STREAMPRO-DUPLEX: A FULL-DUPLEX MULTIMODAL AGENT FRAMEWORK FOR ALWAYS-ON STREAMING INTERACTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Always-on streaming interaction requires assistants to continuously participate in open-ended interactions and to determine not only *when to engage* and *what to say*, but also *how to interact* with the user and *which intent to serve* as user multiple needs emerge, coexist, and evolve over time. Existing streaming systems have advanced the first two capabilities but remain limited in the latter two, largely following an ***Isolated Interaction*** paradigm that segments interaction into separate turns and handles user intents independently. Such isolation hinders natural full-duplex communication and makes it difficult to track multiple ongoing user needs across interleaved interactions. To address these limitations, we propose **StreamPro-Duplex**, a full-duplex multimodal agent framework for always-on streaming interaction. For *how to interact*, we decouple streaming understanding from interaction control and train a plug-and-play *Duplex Controller* that continuously determines and executes appropriate full-duplex actions. For *which intent to serve*, we design a persistent intent management mechanism that maintains evolving intents across interactions and coordinates concurrently triggered intents according to their priorities. We further train a unified *Intent Monitor* that jointly monitors multiple proactive intents over the shared streaming context in a single inference, avoiding repeated model execution for each intent. To support training, we construct **StreamInteract-138K**, a large-scale dataset covering representative always-on streaming interaction scenarios. Experiments demonstrate that StreamPro-Duplex achieves strong performance across full-duplex interaction, real-time streaming, and proactive streaming tasks, validating its effectiveness for always-on streaming interaction.

1 INTRODUCTION

Human–AI interaction is increasingly shifting from offline exchanges toward streaming settings, where audio-visual information unfolds continuously over time and the system needs to interpret the evolving context online (Chen et al., 2024; Xu et al., 2025c; Li et al., 2026). A representative setting for streaming interaction is the ***Always-on Assistant*** (Lu et al., 2026a): instead of terminating after a single request–response exchange, the assistant remains active across successive requests and periods without explicit user input, attending to both the user and the surrounding environment. Such persistent interaction requires capabilities beyond conventional streaming understanding: the assistant must determine not only *when to engage* and *what to say*, but also *how to interact* with the user and *which intent to serve* as multiple user needs emerge, coexist, and evolve over time.

As illustrated in Figure 1, always-on streaming interaction introduces two additional requirements. **(1) Full-duplex interaction control:** as the user and assistant dynamically share the conversational floor, the system must determine *how to interact*—whether to listen, respond, backchannel, interrupt, or stop an ongoing response—to support natural and fluid communication. **(2) Persistent multi-intent management:** as interaction unfolds over time, multiple user needs may emerge, persist, and evolve concurrently, including real-time requests, proactive intents, and context-triggered needs. The system must therefore determine *which intent to serve* at each moment while maintaining the states, priorities, and dependencies of these intents across interactions.

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

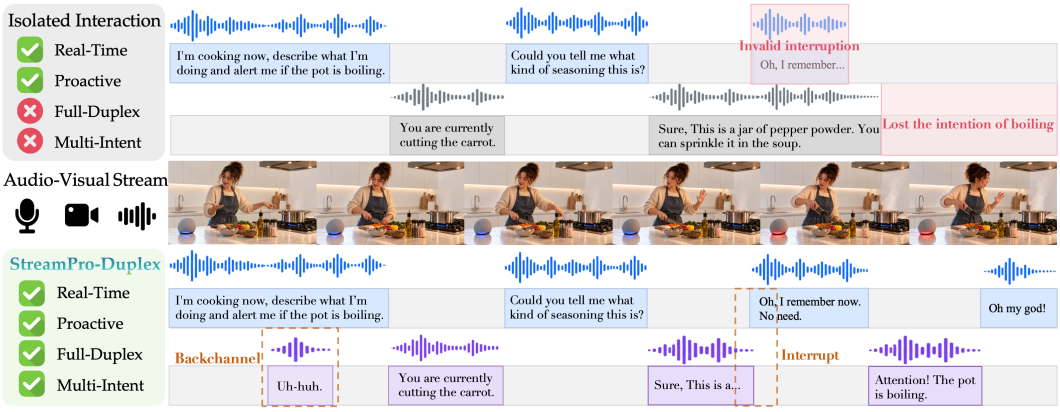

Figure 1: Comparison between **Isolated Interaction** and **StreamPro-Duplex**. Existing streaming approaches typically segment interaction into isolated turns and handle user intents independently, limiting full-duplex interaction and the ability to track multiple ongoing user needs. In contrast, StreamPro-Duplex supports natural full-duplex behaviors such as backchanneling and interruption while persistently maintaining and coordinating intents across interleaved interactions.

These two requirements remain insufficiently addressed by existing streaming systems, limiting their ability to support truly always-on streaming interaction. Streaming video understanding has made substantial progress on *when to engage* and *what to say*, enabling continuous perception, response timing, and context-grounded generation (Chen et al., 2024; Xu et al., 2025c; Lu et al., 2026a). Recent multimodal full-duplex systems further extend these capabilities toward *how to interact*, supporting overlapping listening and speaking, backchanneling, and interruption handling (OpenBMB, 2026; Huang et al., 2026; Orantqing et al., 2026; Team & Group, 2026). However, such capabilities are not yet well integrated into mainstream always-on streaming frameworks. Beyond interaction control, persistent multi-intent management remains another underexplored challenge. Prior studies (Lu et al., 2026b) observe that *when a real-time request arrives, it may take precedence over previously established proactive intents, causing ongoing tasks to be forgotten across interleaved interactions.* This behavior reflects a broader limitation in maintaining and coordinating multiple intents as they emerge and evolve throughout interaction. Together, these limitations reflect an ***Isolated Interaction*** paradigm, where interaction is segmented into isolated turns and intents are handled independently.

To bridge these gaps, we propose **StreamPro-Duplex**, a full-duplex multimodal agent framework for always-on streaming interaction, consisting of three coordinated layers: Real-Time Full-Duplex Control, Memory and Intent Management, and Monitoring and Response Generation. To address *how to interact*, we view streaming video understanding and full-duplex control as largely orthogonal problems. Based on this observation, we adopt a decoupled design and train a *Duplex Controller* dedicated to real-time interaction control. This allows StreamPro-Duplex to retain existing offline or streaming models as backbones while equipping them with full-duplex capabilities in a plug-and-play manner. To address *which intent to serve*, we design a multi-intent management mechanism that maintains evolving intents and their states across turns, while coordinating concurrently triggered intents according to their priorities. In addition, we train an *Intent Monitor* that jointly monitors multiple proactive intents over the shared streaming context in a single inference, avoiding repeated execution of the streaming backbone for each proactive intent (Zhang et al., 2026).

To support streaming interaction training, we construct **StreamInteract-138K**, a large-scale dataset covering representative streaming interaction scenarios, including *interleaved real-time and proactive intents*, *multiple concurrent proactive intents*, and full-duplex interactions involving pauses, backchannels, and interruptions. Experiments demonstrate that StreamPro-Duplex achieves strong performance across full-duplex interaction, real-time streaming, and proactive streaming tasks, validating its effectiveness for always-on streaming interaction.

In summary, our contributions include: (1) We propose **StreamPro-Duplex**, a full-duplex multimodal agent framework for always-on streaming interaction, comprising three coordinated layers: Real-Time Full-Duplex Control, Memory and Intent Management, and Monitoring and Response Generation. (2) We introduce a plug-and-play *Duplex Controller* that decouples streaming understanding from

interaction control, together with persistent multi-intent management and a unified *Intent Monitor* that jointly tracks evolving intents over shared streaming context. (3) We construct **StreamInteract-138K**, a large-scale dataset covering representative full-duplex and multi-intent streaming interaction scenarios. (4) Experiments demonstrate that StreamPro-Duplex achieves strong performance across full-duplex interaction, real-time streaming, and proactive stremaing tasks.

2 Related Work

Streaming Video Understanding. Streaming video understanding extends conventional offline video analysis to continuously arriving visual streams, requiring models to causally process and respond to evolving content. VideoLLM-online (Chen et al., 2024) pioneered streaming video language models for continuous video understanding. Building on this paradigm, subsequent studies have explored two complementary directions. One line focuses on *real-time streaming understanding*, as represented by StreamingVLM (Xu et al., 2025c) and MOSS-VL-Realtime (Wang et al., 2026), emphasizing continuous perception and low-latency interaction. Another line investigates *proactive streaming interaction*, including MMDuet2 (Wang et al., 2025), Streamo (Xia et al., 2025), and StreamPro (Li et al., 2026), enabling models to determine appropriate moments for engagement. More recent systems, such as JoyAI-VL-Interaction (Yao et al., 2026) and StreamArena (Zhang et al., 2026), further integrate these capabilities toward richer and more general streaming interaction.

Multimodal Full-Duplex Interaction. Recent multimodal models have begun moving beyond turn-based interaction toward continuous full-duplex communication. MiniCPM-o 4.5 (OpenBMB, 2026) introduces Omni-Flow to temporally align streaming multimodal inputs and outputs. DuplexOmni (Huang et al., 2026) decouples real-time multimodal interaction from complex reasoning through asynchronous collaboration between interaction and thinking layers. More recent systems further integrate full-duplex interaction with agentic execution. Gander (Orantqing et al., 2026) adopts a Cerebellum–Brain architecture that couples real-time multimodal interaction with asynchronous reasoning, tool use, and long-horizon task execution, while Realtime-Venus (Team & Group, 2026) introduces a dual-loop architecture that preserves foreground conversation while delegating reasoning and tool execution to a background process. These approaches primarily advance full-duplex interaction or its integration with asynchronous agentic execution. In contrast, StreamPro-Duplex focuses on the *always-on streaming assistant* setting, where interaction persists across requests and multiple real-time and proactive intents may coexist, evolve, and become actionable over time.

3 StreamPro-Duplex Framework

We present **StreamPro-Duplex**, a hierarchical full-duplex multimodal agent framework tailored for persistent, always-on streaming human-agent interaction. As illustrated in Figure 2, the proposed architecture consists of three tightly coordinated layers: **Real-Time Full-Duplex Control**, which continuously determines when and how the agent should engage; **Memory and Intent Management**, which preserves long-term streaming context and dynamically tracks evolving, multi-faceted user intents; and **Monitoring and Response Generation**, which adaptively detects actionable intent triggers and produces customized responses for explicit requests and implicit contextual demands. The layered design decouples low-level interactive control from high-level contextual reasoning and response synthesis, enabling full-duplex streaming interaction without backbone model retraining.

3.1 Layer 1: Real-Time Full-Duplex Control

Decoupled Full-Duplex Design. Recent advances in foundation models and streaming video understanding have endowed existing systems with strong capabilities for continuous multimodal perception and reasoning. Full-duplex interaction, however, introduces a largely orthogonal control problem: as the user and agent dynamically share the conversational floor, the system must continuously determine how to engage. This motivates decoupling interaction control from the underlying perception and generation backbone, preserving existing model capabilities while enabling full-duplex interaction without retraining the backbone. Following this principle, we introduce a plug-and-play *Duplex Controller* that can be integrated with different offline or streaming models.

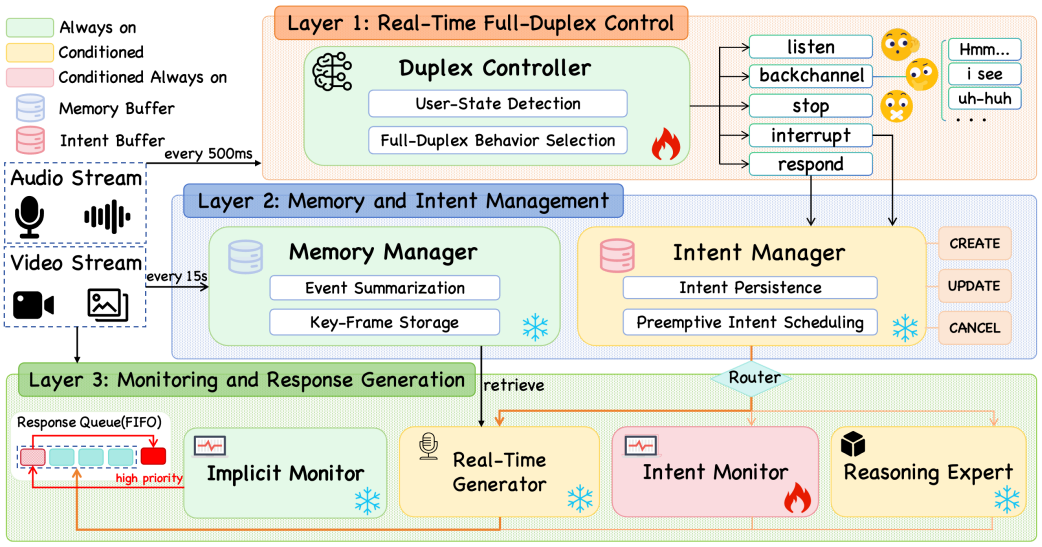

Figure 2: The framework of **StreamPro-Duplex**, organized into three coordinated layers. Layer 1 uses the *Duplex Controller* for user state detection and full-duplex action selection. Layer 2 maintains long-term event memory and persistent user intents through the *Memory Manager* and *Intent Manager*. Layer 3 employs specialized modules to handle different types of response requests arising during interaction, with their outputs coordinated through a shared response queue.

At every 500ms streaming chunk t, the controller receives the current audio observation x_t^{a}, the recent interaction history $\mathcal{H}_{t-1}$, and the system's streaming response state at the previous time step r_{t-1}. Given these inputs, the controller generates a structured output sequence consisting of a user state token s_t, an action token a_t, and, when required, the corresponding ASR transcription z_t:

$$\mathbf{y}_t = [s_t,\, a_t,\, z_t]\,. \tag{1}$$

The user state token is selected from $\mathcal{S} = \{\mathrm{listen}, \mathrm{speak}, \mathrm{pause}, \mathrm{backchannel}, \mathrm{interrupt}\}$, capturing the user's current conversational behavior. The full-duplex control action token is selected from $\mathcal{A} = \{\mathrm{none}, \mathrm{backchannel}, \mathrm{stopinterrupt}, \mathrm{respond}, \}$, corresponding respectively to maintaining the current interaction, initiating a normal response, providing listener feedback, triggering immediate intervention in an urgent situation, or terminating the ongoing streaming response. When the user state transitions from an active speaking state to listen, signaling the completion of the current turn, z_t contains the full transcription of the completed utterance. Otherwise, no ASR tokens are generated.

Group-Wise Decision Loss. The Duplex Controller emits two groups of decision tokens: user state predictions and full-duplex action selections. Common decisions such as listen and none occur much more frequently than rare but critical decisions such as $\mathrm{interrupt}$. For human-like full-duplex interaction, however, assistant-initiated backchanneling and interruption are also essential. Directly optimizing the token-level likelihood can therefore bias the controller toward dominant classes. We apply effective-number reweighting independently within the state and action groups, together with optional per-class multipliers for selected decisions. All other tokens retain unit weight.

For each group $G \in \{\mathcal{S}, \mathcal{A}\}$, let n_c denote the frequency of class c and let C_G be the set of observed classes. The normalized class-balanced weight is

$$\alpha_c^{(G)} = \frac{|C_G|(1-\beta)/(1-\beta^{n_c})}{\sum_{c' \in C_G} (1-\beta)/(1-\beta^{n_{c'}})}, \tag{2}$$

where β controls the strength of class-frequency balancing. Let $m_c^{(G)}$ denote an optional multiplier applied to class c after class-balanced normalization, with $m_c^{(G)} < 1$ for decisions that should be

down-weighted. The loss is

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} w_i \left[-\log p_\theta(y_i \mid y_{<i}, \mathbf{x})\right], \qquad w_i = \begin{cases} m_{y_i}^{(\mathcal{S})}\alpha_{y_i}^{(\mathcal{S})}, & y_i \in \mathcal{S}, \\ m_{y_i}^{(\mathcal{A})}\alpha_{y_i}^{(\mathcal{A})}, & y_i \in \mathcal{A}, \\ 1, & \text{otherwise.} \end{cases} \tag{3}$$

For more training details, please refer to the Appendix B.1.

3.2 Layer 2: Memory and Intent Management

For always-on assistants that continuously interact with users over extended periods, effective management of the evolving video stream and persistent user intents is essential for maintaining interaction continuity and supporting long-term task execution. To this end, we design two dedicated modules: a *Memory Manager* for maintaining continuous video memory and an *Intent Manager* for tracking and coordinating multiple evolving user intents.

Streaming Video Memory Management. The Memory Manager runs asynchronously from the other modules. At each memory update step t, the Memory Manager receives a sliding video window V_t and the existing memory bank $\mathcal{M}_{t-1}$, producing a sequence of `CREATE` and `UPDATE` operations to maintain a *multi-granular* memory bank:

$$\mathcal{O}_t = f_{\text{mem}}\left(V_t, \mathcal{M}_{t-1}\right), \qquad \mathcal{O}_t = \{o_t^{(k)}\}_{k=1}^{K_t}, \quad o_t^{(k)} \in \{\texttt{CREATE}, \texttt{UPDATE}\}, \tag{4}$$

where f_{mem} denotes the Memory Manager, $\mathcal{O}_t$ is the sequence of memory operations at step t, and K_t denotes the number of generated operations. Each event record contains a concise summary, key entities, and key frames, enabling the system to preserve long-term visual context.

Persistent Multi-Intent Management. The Intent Manager maintains a persistent set of user intents across the interaction. When the Duplex Controller produces a new user utterance, the Intent Manager receives the utterance u_t and the existing intent set $\mathcal{I}_{t-1}$, producing a sequence of `CREATE`, `UPDATE`, and `CANCEL` operations:

$$\Gamma_t = f_{\text{int}}\left(u_t, \mathcal{I}_{t-1}\right), \qquad \Gamma_t = \{\gamma_t^{(j)}\}_{j=1}^{N_t}, \quad \gamma_t^{(j)} \in \{\texttt{CREATE}, \texttt{UPDATE}, \texttt{CANCEL}\}, \tag{5}$$

where f_{int} denotes the Intent Manager, Γ_t is the sequence of intent operations at step t, and N_t denotes the number of generated operations. When active intents become actionable, their responses are inserted into a response queue and normally delivered in FIFO order according to their trigger time. For simultaneous triggers, we prioritize *safety-critical events* $>$ *real-time requests* $>$ *proactive intents*, while only safety-critical events may preempt an ongoing response.

3.3 Layer 3: Monitoring and Response Generation

Building on interaction control and persistent intent states, this layer consists of four specialized modules—the *Intent Monitor*, *Implicit Monitor*, *Real-Time Generator*, and *Reasoning Expert*—to handle different response needs in always-on streaming interaction.

Generation Routing and Execution Consistency. When the Duplex Controller detects a new user intent, it is first processed by the Intent Manager and then routed to the appropriate downstream module according to its execution requirements: real-time requests are handled by the Real-Time Generator, complex requests requiring additional reasoning or tool use by the Reasoning Expert, and persistent proactive intents by the Intent Monitor. To ensure consistency under asynchronous execution, each intent is associated with a version identifier. When an intent is updated or cancelled, responses generated from outdated versions are invalidated before delivery.

Unified Multi-Intent Proactive Monitoring. In always-on streaming interaction, users may register multiple proactive intents that remain active over time and must be jointly monitored against the same incoming stream. A straightforward approach is to instantiate a separate monitoring process for each proactive intent (Zhang et al., 2026). However, as the number of active intents grows, this design requires repeated inference over largely shared streaming context, resulting in substantial

computational redundancy and increased inference latency. To address this limitation, we develop a unified *Intent Monitor* that jointly tracks multiple proactive intents and evaluates their triggering conditions within a single forward pass at each streaming step, trained with the *CB-Stream Loss* introduced in StreamPro (Li et al., 2026). For more training details, please refer to the Appendix B.2.

At each 500ms streaming chunk t, the Intent Monitor receives the current timestamp τ_t, video frame v_t, interaction history $\mathcal{H}_{t-1}$, and intent operations Δ_t, where each operation is encoded as <task$k$><op> *instruction* with op $\in \{\texttt{CREATE}, \texttt{UPDATE}, \texttt{DELETE}\}$ and DELETE requiring no instruction. It jointly evaluates all active intents within a single forward pass:

$$y_t = f_{\text{mon}}\left(\tau_t, v_t, \mathcal{H}_{t-1}, \Delta_t\right), \qquad y_t \in \{\texttt{<silence>}, \texttt{<response>}\,\mathcal{R}_t\,\texttt{</response>}\} \tag{6}$$

where f_{mon} denotes the unified Intent Monitor and $\mathcal{R}_t$ contains the responses associated with all intents triggered at step t. The monitor outputs <silence> when no active intent is triggered; otherwise, it returns all triggered responses within a single <response> block.

Task-Agnostic Intent Detection. In always-on streaming interaction, opportunities for timely assistance do not always originate from explicit user intents. The evolving environment itself may reveal situations where user support is needed, even without a predefined request. To capture such opportunities, we introduce an *Implicit Monitor* that monitors the context and directly assists the user when it identifies a need for *immediate intervention*, *error correction*, or *operational assistance*.

3.4 Parallel Inference Framework

To reduce end-to-end latency, StreamPro-Duplex adopts a two-level parallel inference design: **(1) Parallel Stream Processing.** While continuously receiving audio-visual inputs, the system maintains four independently scheduled processing branches: the Duplex Controller operates continuously for real-time interaction control; the Intent Monitor is activated when active proactive intents exist; the Implicit Monitor periodically inspects the streaming context every 2 seconds; and the Memory Manager updates long-term context every 15 seconds. These branches execute in parallel whenever triggered, avoiding unnecessary continuous inference for all modules. **(2) Asynchronous Response Generation.** Once an actionable intent is routed to the Real-Time Generator or Reasoning Expert, response generation is executed asynchronously, while the foreground pipeline continues receiving and processing incoming audio-visual streams. This decouples response generation from continuous stream processing and prevents generation latency from blocking ongoing perception and interaction. More details on the overall system execution are provided in the Appendix B.3

4 StreamInteract-138K

To support streaming interaction training, We construct **StreamInteract-138K**, a large-scale dataset designed to support natural full-duplex communication and persistent multi-intent interaction.

4.1 Full-Duplex Interaction Synthesis

As shown in Figure 3, we construct full-duplex interaction data under three settings: *text-only conversation*, *real-time interaction*, and *proactive interaction*. For text-only conversations, we synthesize diverse dialogues conditioned on scenarios and topics, and inject full-duplex behaviors to form temporally aligned two-channel interactions. For video-grounded settings, real-time and proactive data share a temporal backbone built from timestamped video events. We generate context-grounded real-time dialogues and proactive tasks with refined trigger and response timings, and further inject 18 types of full-duplex behaviors. Finally, we perform audio–visual alignment and timestamp validation to ensure temporal consistency.

4.2 Multi-Intent Interaction Construction

We construct multi-intent interactions covering two representative settings: *interleaved real-time–proactive interaction* and *multiple concurrent proactive intents*. The former is naturally covered by our proactive full-duplex data, where real-time requests may occur while proactive intents remain active. For the latter, we compose multiple proactive intents into the same trajectory and introduce intent-level *creation*, *update*, and *removal* operations. This construction creates overlapping and evolving

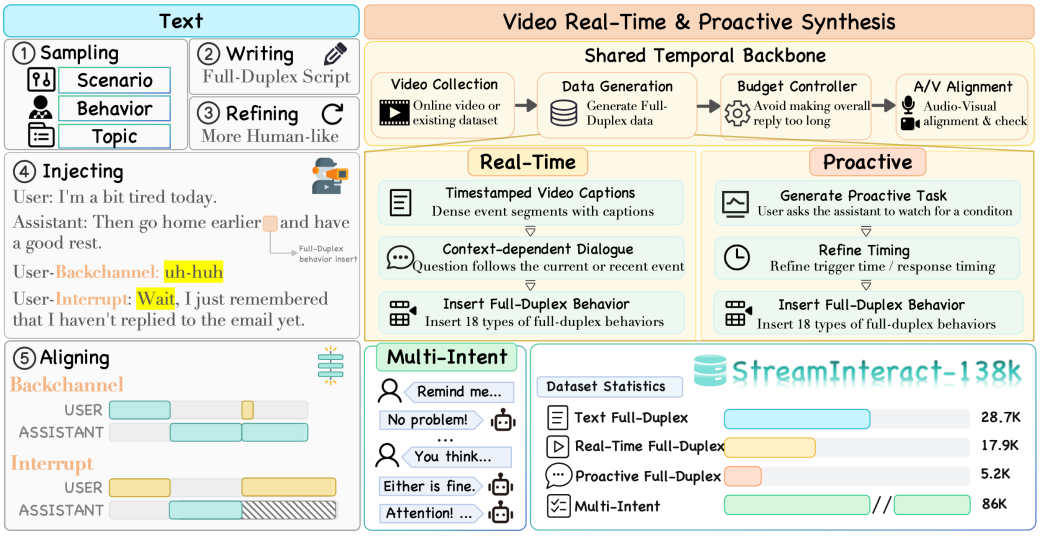

Figure 3: StreamInteract-138K data pipeline. We construct the dataset from two dimensions: full-duplex interaction, covering text-only, real-time, and proactive scenarios, and multi-intent interaction, covering *interleaved real-time–proactive interaction* and *multiple concurrent proactive intents.*

intent states, enabling the model to learn persistent multi-intent management over long-running interactions. For further details on data construction, please refer to Appendix C.

5 Experiment

5.1 Settings

Benchmarks. We evaluate StreamPro-Duplex across three distinct categories of tasks:

- **Duplex Control and Speech Recognition.** We evaluate the Duplex Controller on the *User Interruption* and *Listener Backchannel* categories of Full-Duplex-Bench v1.5 (Lin et al., 2026a), and further assess its ASR performance on LibriSpeech (Panayotov et al., 2015), as recognition quality directly affects downstream intent understanding. We compare against representative full-duplex, speech recognition, and omni-modal models (Hurst et al., 2024; Wang et al., 2024a; Défossez et al., 2024; Bai et al., 2026; 2024; Radford et al., 2023; Tang et al., 2024; OpenBMB, 2026; Team & Group, 2026; Xu et al., 2025a;b; Huang et al., 2026).
- **Proactive Streaming Tasks.** We evaluate the proactive capabilities of the Intent Monitor on StreamPro-Bench (Li et al., 2026), which covers three categories of proactive tasks: *Perceptual Understanding (PU)*, *Temporal Reasoning (TR)*, and *Anticipatory Assistance (AA)*.
- **Real-time Streaming Tasks.** We evaluate the real-time streaming capabilities of the StreamPro-Duplex framework on Real-Time Visual Understanding (RTVU) from StreamingBench (Lin et al., 2026b), as well as Real-Time Visual Perception (RTVP) from OVO-Bench (Niu et al., 2025). For a fair comparison, we evaluate StreamPro-Duplex using only its Memory Manager and Real-Time Generator modules; under this setting, the entire evaluation pipeline is ***training-free***.

Implementation Details. We use Qwen3.6-35B-A3B as the backbone for the Intent Manager, Memory Manager, Implicit Monitor, and Real-Time Generator. The Duplex Controller is trained from Qwen3-Omni-30B-A3B (Xu et al., 2025b), while the Intent Monitor is trained from Qwen3.6-35B-A3B. For both the Duplex Controller and Intent Monitor, we use a batch size of 64 and a learning rate of 1×10^{-5}, while only the LLM parameters are optimized during training. For the Group-Wise Decision Loss, we set $\beta = 0.99$ and apply additional action-level multipliers of $m^{(\mathcal{A})}_{\text{backchannel}} = 0.4$ and $m^{(\mathcal{A})}_{\text{interrupt}} = 0.5$. For the CB-Stream Loss, we follow the original configuration of StreamPro (Li et al., 2026). To reduce inference latency, we adopt a sliding-window strategy with 500-ms streaming chunks. The Duplex Controller processes only the most recent 180 audio chunks; the Intent Monitor and Implicit Monitor use the interaction history together with the most recent 60 visual chunks; and

Table 1: Duplex Controller performance on Full-Duplex-Bench v1.5 (Lin et al., 2026a).

Method	User Interruption		User Backchannel	
	C^{+}_{respond}	C^{-}_{resume}	C^{-}_{respond}	C^{+}_{resume}
Gemini 3.1 Live	0.77	0.20	0.02	0.95
GPT-4o	0.78	0.10	0.03	0.70
Freeze-Omni	0.72	0.12	0.07	0.80
Moshi	0.50	0.26	0.02	0.06
MiniCPM-o 4.5	0.60	0.36	**0.00**	0.95
Joy-Duplex	0.88	0.07	0.01	**0.96**
Realtime-Venus-Omni	0.60	0.37	0.02	0.92
Duplex Controller	**0.99**	**0.01**	0.09	0.91

Notes. $^{+}$ / $^{-}$: higher / lower is better. WER: lower is better.

Table 2: Duplex Controller performance on LibriSpeech (Panayotov et al., 2015).

Method	WER (%)
Whisper-large-v3	1.80
Seed-ASR-Multilingual	1.60
SALMONN	2.10
MiniCPM-o	1.70
Qwen2.5-Omni-7B	1.80
Qwen3-Omni-30B-A3B	**1.22**
Gemini-3.1-Flash-Live	4.10
DuplexOmni	11.92
Duplex Controller	2.15

Table 3: Intent Monitor performance on proactive tasks across StreamPro-Bench (Li et al., 2026). PU: Perceptual Understanding, TR: Temporal Reasoning, AA: Anticipatory Assistance.

Method	StreamPro-Bench									
	PU			TR			AA			Overall F1
	Time	Sem.	F1	Time	Sem.	F1	Time	Sem.	F1	Avg.
Qwen2.5-VL-7B (Bai et al., 2025b)	7.5	7.7	3.4	2.4	0.4	0.1	6.5	2.2	0.9	1.5
Qwen3-VL-8B (Bai et al., 2025a)	60.1	51.4	6.6	27.6	16.6	0.8	23.0	12.2	2.6	3.3
VideoLLM-Online (Chen et al., 2024)	17.0	1.6	0.4	0.5	0.1	0.0	21.8	7.0	5.7	2.0
MMDuet (Wang et al., 2024b)	47.4	33.1	9.3	71.8	20.3	1.5	34.8	15.7	4.0	4.9
MMDuet2 (Wang et al., 2025)	37.2	28.5	9.8	47.2	30.1	3.1	22.3	7.4	2.7	5.2
Streamo (Xia et al., 2025)	20.3	17.2	7.8	27.2	19.1	14.0	6.0	2.6	2.7	8.2
MiniCPM-o-4.5 (OpenBMB, 2026)	20.6	16.9	13.5	5.2	0.8	0.6	6.6	4.9	3.8	6.0
StreamPro-SFT (Li et al., 2026)	66.6	61.7	24.7	45.1	32.5	27.9	46.1	20.5	5.7	19.8
Intent Monitor	65.1	55.1	**37.8**	54.1	47.6	**42.9**	30.2	16.8	**8.5**	**29.7**

the Real-Time Generator and Reasoning Expert additionally retrieve the top-3 relevant events from the memory bank to support long-range understanding.

5.2 Main Result

Duplex Control and Speech Recognition. As shown in Table 1, the Duplex Controller demonstrates strong full-duplex interaction capabilities. For *User Interruption*, it achieves the highest interruption response rate, demonstrating reliable interruption handling. For *User Backchannel*, it maintains a competitive continuation rate compared with recent full-duplex omni-modal models such as Realtime-Venus (Team & Group, 2026), indicating effective conversational-floor maintenance. As shown in Table 2, full-duplex streaming training increases the WER from 1.22 to 2.15, reflecting the performance trade-off incurred by streaming adaptation. Nevertheless, the Duplex Controller remains competitive in ASR performance.

Proactive Streaming Tasks. As shown in Table 3, the Intent Monitor achieves strong performance on StreamPro-Bench, improving the overall F1 from 19.8 to 29.7 over StreamPro-SFT (Li et al., 2026). These results demonstrate the strong and consistent proactive capabilities of the Intent Monitor across diverse proactive streaming tasks.

Real-Time Streaming Tasks. As shown in Table 4, StreamPro-Duplex achieves strong performance on both StreamingBench-RTVU and OVO-Bench-RTVP without additional training for real-time tasks. It obtains average scores of 82.1 and 75.4, respectively, approaching the performance of recent training-based streaming models such as MOSS-VL-Realtime-11B (Wang et al., 2026), while outperforming most other streaming baselines. These results show that the proposed framework preserves strong real-time streaming understanding capabilities even without task-specific adaptation. For additional experiments and analyses, please refer to Appendix A.

Table 4: Evaluation results on real-time tasks in StreamingBench and OVO-Bench. Despite being ***training-free***, StreamPro-Duplex achieves performance comparable to SOTA training-based methods.

Method	StreamingBench-RTVU											OVO-Bench-RTVP						
	OP	CR	CS	ATP	EU	TR	PR	SU	ACP	CT	Avg.	OCR	ACR	ATR	STU	FPD	OJR	Avg.
VideoLLM-Online (Chen et al., 2024)	39.1	40.1	34.5	31.1	45.5	32.4	31.5	34.2	42.5	27.9	35.9	8.1	23.9	12.1	14.0	45.5	21.2	20.8
Dispider (Qian et al., 2025)	74.9	75.5	74.1	73.1	74.4	59.9	76.1	62.9	62.2	45.8	67.9	57.7	49.5	62.1	44.9	61.4	51.6	54.5
TimeChat-Online (Yao et al., 2025)	80.8	79.7	80.8	83.3	74.8	78.8	78.7	64.2	68.8	**58.0**	74.8	69.8	48.6	64.7	44.9	68.3	55.4	58.6
StreamForest (Zeng et al., 2025)	83.1	82.8	82.7	84.3	77.5	78.2	76.9	69.1	75.6	54.4	76.5	68.5	53.2	71.6	47.8	65.4	60.9	61.2
Streamo (Xia et al., 2025)	-	-	-	-	-	-	-	-	-	-	-	79.2	57.8	**75.0**	49.4	64.4	70.1	66.0
StreamPro-SFT (Li et al., 2026)	86.7	78.9	88.0	84.9	**84.3**	**86.0**	82.4	74.0	78.7	33.0	79.3	79.2	68.8	69.0	53.9	66.3	71.7	67.9
MOSS-VL-Realtime (Wang et al., 2026)	-	-	-	-	-	-	-	-	-	-	**82.9**	-	-	-	-	-	-	**75.9**
StreamPro-Duplex (Training-free)	**87.3**	**89.8**	**93.4**	**88.9**	84.1	81.9	**83.3**	**75.1**	**81.8**	44.7	82.1	**85.9**	**76.2**	73.3	**62.9**	**75.3**	**78.8**	75.4

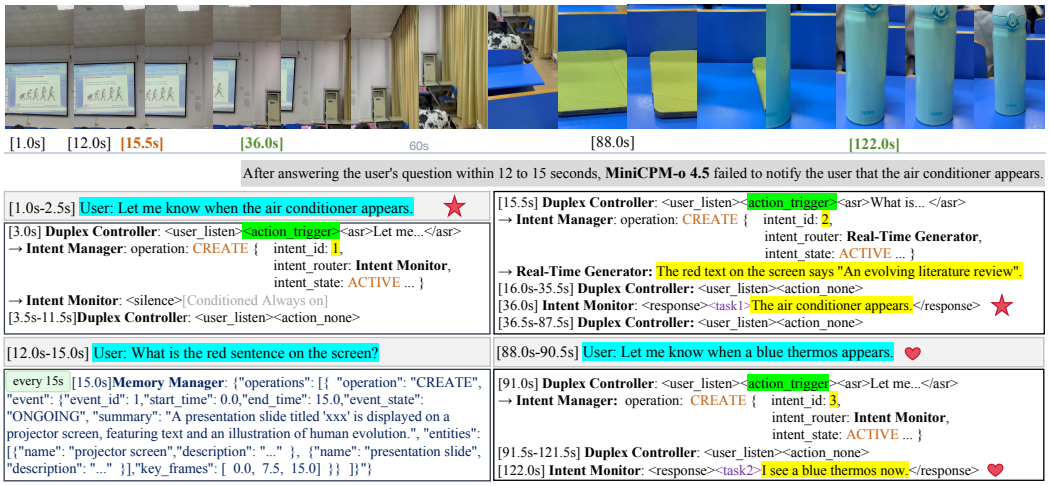

Figure 4: Case study of interleaved real-time–proactive interaction. StreamPro-Duplex preserves pending proactive intents while answering an intervening real-time question, continues monitoring the evolving visual stream, and triggers only when its corresponding visual condition is satisfied.

5.3 Case Study

Figure 4 shows how StreamPro-Duplex coordinates interleaved real-time and proactive intents. At 1.0 s, the user registers an air-conditioner notification, which the Intent Manager routes to the Intent Monitor. At 12.0 s, a separate question about the red sentence is routed to the Real-Time Generator and answered at 15.5 s, while the first intent remains active. The updated visual memory then triggers the air-conditioner reminder at 36.0 s. A later request for a blue thermos is created at 88.0 s and fulfilled at 122.0 s. In contrast, MiniCPM-o 4.5 (OpenBMB, 2026) answers the intervening question but does not return to the original proactive request. The case therefore demonstrates that real-time responses and multiple longer-horizon intents can be coordinated within one continuous stream. More cases are provided in the Appendix D.

6 Conclusion

In this work, we propose **StreamPro-Duplex**, a full-duplex multimodal agent framework for always-on streaming interaction. To move beyond isolated interaction, StreamPro-Duplex integrates a plug-and-play *Duplex Controller* for real-time interaction control, persistent multi-intent management for maintaining evolving user needs, and a unified *Intent Monitor* that jointly monitors multiple proactive intents over a shared streaming context. We further construct *StreamInteract-138K*, covering representative full-duplex and multi-intent interaction scenarios for model training. Extensive experiments demonstrate that StreamPro-Duplex achieves strong performance across full-duplex interaction, proactive streaming tasks and real-time streaming understanding. Case studies further show that StreamPro-Duplex can maintain long-horizon proactive intents while handling interleaved real-time requests within a continuous stream. We hope this work can contribute to the development of more natural and capable always-on multimodal agents.

AI USE STATEMENT

We used generative AI tools to assist in constructing StreamInteract-138K, including synthesizing full-duplex dialogues and rewriting proactive task instructions to create evolving multi-intent interactions. Appendix C describes the data construction procedure, including the use of an LLM to rewrite task requirements. The pipeline includes data filtering, audio–visual alignment, and timestamp validation to check the consistency of synthesized interactions. We also used generative AI tools to assist with language editing, LaTeX formatting, and polishing. We take responsibility for the final content of our work, including its text, claims, code, data, and other artifacts produced with the assistance of generative AI.

ETHICS STATEMENT

Our work studies always-on audio–visual interaction with persistent memory and intent tracking. These capabilities can expose sensitive information about users and bystanders and could be misused for non-consensual monitoring. StreamInteract-138K builds on existing publicly available data and synthesized interactions; no new human-subject data are collected. Any redistribution of source material or derived data must respect the original licenses, access conditions, and privacy constraints. Synthetic dialogues and pretrained models may reproduce biases or generate inaccurate responses, and the evaluated benchmarks do not establish reliability across all languages, user groups, or deployment settings. The framework is not validated as a safety-critical warning system. Deployment would require informed user consent, appropriate safeguards for bystanders, controls to pause sensing and delete retained memories, data minimization, and human oversight. We intend the framework for research and caution against privacy-sensitive or safety-critical deployment without additional validation and safeguards.

REPRODUCIBILITY STATEMENT

Section 3 describes the framework and the Duplex Controller's training objective. Appendix B specifies the training data, context-window construction, Intent Monitor objective, and streaming inference procedure, including memory retrieval and output scheduling. Appendix C describes the data sources and interaction synthesis procedure for StreamInteract-138K, while Appendix E provides the system prompts for the Duplex Controller and Intent Monitor. The experiment section identifies the evaluated benchmarks, backbone models, and loss settings; Appendix A reports the action-multiplier study and the latency measurement setup, including the serving framework and GPU type. We will release StreamInteract-138K, code, prompts, configuration files, and model checkpoints, subject to the source data licenses and privacy constraints.

REFERENCES

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

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025b. URL https://arxiv.org/abs/2502.13923.

Ye Bai, Jingping Chen, Jitong Chen, Wei Chen, Zhuo Chen, Chuang Ding, Linhao Dong, Qianqian Dong, Yujiao Du, Kepan Gao, et al. Seed-asr: Understanding diverse speech and contexts with llm-based speech recognition. *arXiv preprint arXiv:2407.04675*, 2024.

Yinhao Bai, Jinming Chen, Yafeng Chen, Wei Deng, Boya Dong, Nan Duan, Yu Gu, Weisheng Han, Yankun Huang, Ming Ke, et al. Joyai-talker: Full-duplex speech interactive large model built for empathetic voice agents. *arXiv preprint arXiv:2608.01119*, 2026.

Hui Bu, Jiayu Du, Xingyu Na, Bengu Wu, and Hao Zheng. Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline. In *2017 20th conference of the oriental chapter of the international coordinating committee on speech databases and speech I/O systems and assessment (O-COCOSDA)*, pp. 1–5. IEEE, 2017.

Guoguo Chen, Shuzhou Chai, Guanbo Wang, Jiayu Du, Wei-Qiang Zhang, Chao Weng, Dan Su, Daniel Povey, Jan Trmal, Junbo Zhang, et al. Gigaspeech: An evolving, multi-domain asr corpus with 10,000 hours of transcribed audio. *arXiv preprint arXiv:2106.06909*, 2021.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18407–18418, 2024.

Alexandre Défossez, Laurent Mazaré, Manu Orsini, Amélie Royer, Patrick Pérez, Hervé Jégou, Edouard Grave, and Neil Zeghidour. Moshi: a speech-text foundation model for real-time dialogue. *arXiv preprint arXiv:2410.00037*, 2024.

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. *arXiv preprint arXiv:2606.09186*, 2026.

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

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

Guan-Ting Lin, Shih-Yun Shan Kuan, Qirui Wang, Jiachen Lian, Tingle Li, Shinji Watanabe, and Hung-yi Lee. Full-duplex-bench v1. 5: Evaluating overlap handling for full-duplex speech models. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 19447–19451. IEEE, 2026a.

Junming Lin, Zheng Fang, Chi Chen, Haoxuan Cheng, Zihao Wan, Fuwen Luo, Ziyue Wang, Peng Li, Yang Liu, and Maosong Sun. Streamingbench: Assessing the gap for mllms to achieve streaming video understanding. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 12147–12151. IEEE, 2026b.

Xudong Lu, Yang Bo, Jinpeng Chen, Shuhan Li, Xintong Guo, Huankang Guan, Fang Liu, Dunyuan Xu, Peiwen Sun, Heyang Sun, et al. Aura: Always-on understanding and real-time assistance via video streams. *arXiv preprint arXiv:2604.04184*, 2026a.

Xudong Lu, Xueying Li, Annan Wang, Yang Bo, Jinpeng Chen, Zengliang Li, Nianzu Yang, Rui Liu, Xue Yang, Jingwen Hou, et al. Omniinteract: Benchmarking real-world streaming interaction for real-time omnimodal assistants. *arXiv preprint arXiv:2605.26485*, 2026b.

Junbo Niu, Yifei Li, Ziyang Miao, Chunjiang Ge, Yuanhang Zhou, Qihao He, Xiaoyi Dong, Haodong Duan, Shuangrui Ding, Rui Qian, et al. Ovo-bench: How far is your video-llms from real-world online video understanding? In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 18902–18913, 2025.

OpenBMB. Minicpm-o 4.5 technical report. `https://github.com/OpenBMB/MiniCPM-o/blob/main/docs/MiniCPM_o_45_technical_report.pdf`, 2026. GitHub technical report.

Orantqing, Shengpeng Ji, Junlong Tong, Jialong Zuo, Dongjie Fu, Di Cao, Yangzhuo Li, Shangda Wu, Franz, Evan, Theron Veyra, Changhao Pan, Jingyu Lu, Dongchao Yang, Zhifei Xie, Yang Tan, Xiaoyu Shen, Xiaoda Yang, Wenfu Wang, Teddy Sun, Steve Yves, and Zhou Zhao. Multimodal duplex interaction agent, 2026. URL `https://arxiv.org/abs/2609.08977`.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 5206–5210. IEEE, 2015.

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*, pp. 24045–24055, 2025.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International conference on machine learning*, pp. 28492–28518. PMLR, 2023.

Changli Tang, Wenyi Yu, Guangzhi Sun, Xianzhao Chen, Tian Tan, Wei Li, Lu Lu, Zejun Ma, and Chao Zhang. Salmonn: Towards generic hearing abilities for large language models, 2024. URL https://arxiv.org/abs/2310.13289.

Venus Team and Ant Group. Realtime-venus: A full-duplex interaction system with asynchronous delegation, 2026. URL https://arxiv.org/abs/2609.13814.

Pengyu Wang, Chenkun Tan, Shaojun Zhou, Qirui Zhou, Yanxin Chen, Xingyang He, Huazheng Zeng, Jijun Cheng, Chenghao Wang, Xiaomeng Qian, Pengfei Wang, Zhan Huang, Shanqing Gao, Wei Huang, Longjun Cao, Wu Ran, Jie Liu, Changtai Zhu, Hongkai Wang, Yixian Tian, Chenghao Liu, Zhen Ye, Xinghao Wang, Botian Jiang, Guoguo Feng, Zhaoye Fei, Ruixiao Li, Mingshu Chen, Yang Gao, Qinyuan Cheng, Shimin Li, and Xipeng Qiu. Moss-vl technical report, 2026. URL https://arxiv.org/abs/2608.15045.

Xiong Wang, Yangze Li, Chaoyou Fu, Yunhang Shen, Lei Xie, Ke Li, Xing Sun, and Long Ma. Freeze-omni: A smart and low latency speech-to-speech dialogue model with frozen llm. *arXiv preprint arXiv:2411.00774*, 2024a.

Yueqian Wang, Xiaojun Meng, Yuxuan Wang, Jianxin Liang, Jiansheng Wei, Huishuai Zhang, and Dongyan Zhao. Videollm knows when to speak: Enhancing time-sensitive video comprehension with video-text duet interaction format. *arXiv preprint arXiv:2411.17991*, 1(3):5, 2024b.

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. *arXiv preprint arXiv:2512.06810*, 2025.

Jiaer Xia, Peixian Chen, Mengdan Zhang, Xing Sun, and Kaiyang Zhou. Streaming video instruction tuning. *arXiv preprint arXiv:2512.21334*, 2025.

Jin Xu, Zhifang Guo, Jinzheng He, Hangrui Hu, Ting He, Shuai Bai, Keqin Chen, Jialin Wang, Yang Fan, Kai Dang, Bin Zhang, Xiong Wang, Yunfei Chu, and Junyang Lin. Qwen2.5-omni technical report, 2025a. URL https://arxiv.org/abs/2503.20215.

Jin Xu, Zhifang Guo, Hangrui Hu, Yunfei Chu, Xiong Wang, Jinzheng He, Yuxuan Wang, Xian Shi, Ting He, Xinfa Zhu, et al. Qwen3-omni technical report. *arXiv preprint arXiv:2509.17765*, 2025b.

Ruyi Xu, Guangxuan Xiao, Yukang Chen, Liuning He, Kelly Peng, Yao Lu, and Song Han. Streamingvlm: Real-time understanding for infinite video streams. *arXiv preprint arXiv:2510.09608*, 2025c.

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.

Linli Yao, Yicheng Li, Yuancheng Wei, Lei Li, Shuhuai Ren, Yuanxin Liu, Kun Ouyang, Lean Wang, Shicheng Li, Sida Li, et al. Timechat-online: 80% visual tokens are naturally redundant in streaming videos. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 10807–10816, 2025.

Xiangyu Zeng, Kefan Qiu, Qingyu Zhang, Xinhao Li, Jing Wang, Jiaxin Li, Ziang Yan, Kun Tian, Meng Tian, Xinhai Zhao, et al. Streamforest: Efficient online video understanding with persistent event memory. *arXiv preprint arXiv:2509.24871*, 2025.

Xichen Zhang, Guankai Li, Yinghao Zhu, Shijian Wang, Sitong Wu, Shaozuo Yu, Meng Chu, Yuan Lu, and Jiaya Jia. Streamarena: Toward continuous, interactive, and long-horizon agentic streaming video understanding, 2026. URL https://arxiv.org/abs/2608.05703.

APPENDICES

A ADDITIONAL EXPERIMENTS AND ANALYSIS

A.1 HYPERPARAMETER STUDY OF GROUP-WISE DECISION LOSS

While class-balanced reweighting encourages the controller to learn rare full-duplex behaviors, class balancing alone does not guarantee an appropriate activation frequency at inference time. Behaviors such as backchanneling and interruption should be triggered when appropriate, whereas overly frequent activation may disrupt the natural conversational flow. To quantify this effect, we report the Backchannel Trigger Count (BTC) and Interrupt Trigger Count (ITC), defined as the total numbers of streaming inference chunks on Full-Duplex-Bench v1.5 in which the controller predicts `backchannel` and `interrupt`, respectively. As shown in Table 5, under standard CE training, the Duplex Controller triggers only 1 backchannel and no interruption. In contrast, directly applying the Group-Wise Decision (GWD) Loss without additional action-level calibration substantially increases BTC and ITC to 53 and 20, respectively. This indicates that, although GWD effectively alleviates the dominance of frequent actions and improves the learning of rare decisions, it may also over-correct their activation frequencies at inference time.

We therefore introduce action-level multipliers $m^{(\mathcal{A})}_{\text{backchannel}}$ and $m^{(\mathcal{A})}_{\text{interrupt}}$ to explicitly calibrate the activation frequencies of backchanneling and interruption after class-balanced reweighting. Setting them to 0.4 and 0.5, respectively, reduces BTC and ITC to 5 and 15, while achieving the best performance on the User Backchannel subset, with $C^{-}_{\text{respond}} = 0.09$ and $C^{+}_{\text{resume}} = 0.91$. Further decreasing the multipliers to 0.2 and 0.25 reduces the number of backchannel activations to only 2, but does not yield further performance gains; instead, the corresponding scores degrade to 0.14 and 0.86. These results show that the action-level multipliers effectively suppress excessive activation induced by class balancing while preserving sufficient sensitivity to rare full-duplex behaviors. We therefore use $m^{(\mathcal{A})}_{\text{backchannel}} = 0.4$ and $m^{(\mathcal{A})}_{\text{interrupt}} = 0.5$ in our final model.

Table 5: Ablation study of the loss function and hyperparameters.

Loss	**Action Weight**		**User Interruption**		**Listener Backchannel**		**BTC**	**ITC**
	$m^{(\mathcal{A})}_{\text{BC}}$	$m^{(\mathcal{A})}_{\text{INT}}$	C^{+}_{respond}	C^{-}_{resume}	C^{-}_{respond}	C^{+}_{resume}		
CE	–	–	0.99	0.01	0.12	0.88	1	0
GWD	1.0	1.0	0.99	0.01	0.14	0.86	53	20
GWD	**0.4**	**0.5**	**0.99**	**0.01**	**0.09**	**0.91**	**5**	**15**
GWD	0.2	0.25	0.99	0.01	0.14	0.86	2	17

A.2 STREAMPRO-DUPLEX LATENCY ANALYSIS

To evaluate the practical inference latency of StreamPro-Duplex, we select 100 representative samples from OmniInteract (Lu et al., 2026b) that are compatible with our streaming interaction setting. For each sample, we execute the complete inference pipeline and record the latency of each module. Table 6 reports the average latency over these 100 samples. All models are served with vLLM on NVIDIA H100 GPUs. **Latency is measured under a re-prefill setting, where the input context is prefetched at each streaming step without KV-cache reuse across consecutive sliding windows.**

StreamPro-Duplex adopts a parallel inference architecture in which different modules are independently scheduled according to their execution requirements. The Duplex Controller forms the latency-critical foreground loop and runs every 500 ms, requiring only 277.1 ms on average. When a completed user intent is detected, the Intent Manager is additionally invoked for intent parsing and routing, with an average latency of 399.4 ms. Other stream-processing modules are activated

according to their own schedules or conditions and can execute in parallel with the foreground modules. The Memory Manager runs every 15 s with an average latency of 2882.5 ms, while the Intent Monitor is activated only when proactive intents are active and requires 1090.5 ms on average. Since concurrently triggered modules are executed in parallel, the processing latency is determined by the slowest active branch rather than the sum of their individual latencies.

Response generation is further decoupled from stream processing. The Real-Time Generator takes 1301.5 ms on average, but runs asynchronously after a request is routed to it. During generation, the system continues receiving and processing incoming audio-visual streams, and the generated response is written to the shared response queue once available. Therefore, generation latency does not block continuous audio-visual input processing. Overall, the parallel execution of stream-processing modules avoids unnecessary sequential latency, while asynchronous response generation further prevents generation from blocking continuous streaming interaction.

Table 6: Runtime characteristics and average inference latency of StreamPro-Duplex modules.

Module	**Scheduling**	**Trigger / Interval**	**Execution**	**Latency (ms)**
Duplex Controller	Periodic	500 ms	Foreground	277.1
Intent Manager	Event-driven	User request	Foreground	399.4
Intent Monitor	Conditional	Active proactive intents	Parallel branch	1090.5
Memory Manager	Periodic	15 s	Parallel branch	2882.5
Real-Time Generator	On-demand	Real-time request	Asynchronous	1301.5

B Details of StreamPro-Duplex

B.1 Duplex Controller Training Details

Training Data. We train the Duplex Controller on the full-duplex subset of StreamInteract-138K, which contains approximately 52K training samples. To balance interaction context and inference latency, the controller operates with a sliding context window of at most 180 audio chunks, corresponding to approximately 90 seconds of audio. To maintain consistency between training and streaming inference, long training sequences are partitioned into consecutive windows of up to 180 chunks. The partitioning is performed such that each chunk in the original sequence contributes to the prediction target exactly once, avoiding duplicated supervision across overlapping windows. In addition, to strengthen the controller's speech recognition capability under streaming inference, we incorporate 34K Chinese ASR samples from AISHELL-1 (Bu et al., 2017) and 19K English ASR samples from GigaSpeech (Chen et al., 2021) as additional training data.

B.2 Intent Monitor Training Details

Training Data. We train the Intent Monitor on approximately 86K samples from the Multiple Concurrent Proactive Intents subset of the multi-intent data in StreamInteract-138K. Each sample contains multiple proactive tasks that must be monitored concurrently over the same video stream. During the interaction, users may further create new tasks, remove existing ones, or update their monitoring conditions, causing the active intent set to evolve over time.

Similar to the Duplex Controller, the Intent Monitor operates with a sliding context window that retains the most recent 60 streaming chunks. Accordingly, long training sequences are partitioned into multiple windows so that the model is trained under the same bounded streaming context used at inference time. For the interaction history, we retain all user instructions throughout the sequence to preserve the complete set of intent creation, update, and removal operations, while keeping only the 10 most recent assistant responses to control the context length. This design allows the model to preserve persistent user intents while focusing on the most recent streaming observations and interaction feedback.

Such data teaches the model to jointly track multiple proactive intents within a shared streaming context, dynamically maintain their evolving states and conditions, and determine the appropriate response opportunity for each active intent.

Training Objective. Following StreamPro (Li et al., 2026), we train the model with the *CB-Stream Loss*, which alleviates the severe imbalance between response and silence decisions in proactive streaming data. Specifically, for each decision class $k \in \mathcal{S}$ with frequency n_k, its class-balanced weight is computed as

$$\hat{w}_k^{\mathrm{CB}} = \frac{\frac{1-\beta}{1-\beta^{n_k}}}{\sum_{j\in\mathcal{S}} \frac{1-\beta}{1-\beta^{n_j}}} \cdot |\mathcal{S}|, \tag{7}$$

where β controls the strength of class balancing. The final objective jointly reweights decision and language tokens as

$$\mathcal{L}_{\mathrm{CB}} = \frac{1}{N}\sum_{i=1}^{N} w_i^{\mathrm{CB}} \left[-\log p_\theta(y_i \mid y_{<i}, \mathbf{x})\right], \qquad w_i^{\mathrm{CB}} = \begin{cases} \hat{w}_{y_i}^{\mathrm{CB}}, & y_i \in \mathcal{S}, \\ \lambda_{\mathrm{text}}, & y_i \in \mathcal{T}, \end{cases} \tag{8}$$

where λ_{text} balances the optimization between decision and language tokens.

B.3 Details of Inference Pipeline

We describe the inference pipeline of StreamPro-Duplex for different types of streaming tasks.

Real-Time Intents. When the Duplex Controller predicts `trigger` or `interrupt`, it outputs the corresponding ASR transcription of the user's current speech. The transcription is first passed to the Intent Monitor, which determines the appropriate intent operation and identifies the intent associated with the current request. Besides, the intent is used to retrieve the top-3 most relevant events from the persistent memory bank. For each retrieved event, its textual summary and key frames are returned as long-term contextual evidence for response generation. These retrieved memories are combined with the most recent 60 streaming chunk and the interaction history associated with the current intent, and are jointly fed into the Real-Time Generator to produce the response. The generated response is placed into the response queue.

Proactive Intents. When the Intent Manager identifies a newly created intent as proactive, it forwards the intent to the Intent Monitor for continuous monitoring. The Intent Monitor jointly tracks the active proactive intents against the evolving streaming context and determines whether the response condition of each intent has been satisfied. Once a response opportunity is detected, the Intent Monitor generates the corresponding response and places it into the response queue for subsequent scheduling. Unlike real-time intents, proactive intents are typically conditioned on information that emerges in the future stream rather than on previously observed events. Therefore, they do not require retrieval of historical information.

Implicit Assistance. Implicit assistance does not originate from an explicit user request or a predefined proactive intent. Instead, the system continuously observes the evolving environment and determines whether a newly emerging situation warrants timely intervention or assistance. We focus on three types of high-value situations: *immediate intervention*, where an imminent risk requires an urgent warning; *error correction*, where a concrete ongoing mistake should be corrected; and *operational assistance*, where a newly visible situation has a clear and useful next step. **(1) Immediate Intervention.** This category targets situations involving imminent physical-safety, property, privacy, or irreversible-action risks that require the assistant to intervene immediately. The key criterion is urgency: delaying the response may lead to direct harm or an irreversible consequence. Typical cases include newly emerging hazards, dangerous operations, or destructive actions that are about to occur. The response should therefore be brief, direct, and action-oriented, clearly stating both the visible risk and the action the user should take immediately. **(2) Error Correction.** This category is used when the monitor observes a concrete and visually verifiable mistake that has just occurred or is currently being made. Unlike immediate intervention, the primary concern is not imminent danger, but preventing the user from continuing with an incorrect operation that is likely to produce an undesired result. The assistant should identify the specific mistake and provide the corresponding correction. Cases that merely reflect alternative valid strategies, minor inefficiencies, or uncertain interpretations are not considered error correction. **(3) Operational Assistance.** This category covers newly emerging situations in which the user is not necessarily making an error or facing an immediate hazard, but the current visual context reveals a clear and time-sensitive next step where a short prompt

can materially help the user proceed. The assistance should be grounded in visible evidence and provide a concrete actionable suggestion, rather than general advice or scene commentary. This category therefore complements the other two by addressing situations where timely guidance is useful even in the absence of explicit danger or error.

Response Queue and Streaming Response Strategy. As text-to-speech synthesis is not considered in the current framework, we simulate streaming responses to better approximate the temporal characteristics of natural human–human conversation. All responses generated by different modules are first placed into a shared response queue and delivered sequentially according to the queue order. Following Gander (Orantqing et al., 2026), instead of releasing the entire response at once, the system emits each queued response incrementally, with each streaming chunk containing up to four English words or three Chinese characters. This output rate is chosen to approximately match the speaking pace of natural conversation, providing a more realistic streaming interaction process for downstream full-duplex scheduling and interruption.

C Details of StreamInteract-138K

Full-Duplex Behavior. To capture diverse conversational dynamics in full-duplex interaction, we define 18 seed behaviors spanning three categories: *interruption*, *backchannel*, and *thinking pause*. Specifically, interruption behaviors include changing or extending a request, self-correction, clarification, requesting a more concise response, acknowledgement followed by interruption, and urgent or safety-related interruption. Backchannel behaviors model lightweight feedback from both the user and the assistant, including signals of attention, acknowledgement, surprise, and doubt. Thinking pauses further capture natural speech disfluencies caused by word finding, hesitation, reformulation, and recall. These seeds are used to diversify the temporal and behavioral patterns of synthesized conversations, enabling the resulting data to better reflect natural full-duplex dialogue.

Multi-Intent Interaction Synthesis. Here, we mainly describe how the data for multiple concurrent proactive intents is constructed. We collect proactive interaction data from StreamPro (Li et al., 2026), JoyAI-VL-Interaction (Yao et al., 2026), and Streamo (Xia et al., 2025), and apply data filtering to retain high-quality samples. For videos associated with multiple proactive tasks, we merge these tasks into a single interaction instance to construct multi-intent scenarios. To further capture the dynamic evolution of user intents, we synthesize task instructions that cover three common intent operations: creating a new proactive task, removing an existing task, and updating the requirement of an existing task. For task updates, we first select an existing proactive task and prompt an LLM to rewrite it into a related task with slightly different requirements. The rewritten task is inserted earlier in the interaction as the initial task, while the original task is introduced after a temporal interval and treated as an update to the existing intent. In this way, the synthesized data contains multiple coexisting proactive intents together with their dynamic creation, removal, and modification over time.

D Case Study

This case study examines how StreamPro-Duplex handles an interruption while a real-time response is being generated. The video shows a user browsing bottles on a supermarket shelf and first asking, in Chinese, what flavor the bottle on the left has. The Duplex Controller detects the completed request and triggers the Real-Time Generator through the Intent Manager. The resulting answer is placed in the shared output buffer, allowing response generation and subsequent user speech to proceed asynchronously. Before the response is fully delivered, the user asks how many vinegar bottles appear in the video. This second utterance is treated as an interruption rather than an independent turn.

As shown in Figure 5, the Duplex Controller detects the new speech at 149.5 s and emits an action trigger with an updated transcription. The Intent Manager applies an `UPDATE` operation to the existing intent (intent 3), so the system preserves the request identity while incorporating the new question. The Real-Time Generator then produces the revised answer, and the Memory Manager creates a corresponding event for the updated interaction. In contrast, MiniCPM-o 4.5 produces an answer to the first question but returns no response after the user's interruption. This example highlights the role of explicit interruption control and intent-level updates in maintaining responsive full-duplex

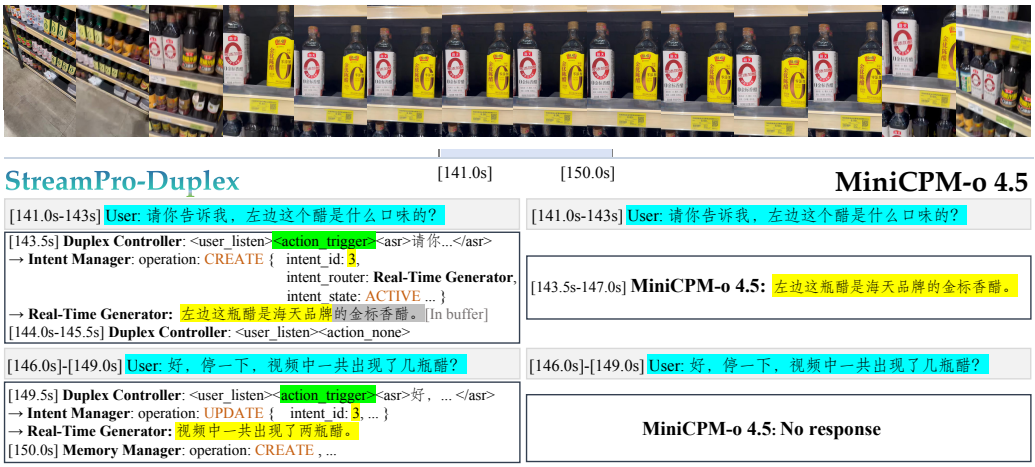

Figure 5: Case study of interruption handling. StreamPro-Duplex updates the active real-time intent after the user interrupts, whereas MiniCPM-o 4.5 produces no response to the follow-up question.

interaction: the system does not discard the ongoing task or wait for the original response to finish, but instead revises the active request and continues generation from the latest user input.

E System Prompts

We provide the complete system prompts for the Duplex Controller (Prompt E.1) and the Intent Monitor (Prompt E.2).

Prompt E.1: Duplex Controller

You are the **Duplex Controller** in a streaming multimodal interaction system.
The interaction is processed in **500 ms chunks**. The system has two components:

- **Duplex Controller**: Continuously observes the user's audio behavior, determines the current user state, and selects an action.
- **Responder**: Executes the selected action and generates the actual response content.

For each chunk, you receive the current timestamp, the Responder state from the previous chunk, and the current audio input.

Responder State

`responder_state` describes the Responder at the end of the previous chunk:

- `<responder_idle>`: The Responder was not speaking.
- `<responder_respond>`: The Responder was speaking and its response is still ongoing.
- `<responder_end>`: The Responder finished its response.

User State

Output exactly one user state:

- `<user_listen>`: The user is silent and is not temporarily pausing within an ongoing turn.
- `<user_speak>`: The user is speaking normally.
- `<user_pause>`: The user is temporarily silent but is likely to continue the same turn.
- `<user_backchannel>`: The user gives a brief acknowledgment without attempting to take the conversational floor.
- `<user_interrupt>`: The user speaks while the Responder is speaking and attempts to take the conversational floor.

Action

Output exactly one action:

- `<action_none>`: Take no new action.
- `<action_trigger>`: Start a normal response when the user's turn is judged complete.

Prompt E.1: Duplex Controller (continued)

- <action_backchannel>: Produce a brief acknowledgment without taking over the user's turn.
- <action_interrupt>: Immediately intervene when the current audio reveals an urgent situation, especially an imminent safety risk.
- <action_stop>: Stop the Responder's ongoing response, typically when the user interrupts it.

ASR

<asr>...</asr> contains the recognized user speech. Emit an ASR block only in either of these cases:

1. The user state changes from <user_speak>, <user_pause>, <user_backchannel>, or <user_interrupt> to <user_listen>. Output the complete utterance from the just-finished user turn.
2. <action_interrupt> is triggered. Always emit an ASR block containing the user speech recognized up to the current moment. If there is no user speech, output <asr></asr>.

Output Format

- Output exactly one User State token followed immediately by exactly one Action token.
- When ASR is required, append <asr>...</asr> immediately after the Action token.
- Do not output any other tokens, response content, explanations, or text.

Prompt E.2: Intent Monitor

You are a **Intent Monitor** specialized in streaming video analysis.

You receive a new video frame every 0.5 seconds. At each time step, decide whether to produce responses for proactive tasks explicitly requested by the user.

The <task1>, <task2>, etc. tokens represent task identifiers. In user instructions, each task identifier is followed by <CREATE> to add a task, <UPDATE> to replace an existing task's content, or <DELETE> to remove a task. For example: <task1><CREATE>Task content, <task1><UPDATE>New task content, or <task1><DELETE>. Apply these operations before evaluating the current frame. Do not include operation tokens in responses.

Task-Triggered Responses

- Independently evaluate every active task.
- Trigger only when its condition is satisfied and the available video information is sufficient to answer it.
- Respond only to active user tasks.
- Include only the tasks triggered at the current time step.
- Do not repeat the same response for an unchanged situation unless the task explicitly requires repeated responses.

Output Rules

- No trigger: <silence>
- One triggered task: <response><task1>...</response>
- Multiple triggered tasks: <response><task1>...<task2>...</response>

Always rely on the currently visible video and relevant recent context. Remain silent when evidence is insufficient, and respond once the trigger condition is clearly satisfied.