-aware deduplication that suppresses unchanged or already-covered events	+11.577
Two-Stage	P3	Extends the regular game-state tracking interval from 10 to 11 seconds while retaining the 5-second priority path	+2.056
Joint	P1	Removes periodic coordinator triggering in favor of substantive game-state or priority-event updates and restricts commentary history to actual playouts	+1.739

Table 10: Validated promotions during HoloAssist-to-LiveGaming adaptation.

Evaluation sampling. The source annotations are strongly skewed toward CONTINUE. To ensure sufficient coverage of both maintaining and adapting an active response, VIF uses a fixed stratified evaluation set rather than directly following the source-label distribution. For each domain, we sample 600 CONTINUE cases and 600 adaptation cases, with the latter evenly divided into 300 QUEUE and 300 REVISE cases. This yields 1,200 cases per domain and 2,400 cases in total. The same frozen evaluation cases are used for all compared systems.

Reference annotation. VIF reference annotations are constructed offline using GPT-5.4 over complete source timelines, allowing the annotator to jointly inspect the temporally aligned video, target-speaker transcript, and other-speaker context. Rather than densely labeling every time step, the annotator identifies sparse QUEUE and REVISE events. QUEUE indicates that the ongoing response remains valid but newly available visual evidence warrants a subsequent contribution, whereas REVISE indicates that new visual evidence invalidates or changes the trajectory of the ongoing response. Target-speaker speech intervals without either event are assigned CONTINUE, while silent intervals remain unlabeled. For each event, the annotation records the relevant temporal anchors: t_v denotes the onset of the target visual event, t_m the first mention of the target information for QUEUE, and t_p the point at which the target information begins to alter the response trajectory for REVISE. Each target-speaker interval permits at most one event, and QUEUE and REVISE annotations require the corresponding visual evidence and event-specific metadata. Outputs are automatically checked for schema validity, valid labels and timestamps, uniqueness within each target-speaker interval, and required metadata. Invalid outputs are regenerated once from the complete timeline.

F VIF CAUSAL-PREFIX PROTOCOL AND DIAGNOSTICS

The VIF action is defined relative to this ongoing response. CONTINUE indicates that the current response remains appropriate after the new evidence; QUEUE indicates that the current response remains valid but the new evidence warrants a subsequent response; and REVISE indicates that the

Partition	Source split	Timelines	Intervals	Continue	Queue	Revise
HoloAssist	train, val	8,826	79,995	76,458	726	2,811
LiveGaming-Multi	train, val	1,500	26,400	25,322	503	575
LiveGaming-Solo	train, val, test	1,440	26,215	25,003	355	857

Table 11: Frozen VIF partition and label statistics.

Domain	Continue	Queue	Revise	Total
HoloAssist	600	300	300	1,200
LiveGamingBenchmark	600	300	300	1,200
Total	1,200	600	600	2,400

Table 12: Composition of the stratified VIF evaluation set.

new evidence invalidates or changes content in the ongoing response, requiring it to be revised in flight.

For an event first visible at t_v, the controlled-active Prefix replay reconstructs the observable interaction context preceding the event. The benchmark provides the completed target-speaker interaction history and the portion of the ongoing target-speaker response already played before t_v as textual context. Historical visual observations from the Prefix are then replayed causally up to t_v without requesting new model output. Both the textual context and replayed visual input are restricted to information available up to t_v. This reconstruction provides systems with the same observable active-response context without independently generating the pre-event response. Because visual observations are replayed at discrete sampling times, new response generation is enabled at the first sampling time at or after t_v, so that evaluated outputs are generated only after the target visual evidence becomes observable.

Timing and action matching. VIF uses the source-video timeline as a unified logical clock for replay and scoring. Visual observations, reference event times, generation ticks, and predicted actions are all timestamped in source-time seconds; API wall-clock latency does not affect behavioral scoring. Model actions are assigned to the source-time generation tick at which the corresponding request is issued. For QUEUE and REVISE, a case is successful if at least one action matching the reference label occurs within the corresponding scoring window, $[t_m, t_m + 3]$ and $[t_p, t_p + 3]$, respectively, with intervals clipped to the available source bounds. An earlier non-matching action does not invalidate a subsequent matching action within the same window. For CONTINUE, the scoring window is $[t_v, t_v + 3]$, and the case is successful if neither a QUEUE nor a REVISE action occurs within the window. Consequently, no output within the scoring window counts as success for CONTINUE and failure for QUEUE or REVISE. All scoring windows use inclusive boundaries.

VIF evaluates each controlled rollout using Success Rate, measuring whether the system exhibits the reference in-flight behavior within the corresponding evaluation window. For systems that produce open-ended responses, a fixed GPT-5.4 semantic evaluator maps the rollout output to CONTINUE, QUEUE, and REVISE actions after inference. A rollout is counted as successful when the agent's behavior matches the reference under the controlled Prefix protocol, and Success Rate reports the fraction of successful cases. We report QUEUE and REVISE success in the main comparison, with CONTINUE success reported separately as a complementary control.

Action-balanced performance. Because high CONTINUE success can be achieved by systems that rarely alter an ongoing response, we evaluate persistence and adaptation jointly using an action-balanced score. Specifically, we define

$$\text{Balanced} = \frac{S_{\text{CONTINUE}} + S_{\text{QUEUE}} + S_{\text{REVISE}}}{3},$$

where S_a denotes the Success Rate for action a. This macro-average assigns equal weight to the three VIF behaviors and is therefore independent of their sampling proportions in the evaluation set.

Model	CONTINUE	QUEUE	REVISE	Balanced
ProAct-VL	98.25	3.33	7.83	36.47
LiveCC	98.84	3.00	5.33	35.72
StreamingVLM	93.41	6.67	17.33	39.14
MMDuet2	99.83	0.67	0.17	33.56
LiveStar	83.00	8.67	22.00	37.89
GPT-Realtime	70.42	10.33	34.00	38.25
RTVA	86.33	**33.50**	**39.17**	**53.00**

Table 13: VIF success rates (%) across all three in-flight actions. Balanced success is the macro-average of CONTINUE, QUEUE, and REVISE, assigning equal weight to each action.

Table 14: Human verification of VIF semantic labels on HoloAssist and LiveGamingBenchmark.

Action	**HoloAssist**	**LiveGamingBenchmark**	**Overall**
QUEUE	84.8% (28/33)	**97.0%** (32/33)	90.9% (60/66)
REVISE	**90.9%** (30/33)	87.9% (29/33)	89.4% (59/66)
CONTINUE	**100.0%** (34/34)	**100.0%** (34/34)	**100.0%** (68/68)
Overall	**92.0%** (92/100)	**95.0%** (95/100)	**93.5%** (187/200)

High CONTINUE success should be interpreted jointly with adaptation performance, as systems that rarely alter an ongoing response can perform well on CONTINUE while failing to react when new evidence requires adaptation. RTVA achieves 53.00% Balanced success, compared with 39.14% for the strongest baseline under this measure, showing that its advantage persists when CONTINUE performance is considered jointly with adaptation success. For the matched GPT-Realtime comparison, System 2 reasoning and Gate-based thought commitment are disabled in this evaluation, such that the RTVA configuration differs only by its in-flight adaptation mechanism.

Human Verification of Semantic Evaluation We assess the reliability of VIF semantic labeling through human verification on HoloAssist and LiveGamingBenchmark. For each dataset, we sample 100 examples stratified across the three VIF actions, comprising 33 QUEUE, 33 REVISE, and 34 CONTINUE cases. Human review assesses whether each assigned semantic label is consistent with the observed interaction behavior.

The semantic evaluator achieves a 93.5% human verification rate overall, with 92.0% on HoloAssist and 95.0% on LiveGamingBenchmark. CONTINUE is verified in all audited cases, while errors are concentrated on the two adaptation behaviors, QUEUE and REVISE.

G RUNTIME LATENCY ANALYSIS

We additionally analyze the runtime latency of RTVA to distinguish latency inherited from the underlying models and serving deployment from overhead introduced by RTVA's interaction control.

Response initiation latency. RTVA places response-initiation control on the first-token generation path rather than introducing a separate post-generation decision step. Native proactive models use their built-in response-initiation behavior; for other backends, RTVA exposes a `speak`/`silent` decision through prompting and, when next-token log probabilities are available, aggregates the first-token probability mass over the corresponding alternatives. If the speaking probability exceeds the task-specific threshold, generation continues directly as a streaming response; otherwise, System 1 remains silent. Response initiation therefore requires no additional model round after the first output becomes available, making its critical-path latency approximately the time-to-first-token (TTFT) of the underlying model, which varies with the model and serving deployment rather than the RTVA control mechanism itself. As a conservative bound, the observed request-to-completion latency across foreground generations is 1.626 s at P50, 3.160 s at P95, and 4.358 s at P99; first-token availability necessarily precedes completion.

Dataset	P50 (s)	P95 (s)	P99 (s)
LiveGaming	2.086	3.809	5.411
HoloAssist	2.561	4.294	5.823
GUIDE-Bench	2.743	4.269	4.482
Overall	2.148	3.926	5.452

Table 15: Wall-clock request-to-completion latency for revised generations observed during the main experiments. Because revised responses are streamed, completion latency provides a conservative upper bound on revision-onset latency; historical traces do not retain exact first-output timestamps.

In-flight adaptation latency. RTVA supports `queue` and `revise` through different latency paths. A queued update is scheduled for delivery immediately after the active utterance and therefore does not initiate a separate generation path whose latency can be measured independently. Its delivery naturally waits for the active utterance to finish. A revision instead redirects the active response to a new generation without waiting for the original response to complete, and can begin affecting the interaction once the first token of the revised generation becomes available. We measure the wall-clock interval from an urgent revision request to completion of the corresponding revised generation. Across revision events observed during the main experiments, the median, P95, and P99 request-to-completion latencies are 2.148 s, 3.926 s, and 5.452 s, respectively. Because revised responses are streamed as they are generated, these completion times provide conservative upper bounds on when a revision can begin affecting the interaction. Table 15 reports the corresponding per-domain distributions.

Gate overhead. The Gate performs a lightweight online decision that determines whether an asynchronous update remains valid for commitment under the current interaction state. We measure its latency from invocation to the returned Gate decision. Across 12,167 Gate calls observed during our experiments on A100, all calls completed successfully. The median decision latency is 139 ms, with P95 and P99 latencies of 163 ms and 189 ms, respectively.