ObjectStream: Latent Objects as Memory Anchors for Streaming Video Understanding

Mingkang Dong[1]*, Muxin Pu[2]*, Jie Li[3], Bohan Guo[1], Songruo Chen[4], Bin Ren[5], Xu Zheng[6], Chen Zhao[7], Tianwen Qian[8], Mohamed Elhoseiny[7], Yuqian Fu[†7]

[1]Universiti Malaya [2]Monash University [3]SJTU [4]Fudan University
[5]MBZUAI [6]Great Bay University [7]KAUST [8]ECNU

Abstract

Streaming video understanding requires models to continuously retain useful visual evidence before future questions are known. Existing approaches primarily manage the growing visual context according to token importance, temporal redundancy, or segment-level relevance, but rarely organize evidence around objects that persist and evolve over time. Thus, in this paper, we introduce **ObjectStream**, a training-free framework that treats ***latent objects as memory anchors*** for streaming video understanding. ObjectStream induces spatially coherent latent objects directly from frozen Video-LLM representations, links them across frames into persistent anchors, and maintains their histories under a bounded memory budget, without requiring external object detectors or segmentation models. Built on these anchors, ObjectStream preserves three complementary forms of evidence: *persistent object histories*, *transient object changes*, and *recent visual context*. This design enables existing Video Large Language Models (Video-LLMs) to reason over object identities, interactions, and state changes while leaving the underlying model unchanged. Extensive experiments on online streaming and offline long-video benchmarks demonstrate both effectiveness and efficiency. In online streaming evaluation, ObjectStream improves Qwen2.5-VL-7B by 10.0 points on OVO-Bench Real-Time Visual Perception, while reducing peak GPU memory and TTFT by approximately 50%. On offline long-video benchmarks, it surpasses the full-token baseline while discarding 82.5% of visual tokens. These results highlight latent objects as a practical and effective organizing principle for compact streaming video memory.

Introduction

Streaming video understanding (Maaz et al. 2024; Zhang et al. 2025; Bai et al. 2025) requires models to process continuously arriving visual inputs and respond to questions that may be issued at any time. This capability is essential for applications such as wearable assistants (Plizzari et al. 2024), embodied agents (Wang et al. 2023), autonomous systems (Li, Zhang, and Chen 2026), and live monitoring (Dumitru and Spînu 2026). Unlike offline video understanding, where the complete video can be revisited after a question is given, streaming models must decide what information to retain before future queries are known, while operating under strict memory and latency constraints.

Recent studies (Wu et al. 2026; Liang et al. 2026; Zhang et al. 2026a; Guan et al. 2026) have extended Video Large Language Models (Video-LLMs) to streaming scenarios from various perspectives, including online adaptation (Shen et al. 2024; Lu et al. 2026), streaming-oriented inference (Zhang et al. 2024; Xu et al. 2026), and memory-based context management (Xie et al. 2026; Liang et al. 2026; Zhang et al. 2026a; Ge et al. 2026; Wu et al. 2026). Among them, training-free approaches (Xie et al. 2026; Chen et al. 2026; Liang et al. 2026) are particularly attractive because they can adapt existing offline Video-LLMs without additional training or modification to the underlying model. These methods typically control the growing visual context through frame selection (Shen et al. 2026), visual-token pruning (Xie et al. 2026), KV-cache compression (Di et al. 2025; Chen et al. 2025a), and temporal or segment-level memory organization (Liang et al. 2026; Zhang et al. 2026a). By reducing redundant historical information and prioritizing potentially relevant evidence, they improve memory efficiency and response latency, facilitating the transition of Video-

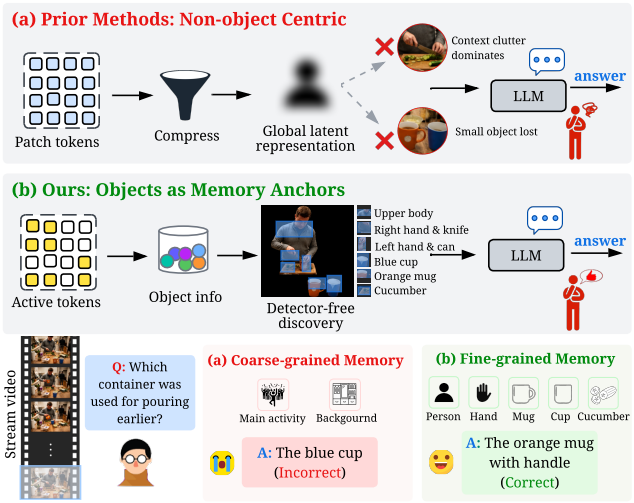

Figure 1: Comparison between existing streaming memory and ObjectStream. Existing methods compress streaming observations into coarse latent representations, whereas ObjectStream organizes memory around latent objects to preserve fine-grained object evidence for long-term reasoning.

*These authors contributed equally.
†Corresponding author.

LLMs from offline video understanding to online streaming.

However, deciding which visual units or temporal portions to retain addresses only part of the streaming memory problem; how the retained evidence is organized is equally important for subsequent reasoning. While existing designs are effective at preserving coarse-grained contextual and temporal semantics, finer-grained object-level information is often less explicitly organized across time. This limitation is particularly problematic for questions that require tracing an object's identity, interactions, or state changes. These observations motivate an object-centric organization of streaming visual memory.

Based on this insight, we introduce **ObjectStream**, a training-free and plug-and-play visual memory framework that treats ***latent objects as memory anchors*** for streaming Video-LLMs. Rather than managing historical visual tokens or KV states in isolation, ObjectStream organizes the incoming stream through a *Latent Object-Anchored Memory*. Specifically, it induces spatially coherent object candidates directly from frozen Video-LLM representations, associates them across frames, and maintains their evolving histories under a bounded memory budget. Since these object-like units emerge entirely from the model's latent feature space, without requiring external object detectors or segmentation models, we refer to them as *latent objects*. The resulting cross-frame object states provide a compact semantic index over the stream, enabling the model to preserve object-related evidence beyond locally salient visual content. To complement persistent object histories, ObjectStream introduces two additional components. *Object-Conditioned Temporal Residuals* preserve short-lived but important evidence around abrupt object-level changes, while the *Recent Visual Grounding Window* retains the latest observations for timely visual grounding. Together, these components capture three complementary forms of memory: *persistent object histories*, *transient object changes*, and *recent visual context*, while leaving the underlying Video-LLM unchanged.

We validate ObjectStream on both online streaming and offline long-video understanding benchmarks. It improves Qwen2.5-VL-7B by 10.0, 4.9, and 2.9 points on OVO-Bench Real-Time Visual Perception, OVO-Bench Backward Tracing, and StreamingBench, respectively. It also surpasses the full-token baseline on offline long-video benchmarks while discarding 82.5% of visual tokens. These results well demonstrate the effectiveness and token efficiency of ObjectStream.

Our main contributions are summarized as follows:

- We propose ObjectStream, a training-free and plug-and-play visual memory framework for efficient streaming video understanding, without architectural modifications.
- We introduce *Latent Object-Anchored Memory*, using latent objects as persistent anchors to organize long-term visual evidence without external detectors or segmenters.
- We complement persistent object memory with *Object-Conditioned Temporal Residuals* and a *Recent Visual Grounding Window*, achieving strong performance and token efficiency across online streaming and offline video understanding benchmarks.

Related Work

Video Large Language Models. Many Video Large Language Models (Video-LLMs) have been proposed to address video-language alignment, temporal modeling, and long-context reasoning. Representative proprietary models include Gemini 1.5 Pro (Team and Vinyals 2024) and GPT-4o (OpenAI:, Dai, and Malkov 2024), while widely used open-source models include LLaVA-Video (Zhang et al. 2025), Qwen2-VL (Wang et al. 2024), InternVL2 (Chen et al. 2025b), and LongVU (Shen et al. 2024). Despite their strong performance, these models primarily follow an offline inference paradigm, assuming that the complete video is available before question answering. This assumption limits their direct applicability to the more realistic and challenging streaming setting, where visual inputs arrive continuously and future observations are unavailable.

Streaming Video-LLMs and Visual Memory. Existing streaming video understanding models explore either training-based adaptation or training-free memory management. Training-based methods include VideoLLM-Online (Chen et al. 2024), which introduces online video instruction tuning for continuous video perception and interaction; Dispider (Qian et al. 2025), which enables active real-time interaction by disentangling perception, decision, and reaction; and ThinkStream (Liu et al. 2026), which adopts a Watch–Think–Speak paradigm and learns compressed semantic memory for long-horizon streaming. ViSpeak (Fu et al. 2025b) and StreamForest (Zeng et al. 2025) further improve online video understanding through streaming-oriented datasets, training objectives, and model adaptations. These methods generally require additional optimization or model-specific modifications.

Training-free approaches instead preserve the underlying Video-LLM and manage historical visual information directly at inference time. StreamChat (Liu et al. 2025) maintains external visual memory to incorporate previous observations, while ReKV and StreamKV (Di et al. 2025; Chen et al. 2025a) reduce memory overhead through KV-cache retention and compression. More recent methods explore structured memory management: FluxMem (Xie et al. 2026) introduces hierarchical visual memory to balance long-term history and recent observations; QueryStream (Zhang et al. 2026b) performs query-aware token pruning to retain request-relevant evidence; and OASIS (Liang et al. 2026) organizes streaming history into event-level structures for on-demand retrieval. However, these methods do not explicitly maintain persistent latent object representations across frames, limiting their ability to preserve object identities and state evolution over long video streams.

Method

Overview. Our design follows three core objectives: maintaining persistent object histories, preserving transient object-level changes, and retaining recent visual context for real-time grounding. The overall framework of our proposed **ObjectStream** is illustrated in Fig. 2 and consists of three components. First, *Latent Object-Anchored Memory* discovers spatially coherent latent object candidates from frozen

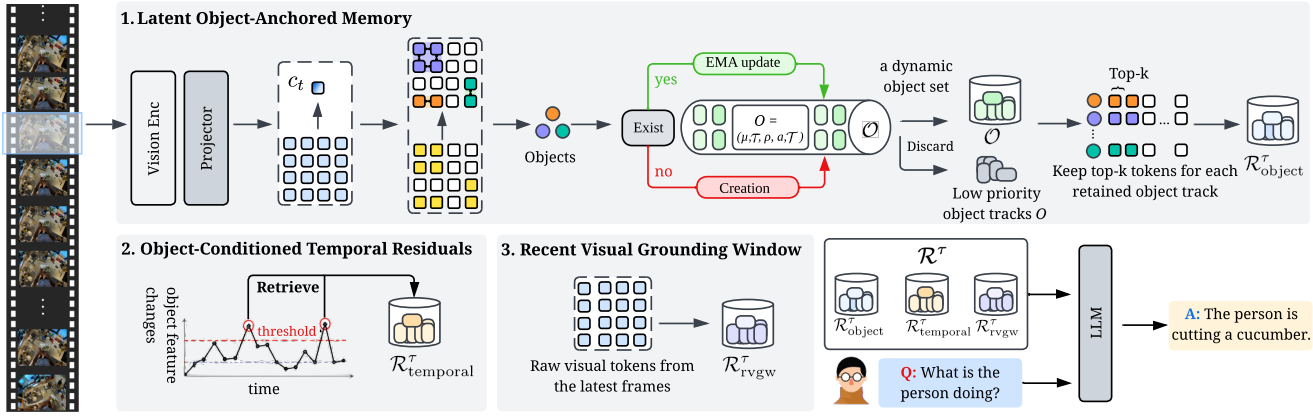

Figure 2: Overview of ObjectStream. It comprises three memory modules: *Latent Object-Anchored Memory* discovers and tracks latent object anchors and stores their persistent evidence; *Object-Conditioned Temporal Residuals* preserve transient changes; and the *Recent Visual Grounding Window* retains recent observations for immediate grounding.

Video-LLM representations, associates them across frames, and maintains their evolving histories under a bounded memory budget. Second, *Object-Conditioned Temporal Residuals* preserve short-lived evidence around abrupt object-level changes. Third, the *Recent Visual Grounding Window* retains the latest observations for timely question answering.

Formally, let $\mathcal{V}_\tau = \{I_t\}_{t=1}^{\tau}$ denote the video stream observed up to the current timestamp τ. ObjectStream processes each frame incrementally upon arrival. At timestamp $t \le \tau$, the newly observed frame I_t is encoded into a set of patch-level visual tokens: $\mathcal{X}_t = \{x_{t,j}\}_{j=1}^{N}$, where N is the number of visual tokens extracted from frame I_t, and each token $x_{t,j} \in \mathbb{R}^d$ is a d-dimensional visual representation. After processing all observations up to timestamp τ, ObjectStream maintains the retained visual memory as:

$$\mathcal{R}^{\tau} = \mathcal{R}^{\tau}_{\text{object}} \cup \mathcal{R}^{\tau}_{\text{temporal}} \cup \mathcal{R}^{\tau}_{\text{rvgw}}, \tag{1}$$

where $\mathcal{R}^{\tau}_{\text{object}}$, $\mathcal{R}^{\tau}_{\text{temporal}}$, and $\mathcal{R}^{\tau}_{\text{rvgw}}$ are produced by the aforementioned modules, retaining features associated with persistent object histories, transient object-level changes, and recent visual context, respectively.

Latent Object-Anchored Memory

To use latent objects as memory anchors for object-centric evidence retention, this module comprises four stages, denoted as *S1–S4*: *S1: Query-Agnostic Token Saliency Estimation*, *S2: Spatial Clustering for Latent Object Anchor Discovery*, *S3: Cross-Frame Association and Bounded Object-Set Maintenance* and *S4: Object-Centric Evidence Retention*. We describe each stage below.

S1: Query-Agnostic Token Saliency Estimation. For each frame I_t, we first summarize its overall visual content using a frame-level context vector c_t. Specifically, we ℓ_2-normalize each visual token and aggregate the normalized tokens into

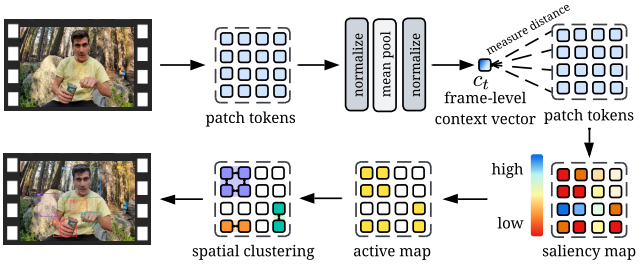

Figure 3: Illustration of latent object discovery. ObjectStream discovers latent object candidates directly from frozen visual tokens by estimating query-agnostic saliency and spatially clustering salient regions. The resulting latent objects provide the semantic object anchors for memory construction.

a global context representation :

$$\hat{x}_{t,j} = \frac{x_{t,j}}{\|x_{t,j}\|_2}, \qquad c_t = \frac{\sum_{j=1}^{N} \hat{x}_{t,j}}{\left\|\sum_{j=1}^{N} \hat{x}_{t,j}\right\|_2}. \tag{2}$$

The context vector c_t captures the dominant visual semantics of frame I_t. We then estimate the saliency of each token by its cosine dissimilarity from this frame-level context:

$$s_{t,j} = 1 - \cos(\hat{x}_{t,j}, c_t) = 1 - \hat{x}_{t,j}^{\top} c_t. \tag{3}$$

The score $s_{t,j}$ measures how visually distinctive token $x_{t,j}$ is within the current frame. Tokens that deviate more from the global frame representation are more likely to correspond to informative objects, persons, or regions. We then select active token candidates using a per-frame quantile threshold:

$$\mathcal{A}_t = \left\{ j \mid s_{t,j} \ge \text{Quantile}_q\left(\mathcal{S}_t\right) \right\}, \tag{4}$$

where $\mathcal{S}_t = \{s_{t,j}\}_{j=1}^{N}$ is the set of saliency scores for all visual tokens in frame t, and $\text{Quantile}_q(\mathcal{S}_t)$ denotes the q-th quantile of saliency scores within frame t; thus larger q selects fewer, more salient tokens as active object candidates.

S2: Spatial Clustering for Latent Object Anchor Discovery. As shown in Fig. 3, the active tokens indexed by $\mathcal{A}_t$ are still patch-level candidates and may be spatially fragmented. To form coherent object-level regions, ObjectStream exploits the 2D patch-grid structure of visual tokens. Let $p_{t,j}$ denote the spatial position of token $x_{t,j}$. We build an undirected graph $\mathcal{G}_t$ over active tokens, where edges connect spatially adjacent tokens:

$$\begin{aligned} \mathcal{G}_t &= (\mathcal{A}_t, \mathcal{E}_t), \\ (j,k) \in \mathcal{E}_t &\iff j,k \in \mathcal{A}_t \wedge p_{t,k} \in \mathcal{N}(p_{t,j}), \\ \mathcal{C}_t &= \mathrm{CC}(\mathcal{G}_t). \end{aligned} \tag{5}$$

Here, $\mathcal{E}_t$ is the edge set, $\mathcal{N}_4(p_{t,j})$ denotes the 4-connected spatial neighborhood of patch $p_{t,j}$ on the patch grid, including its upper, lower, left, and right neighbors. $\mathrm{CC}(\cdot)$ denotes connected components, and $\mathcal{C}_t = \{C_{t,m}\}_{m=1}^{M_t}$ is the set of frame-level object candidates.

$$\mu = f(C_{t,m}) = \frac{\sum_{j \in C_{t,m}} s_{t,j} x_{t,j}}{\left\| \sum_{j \in C_{t,m}} s_{t,j} x_{t,j} \right\|_2}. \tag{6}$$

This transforms scattered salient patch tokens into coarse object-level units without requiring external detectors or segmentation annotations.

S3: Cross-Frame Association and Bounded Object-Set Maintenance. After obtaining frame-level object candidates, ObjectStream associates them across frames to build persistent object tracks. We maintain a dynamic object set $\mathcal{O} = \{O_i\}$, where each object track O_i stores an object feature μ_i, the last observed timestamp τ_i, a temporal consistency score ρ_i, an object saliency score a_i, and a historical token set $\mathcal{T}_i$. The temporal consistency score ρ_i is computed as the cosine similarity between the track feature and the candidate feature at its most recent match, indicating the confidence of cross-frame association. Upon each successful match, the temporal consistency score ρ_i is updated as the cosine similarity. The object saliency score a_i is inherited from the token-level saliency introduced in S2 by running averaging the saliency values of the visual tokens belonging to the corresponding object candidate.

For each frame-level candidate $C_{t,m}$, ObjectStream first finds the most similar existing object track:

$$i^\star = \arg\max_i \cos\left(f(C_{t,m}), \mu_i\right). \tag{7}$$

If $\cos(f(C_{t,m}), \mu_{i^*}^{t-1}) \geq \theta$, the candidate is assigned to $O_{i^\star}$ and the object feature is updated by Exponential Moving Average (EMA):

$$\mu_{i^*}^t = \frac{\beta \mu_{i^*}^{t-1} + (1-\beta) f(C_{t,m})}{\left\| \beta \mu_{i^*}^{t-1} + (1-\beta) f(C_{t,m}) \right\|_2}. \tag{8}$$

If no existing object satisfies the matching threshold, ObjectStream initializes a new object track. For a matched object, we set τ_{i^*} to the current timestamp, update its temporal consistency ρ_{i^*} and average saliency a_{i^*}, and insert all tokens in the matched component into $\mathcal{T}_{i^*}$.

Since streaming videos continuously introduce new objects, the object set must remain bounded. When the number of maintained tracks exceeds the budget M, ObjectStream evicts low-priority objects according to recency, saliency, and temporal consistency.

$$\begin{aligned} \mathrm{rank}(O_i) &= (\tau_i,\ a_i, \rho_i), \\ \mathcal{O} &= \text{Top-M}_{O_i \in \mathcal{O}} \left(\mathrm{rank}(O_i)\right). \end{aligned} \tag{9}$$

The ranking is applied in lexicographic order, prioritizing object anchors that are recently observed, visually salient, and temporally consistent.

S4: Object-Centric Evidence Retention. Once persistent object tracks in the set $\mathcal{O}$ are maintained under a fixed budget, ObjectStream constructs the long-term object-centric evidence $\mathcal{R}^\tau_{\text{object}}$ by selecting representative tokens from each retained object anchor.

Object eviction and token retention operate at different granularities: the bounded object set $\mathcal{O}$ determines which object tracks are maintained up to timestamp τ, while token retention is performed over the historical token pool $\mathcal{T}_i^\tau$ of each retained track $O_i \in \mathcal{O}$. For each retained object O_i, its historical token pool contains token indices (t, j) assigned to this object up to timestamp τ. Instead of globally selecting top-scoring tokens from the whole video, which may be dominated by a few salient regions or frames, ObjectStream allocates token retention capacity at the object level:

$$\begin{aligned} \mathcal{I}_i^\tau &= \text{Top-K}_{(t,j) \in \mathcal{T}_i^\tau,\ t \leq \tau} \left(s_{t,j}, k_i\right), \\ \mathcal{R}^\tau_{\text{object}} &= \bigcup_{O_i \in \mathcal{O}} \left\{ x_{t,j} \mid (t,j) \in \mathcal{I}_i^\tau \right\}. \end{aligned} \tag{10}$$

The operator Top-K ranks tokens within the object-specific pool $\mathcal{T}_i^\tau$ by their saliency scores $s_{t,j}$ and returns the top-k_i token indices. The retained object memory $\mathcal{R}^\tau_{\text{object}}$ then collects the corresponding visual tokens from all maintained object tracks, forming a compact yet comprehensive representation of persistent visual histories.

Object-Conditioned Temporal Residuals

Latent Object-Anchored memory preserves persistent visual semantics by smoothing object representations over time. However, this smoothing may suppress short-lived but important changes, such as motion, action transitions, object appearance or disappearance, state changes, and interaction changes. Thus, this module is further proposed to detect abrupt object-level feature changes and preserve additional raw tokens around the corresponding timestamps.

For each object track O_i, we denote its matched sequence of historical observations as $\{z_{i,l}\}_{l=1}^{L_i}$. Specifically, if the frame-level object candidate $f(C_{t,m})$ is successfully assigned to track O_i at a timestamp t, this observation is mapped as $z_{i,l} = f(C_{t,m})$, where l indexes the chronologically ordered observation sequence up to length L_i, where $2 \leq l \leq L_i$, and $t_{i,l} = t$ preserves its corresponding timestamp. We then compute the feature change between neighboring observations and use an object-specific adaptive threshold to detect temporal events:

$$\begin{aligned} \Delta_{i,l} &= 1 - \cos(z_{i,l}, z_{i,l-1}), \\ \gamma_{i,l} &= \mathrm{mean}\left(\{\Delta_{i,k}\}_{k=2}^{l}\right) + \mathrm{std}\left(\{\Delta_{i,k}\}_{k=2}^{l}\right), \\ \mathcal{B}^\tau &= \{(i, t_{i,l}) \mid \Delta_{i,l} > \gamma_i,\ t_{i,l} \leq \tau\}. \end{aligned} \tag{11}$$

Here, $\Delta_{i,l}$ measures the temporal change of object O_i between two consecutive matched observations, $\gamma_{i,l}$ is its adaptive temporal threshold, and $\mathcal{B}^\tau$ denotes the temporal events detected up to timestamp τ.

For each temporal event $(i,t) \in \mathcal{B}^\tau$, ObjectStream retrieves the original visual tokens assigned to the same object within a local temporal window $\mathcal{W}(t,r) = \{u \mid t-r \le u \le t+r,\ u \le \tau\}$, where r denotes the temporal window frame offset that defines the local neighborhood around event timestamp t. Within this object-specific temporal neighborhood, we select token indices according to their saliency scores:

$$\begin{aligned} \mathcal{I}^\tau_{i,t} &= \mathrm{TopK}_{(u,j)\in\mathcal{T}^\tau_i,\ u\in\mathcal{W}(t,r)}\left(s_{u,j}, k_b\right), \\ \mathcal{R}^\tau_{\text{temporal}} &= \bigcup_{(i,t)\in\mathcal{B}^\tau} \left\{x_{u,j} \mid (u,j) \in \mathcal{I}^\tau_{i,t}\right\}. \end{aligned} \tag{12}$$

In Eq.12 Top-K returns the indices of the top-k_b tokens ranked by saliency within the local temporal neighborhood, and $\mathcal{R}^\tau_{\text{temporal}}$ stores the corresponding raw visual tokens.

Unlike object-level representative tokens, temporal residual tokens preserve raw visual evidence around rapid object-level changes. Therefore, they complement the smooth long-term object memory and improve the ability to reason about short-term temporal events.

Recent Visual Grounding Window

Although object memory and temporal residuals preserve long-term semantics and short-term dynamics, streaming questions may also require the latest visual state. For example, questions about what is currently happening or which object is now visible depend strongly on the most recent frames. To support such real-time questions, ObjectStream maintains a lightweight *Recent Visual Grounding Window* (rvgw), which directly preserves raw visual tokens from the latest L frames:

$$\mathcal{R}^\tau_{\text{rvgw}} = \bigcup_{t=T-L+1}^{T} \mathcal{X}_t. \tag{13}$$

The rvgw complements the compressed object memory by providing high-fidelity local evidence for the current visual state, while the object and temporal residual memories preserve compact historical context.

Inference. Given a textual question Q, the language model generates the answer by conditioning on the retained visual memory:

$$\hat{Y} = \mathrm{LLM}(Q, \mathcal{R}^\tau). \tag{14}$$

By integrating persistent object histories, transient object-level changes, and recent visual context, ObjectStream provides the underlying Video-LLM with a compact yet expressive representation of the observed stream. This unified memory supports both historical reasoning and grounding in the current visual state under a bounded visual-token budget.

Experiments

Implementation Details. For all experiments, we build our method on Qwen2.5-VL-3B and 7B (Bai et al. 2025). For online benchmarks, we sample videos at 1 fps and set the maximum video length to 256 frames. We use 8 frames as *the Recent Visual Grounding Window* (rvgw) for real-time question answering and compress older frames into object-centric memory. We keep up to 64 objects with 8 tokens per object, and further enable temporal residuals with at most 32 events, radius 1, and extra 2 tokens per selected frame. For offline benchmarks, we limit the maximum sequence length to 1024 frames. We use a larger object memory budget to preserve longer-range temporal context, keeping up to 512 objects with 80 tokens per object. We preserve the most recent 80 frames as the rvgw, while compressing older visual content into object-centric long-term memory.

Benchmarks. We evaluate our method on both online and offline benchmarks. For online evaluation, we use StreamingBench (Lin et al. 2024) and OVO-Bench (Li et al. 2025). StreamingBench evaluates streaming video question answering under causal access, requiring the model to answer questions incrementally as video frames arrive over time. OVO-Bench focuses on online video understanding with an emphasis on real-time reasoning, covering abilities such as object perception, temporal reasoning, causal understanding, and event-based decision making under constrained memory. For offline evaluation, we use EgoSchema (Mangalam, Akshulakov, and Malik 2023) and VideoMME-Long (Fu et al. 2025a). These benchmarks evaluate holistic video understanding when the entire video is available, with a particular focus on long-range temporal reasoning, egocentric understanding, and comprehensive multi-step video QA.

Results on Online Benchmark. We compare ObjectStream with strong proprietary models, open-source offline Video-LLMs, training-based online Video-LLMs, and training-free online methods, including Gemini 1.5 Pro (Team and Vinyals 2024), GPT-4o (OpenAI:, Dai, and Malkov 2024), LLaVA-Video (Zhang et al. 2025), Qwen2-VL (Wang et al. 2024), InternVL2 (Chen et al. 2025b), LongVU (Shen et al. 2024), VideoLLM-Online (Chen et al. 2024), Dispider (Qian et al. 2025), Flash-VStream (Zhang et al. 2024), ViSpeak (Fu et al. 2025b), ThinkStream (Liu et al. 2026), TimeChat-Online (Yao et al. 2025), StreamForest (Zeng et al. 2025), QueryStream (Zhang et al. 2026b), OASIS (Liang et al. 2026) and FluxMem (Xie et al. 2026).

Main Results on Streaming Benchmarks. As shown in Tab. 1, ObjectStream consistently improves streaming video understanding across online benchmarks without fine-tuning or architectural modification, demonstrating its effectiveness as a lightweight and plug-and-play visual memory mechanism for Video-LLMs.

On OVO-Bench, ObjectStream achieves substantial improvements over Qwen2.5-VL baselines. With the 7B backbone, it improves the real-time visual perception score from 63.3 to 73.3 (+10.0) and the Backward Tracing score from 44.6 to 49.5, indicating better preservation of historical visual cues. Compared with FluxMem, ObjectStream further improves the overall score from 60.4 to 65.4 on the 7B backbone and from 54.0 to 60.2 on the 3B backbone; On StreamingBench, ObjectStream improves the average score from 73.9 to 76.8 with the 7B backbone and from 68.0 to 72.4 with the 3B backbone. These results demonstrate that latent object memory construction provides a general and effective

Method	Frames	OVO-Bench Real-Time							BT	StreamingBench										
		OCR	ACR	ATR	STU	FPD	OJR	Avg.	Avg.	OP	CR	CS	ATP	EU	TR	PR	SU	ACP	CT	Avg.
Proprietary Models																				
Gemini 1.5 Pro	1 fps	85.9	67.0	79.3	58.4	63.4	62.0	69.3	62.5	79.0	80.5	83.5	79.7	80.0	84.7	77.8	64.2	72.0	48.7	75.7
GPT-4o	64	69.8	64.2	71.6	51.1	70.3	59.8	64.5	60.8	77.1	80.5	83.9	76.5	70.2	83.8	66.7	62.2	69.1	49.2	73.3
Open-source Offline MLLMs																				
LLaVA-Video	64	69.8	59.6	66.4	50.6	72.3	61.4	63.3	41.7	–	–	–	–	–	–	–	–	–	–	–
Qwen2-VL	64	69.1	53.2	63.8	50.6	66.3	60.9	60.7	48.6	–	–	–	–	–	–	–	–	–	–	–
InternVL2	64	68.5	58.7	69.0	44.9	67.3	56.0	60.7	44.0	68.1	60.9	69.4	77.1	67.7	62.9	59.3	53.3	55.0	56.5	63.7
LongVU	1 fps	55.7	49.5	59.5	48.3	68.3	63.0	57.4	39.5	–	–	–	–	–	–	–	–	–	–	–
Open-source Online MLLMs (Training-Based)																				
VideoLLM-Online-8B	2 fps	8.1	23.9	12.1	14.0	45.5	21.2	20.8	17.7	39.1	40.1	34.5	31.1	46.0	32.4	31.5	34.2	42.5	27.9	36.0
Dispider-7B	1 fps	57.7	49.5	62.1	44.9	61.4	51.6	54.6	36.1	74.9	75.5	74.1	73.1	74.4	59.9	76.1	62.9	62.2	45.8	67.6
Flash-VStream-7B	1 fps	25.5	32.1	29.3	33.7	29.7	28.8	29.9	25.4	25.9	43.6	24.9	23.9	27.3	13.1	18.5	25.2	23.9	48.7	23.2
ViSpeak	1 fps	75.2	58.7	71.6	51.1	74.3	66.9	66.3	**57.5**	79.8	**88.3**	83.3	81.1	76.4	75.1	70.4	65.9	**77.3**	34.2	74.4
ThinkStream	1 fps	85.2	64.2	69.8	49.4	69.3	64.1	67.0	52.3	–	–	–	–	–	–	–	–	–	–	–
TimeChat-Online-7B	1 fps	75.2	46.8	70.7	47.8	69.3	61.4	61.9	41.7	80.8	79.7	80.8	83.3	74.8	78.8	78.7	64.2	68.8	**58.0**	75.3
StreamForest-7B	1 fps	68.5	53.2	71.6	47.8	65.4	60.9	61.2	52.0	**83.1**	82.8	82.7	84.3	77.5	78.2	76.9	69.1	75.6	54.4	**77.3**
Open-source Online MLLMs (Training-Free)																				
Qwen2.5-VL-3B†	1 fps	76.5	44.0	67.2	42.1	68.3	62.0	60.0	42.0	76.2	68.8	75.4	79.2	73.0	72.3	71.3	61.4	71.6	26.1	68.0
+ FluxMem	1 fps	83.2	56.9	67.2	47.8	68.3	63.6	64.5	42.1	72.5	73.8	73.8	79.4	72.6	79.7	76.9	64.2	65.1	42.6	70.9
+ **ObjectStream (Ours)**	1 fps	87.3	68.8	73.3	54.5	66.3	65.2	68.6	42.5	72.7	69.8	79.8	80.4	73.9	82.8	65.7	71.1	72.0	34.6	72.4
Qwen2.5-VL-7B†	1 fps	79.2	53.2	67.2	51.7	71.3	57.1	63.3	44.6	78.3	80.5	79.8	82.4	75.5	80.4	74.1	62.6	67.6	51.1	73.9
+ FluxMem	1 fps	81.2	59.6	70.7	53.4	75.2	63.0	67.2	46.8	80.2	81.1	81.4	**85.3**	**78.0**	83.8	80.6	65.9	69.6	52.1	76.4
+ QueryStream	1 fps	75.2	49.5	69.8	50.0	71.3	62.5	63.1	44.9	82.4	84.4	79.2	82.4	**78.0**	81.3	78.7	65.0	69.3	47.3	75.3
+ OASIS	–	85.2	**72.5**	66.4	52.3	67.3	64.7	64.7	52.6	–	–	–	–	–	–	–	–	–	–	70.6
+ **ObjectStream (Ours)**	1 fps	**89.9**	70.6	**74.1**	**59.0**	**75.3**	**70.7**	**73.3**	49.5	74.1	80.2	**88.6**	82.6	77.1	**85.0**	**82.4**	**73.6**	73.2	44.2	76.8

Table 1: Performance comparison on OVO-Bench and StreamingBench. For OVO-Bench, we report real-time visual perception scores and the average backward tracing score. For StreamingBench, we report real-time category scores and the average score. **Bold** indicates the best result, and underlining indicates the second-best result. † indicates the backbone model.

solution for online perception and temporal reasoning.

Effect of Memory Components. To evaluate the contributions of Recent Visual Grounding Window (RVGW), Latent Object-Anchored Memory (LOAM), and Object-Conditioned Temporal Residuals (OCTR), we progressively enable each module. As shown in Tab. 2, RVGW mainly improves StreamingBench by preserving recent visual observations, while LOAM enhances OVO-Bench through structured object memory. Combining the two consistently improves performance across both streaming and offline settings. Adding OCTR further preserves abrupt object changes, resulting in the best overall performance.

Table 2: Component ablation of ObjectStream.

Component			VME	Ego	OVO	Streaming
RVGW	LOAM	OCTR	Long	Schema	Bench	Bench
✓			52.3	58.6	62.3	73.9
	✓		51.1	58.3	62.0	74.8
✓	✓		53.4	59.4	60.6	76.5
	✓	✓	53.8	60.4	62.1	74.9
✓	✓	✓	**54.0**	**60.8**	**65.4**	**76.8**

Extension to Offline Long-video Benchmarks. We also evaluate ObjectStream on offline long-video benchmarks to verify whether the proposed memory design remains effective in the offline scenario. As shown in Tab. 3, ObjectStream consistently improves the original Qwen2.5-VL baselines on both VideoMME-Long and EgoSchema. With Qwen2.5-VL-7B, our method improves VideoMME-Long from 51.3 to 54.0 and EgoSchema from 58.5 to 60.8, achieving gains of 2.7 and 2.3 points, respectively. With Qwen2.5-VL-3B, ObjectStream also improves VideoMME-Long from 49.3 to 52.9 and EgoSchema from 54.9 to 57.7, showing that the benefit is consistent across model scales. These results demonstrate that ObjectStream is not limited to online streaming inference.

Object Memory Construction and Efficiency Analysis. We further analyze different object memory construction strategies under the same Qwen2.5-VL-7B backbone on OVO-Bench. As shown in Tab. 4, Patch Top-K directly selects salient patch tokens without explicit object modeling. Although it keeps memory usage low, it yields lower performance and much higher TTFT than our method, showing that unstructured token selection is insufficient for streaming visual memory. DINOv2 Clustering introduces external patch features for object memory construction but brings

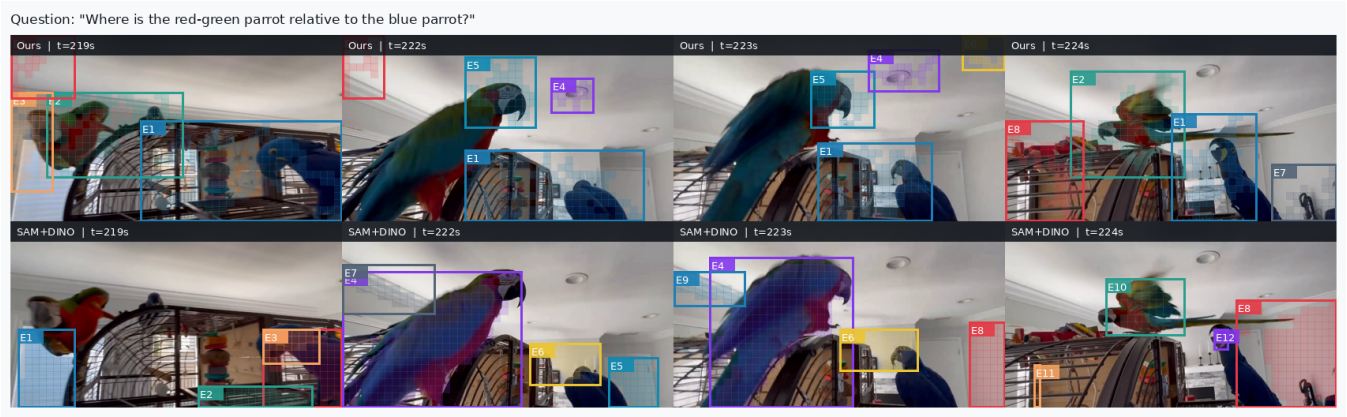

Figure 4: Qualitative analysis of latent object discovery. ObjectStream maintains task-relevant object states across consecutive observations, while SAM+DINOv2 produces finer masks but less consistent temporal associations.

Model	#Frames	VideoMME-L	EgoSchema
Proprietary MLLMs			
Gemini 1.5 Pro	1 fps	62.5	69.3
GPT-4o	64	60.7	64.5
Open-source Offline MLLMs			
LLaVA-Video-7B	32	–	57.3
Qwen2-VL-7B	64	–	**66.7**
LongVU-7B	1 fps	**60.6**	58.2
Open-source Online MLLMs			
TimeChat-Online-7B	1 fps	48.4	61.9
Dispider-7B	1 fps	49.7	55.6
Vista	1 fps	–	58.7
Training-free Offline-to-Online Methods			
Qwen2.5-VL-3B†	1 fps	49.3	54.9
+ FluxMem	1 fps	51.2	57.8
+ **ObjectStream (Ours)**	1 fps	52.9	57.7
Qwen2.5-VL-7B†	1 fps	51.3	58.5
+ FluxMem	1 fps	53.9	60.1
+ QueryStream	1 fps	52.9	–
+ **ObjectStream (Ours)**	1 fps	54.0	60.8

Table 3: Performance comparison (%) on offline benchmarks.

additional memory and latency overhead while still underperforming our method. SAM+DINOv2 achieves the highest accuracy by using segmentation-based regions, yet it requires expensive segmentation and feature extraction, resulting in substantially higher latency and memory cost.

In contrast, ObjectStream discovers objects directly in the backbone visual token space and combines them with temporal retention. It achieves 73.3 performance with only 19.89 GB peak memory and 6.697s TTFT, offering a better balance between accuracy and efficiency. Fig. 4 also illustrates this difference qualitatively. Although SAM+DINOv2 produces finer object masks, its outputs are optimized for pixel-level segmentation and do not directly provide a temporally reusable memory state for VLM reasoning. ObjectStream produces coarser but sufficiently localized object regions and maintains task-relevant objects across consecutive observations despite changes in scale, pose, and camera viewpoint. These results suggest that ObjectStream does not aim to replace segmentation models; instead, it provides a lightweight and temporally grounded object construction mechanism that better satisfies the low-latency and memory-bounded requirements of streaming video understanding.

Table 4: Analysis on different object memory construction strategies with Qwen2.5-VL-7B as backbone.

Variant	Peak Mem↓	TTFT↓	TPOT↓	Perf↑
Reference Baselines				
Qwen2.5-VL-7B	40.50GB	13.431s	0.336s	64.2
+ FluxMem	25.78GB	**6.641s**	0.252s	67.7
object Construction Variants				
Patch Top-K	**19.89GB**	16.223s	0.320s	72.8
DINOv2 Cluster	23.09GB	18.120s	1.230s	72.3
SAM+DINOv2	26.78GB	19.060s	0.785s	**73.8**
Latent Objects (Ours)	**19.89GB**	6.697s	**0.240s**	73.3

Conclusion

To conclude, in this paper, we propose ObjectStream, a training-free object-centric memory mechanism for efficient streaming video understanding. By discovering object-latent anchors and linking them into persistent tracks, ObjectStream builds a compact semantic object related memory over incoming video streams. This design preserves persistent visual evidence while remaining sensitive to dynamic changes and the latest visual state, enabling effective reasoning under memory and latency constraints. Experiments on both streaming and offline long-video benchmarks show that ObjectStream substantially improves online and offline video understanding while maintaining strong robustness under limited token budgets.

References

Bai, S.; Chen, K.; Liu, X.; Wang, J.; Ge, W.; Song, S.; Dang, K.; Wang, P.; Wang, S.; Tang, J.; Zhong, H.; Zhu, Y.; Yang, M.; Li, Z.; Wan, J.; Wang, P.; Ding, W.; Fu, Z.; Xu, Y.; Ye, J.; Zhang, X.; Xie, T.; Cheng, Z.; Zhang, H.; Yang, Z.; Xu, H.; and Lin, J. 2025. Qwen2.5-VL Technical Report. arXiv:2502.13923.

Chen, J.; Lv, Z.; Wu, S.; Lin, K. Q.; Song, C.; Gao, D.; Liu, J.-W.; Gao, Z.; Mao, D.; and Shou, M. Z. 2024. VideoLLM-online: Online Video Large Language Model for Streaming Video. arXiv:2406.11816.

Chen, X.; Tao, K.; Shao, K.; and Wang, H. 2026. StreamingTOM: Streaming Token Compression for Efficient Video Understanding. arXiv:2510.18269.

Chen, Y.; Bai, X.; Wang, Z.; Bai, C.; Dai, Y.; Lu, M.; and Zhang, S. 2025a. StreamKV: Streaming Video Question-Answering with Segment-based KV Cache Retrieval and Compression. arXiv:2511.07278.

Chen, Z.; Wang, W.; Cao, Y.; Liu, Y.; Gao, Z.; Cui, E.; Zhu, J.; Ye, S.; Tian, H.; Liu, Z.; Gu, L.; Wang, X.; Li, Q.; Ren, Y.; Chen, Z.; Luo, J.; Wang, J.; Jiang, T.; Wang, B.; He, C.; Shi, B.; Zhang, X.; Lv, H.; Wang, Y.; Shao, W.; Chu, P.; Tu, Z.; He, T.; Wu, Z.; Deng, H.; Ge, J.; Chen, K.; Zhang, K.; Wang, L.; Dou, M.; Lu, L.; Zhu, X.; Lu, T.; Lin, D.; Qiao, Y.; Dai, J.; and Wang, W. 2025b. Expanding Performance Boundaries of Open-Source Multimodal Models with Model, Data, and Test-Time Scaling. arXiv:2412.05271.

Di, S.; Yu, Z.; Zhang, G.; Li, H.; Zhong, T.; Cheng, H.; Li, B.; He, W.; Shu, F.; and Jiang, H. 2025. Streaming Video Question-Answering with In-context Video KV-Cache Retrieval. arXiv:2503.00540.

Dumitru, E.; and Spînu, S. 2026. A Multi-Task Deep Learning Framework for Real-Time Intelligent Video Surveillance with Temporal Event Validation. arXiv:2607.03131.

Fu, C.; Dai, Y.; Luo, Y.; Li, L.; Ren, S.; Zhang, R.; Wang, Z.; Zhou, C.; Shen, Y.; Zhang, M.; Chen, P.; Li, Y.; Lin, S.; Zhao, S.; Li, K.; Xu, T.; Zheng, X.; Chen, E.; Shan, C.; He, R.; and Sun, X. 2025a. Video-MME: The First-Ever Comprehensive Evaluation Benchmark of Multi-modal LLMs in Video Analysis. arXiv:2405.21075.

Fu, S.; Yang, Q.; Li, Y.-M.; Peng, Y.-X.; Lin, K.-Y.; Wei, X.; Hu, J.-F.; Xie, X.; and Zheng, W.-S. 2025b. ViSpeak: Visual Instruction Feedback in Streaming Videos. arXiv:2503.12769.

Ge, H.; Wang, Y.; Wu, H.; and Cai, Y. 2026. What Should a Streaming Video Model Remember? arXiv:2606.16353.

Guan, Y.; Yin, L.; Liang, D.; Ju, J.; Luo, Z.; Luan, J.; Liu, Y.; and Bai, X. 2026. Video Streaming Thinking: VideoLLMs Can Watch and Think Simultaneously. arXiv:2603.12262.

Li, F.; Zhang, C.; and Chen, G. 2026. Sparse-Aware Vector Quantization for Bandwidth-Efficient Collaborative 3D Semantic Occupancy Prediction. arXiv:2607.01928.

Li, Y.; Niu, J.; Miao, Z.; Ge, C.; Zhou, Y.; He, Q.; Dong, X.; Duan, H.; Ding, S.; Qian, R.; Zhang, P.; Zang, Y.; Cao, Y.; He, C.; and Wang, J. 2025. OVO-Bench: How Far is Your Video-LLMs from Real-World Online Video Understanding? arXiv:2501.05510.

Liang, Z.; Li, J.; Chen, W.; Zhang, Y.; Lu, H.; and Li, G. 2026. OASIS: On-Demand Hierarchical Event Memory for Streaming Video Reasoning. arXiv:2604.17052.

Lin, J.; Fang, Z.; Chen, C.; Wan, Z.; Luo, F.; Li, P.; Liu, Y.; and Sun, M. 2024. StreamingBench: Assessing the Gap for MLLMs to Achieve Streaming Video Understanding. arXiv:2411.03628.

Liu, J.; Yu, Z.; Lan, S.; Wang, S.; Fang, R.; Kautz, J.; Li, H.; and Alvare, J. M. 2025. StreamChat: Chatting with Streaming Video. arXiv:2412.08646.

Liu, Z.; Guo, L.; Li, H.; Zhen, R.; He, X.; Ji, R.; Ren, X.; Zhang, Y.; Lu, H.; and Liu, J. 2026. Thinking in Streaming Video. arXiv:2603.12938.

Lu, X.; Bo, Y.; Chen, J.; Li, S.; Guo, X.; Guan, H.; Liu, F.; Xu, D.; Sun, P.; Sun, H.; Liu, R.; and Li, H. 2026. AURA: Always-On Understanding and Real-Time Assistance via Video Streams. arXiv:2604.04184.

Maaz, M.; Rasheed, H.; Khan, S.; and Khan, F. S. 2024. Video-ChatGPT: Towards Detailed Video Understanding via Large Vision and Language Models. arXiv:2306.05424.

Mangalam, K.; Akshulakov, R.; and Malik, J. 2023. EgoSchema: A Diagnostic Benchmark for Very Long-form Video Language Understanding. arXiv:2308.09126.

OpenAI:; Dai, Y.; and Malkov, Y. 2024. GPT-4o System Card. arXiv:2410.21276.

Plizzari, C.; Goletto, G.; Furnari, A.; Bansal, S.; Ragusa, F.; Farinella, G. M.; Damen, D.; and Tommasi, T. 2024. An Outlook into the Future of Egocentric Vision. arXiv:2308.07123.

Qian, R.; Ding, S.; Dong, X.; Zhang, P.; Zang, Y.; Cao, Y.; Lin, D.; and Wang, J. 2025. Dispider: Enabling Video LLMs with Active Real-Time Interaction via Disentangled Perception, Decision, and Reaction. arXiv:2501.03218.

Shen, X.; Xiong, Y.; Zhao, C.; Wu, L.; Chen, J.; Zhu, C.; Liu, Z.; Xiao, F.; Varadarajan, B.; Bordes, F.; Liu, Z.; Xu, H.; Kim, H. J.; Soran, B.; Krishnamoorthi, R.; Elhoseiny, M.; and Chandra, V. 2024. LongVU: Spatiotemporal Adaptive Compression for Long Video-Language Understanding. arXiv:2410.17434.

Shen, Y.; Tian, S.; Yang, J.; and Liu, Z. 2026. A Simple Baseline for Streaming Video Understanding. arXiv:2604.02317.

Team, G.; and Vinyals, P. G. O. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. arXiv:2403.05530.

Wang, P.; Bai, S.; Tan, S.; Wang, S.; Fan, Z.; Bai, J.; Chen, K.; Liu, X.; Wang, J.; Ge, W.; Fan, Y.; Dang, K.; Du, M.; Ren, X.; Men, R.; Liu, D.; Zhou, C.; Zhou, J.; and Lin, J. 2024. Qwen2-VL: Enhancing Vision-Language Model's Perception of the World at Any Resolution. arXiv:2409.12191.

Wang, X.; Kwon, T.; Rad, M.; Pan, B.; Chakraborty, I.; Andrist, S.; Bohus, D.; Feniello, A.; Tekin, B.; Frujeri, F. V.; Joshi, N.; and Pollefeys, M. 2023. HoloAssist: an Egocentric Human Interaction Dataset for Interactive AI Assistants in the Real World. arXiv:2309.17024.

Wu, H.; Mathews, S. M.; Cai, Y.; Yang, M.-H.; and Wang, Y. 2026. Semantic-Aware Adaptive Visual Memory for Streaming Video Understanding. arXiv:2605.07897.

Xie, Y.; He, B.; Wang, J.; Zheng, X.; Ye, Z.; and Wu, Z. 2026. FluxMem: Adaptive Hierarchical Memory for Streaming Video Understanding. arXiv:2603.02096.

Xu, R.; Xiao, G.; Chen, Y.; He, L.; Lu, Y.; and Han, S. 2026. StreamingVLM: Real-Time Understanding for Infinite Video Streams. arXiv:2510.09608.

Yao, L.; Li, Y.; Wei, Y.; Li, L.; Ren, S.; Liu, Y.; Ouyang, K.; Wang, L.; Li, S.; Li, S.; Kong, L.; Liu, Q.; Zhang, Y.; and Sun, X. 2025. TimeChat-Online: 80 arXiv:2504.17343.

Zeng, X.; Qiu, K.; Zhang, Q.; Li, X.; Wang, J.; Li, J.; Yan, Z.; Tian, K.; Tian, M.; Zhao, X.; Wang, Y.; and Wang, L. 2025. StreamForest: Efficient Online Video Understanding with Persistent Event Memory. arXiv:2509.24871.

Zhang, H.; Wang, Y.; Tang, Y.; Liu, Y.; Feng, J.; Dai, J.; and Jin, X. 2024. Flash-VStream: Memory-Based Real-Time Understanding for Long Video Streams. arXiv:2406.08085.

Zhang, H.; Yang, S.; Fu, J.; Ng, S.-K.; and Qiu, X. 2026a. HERMES: KV Cache as Hierarchical Memory for Efficient Streaming Video Understanding. arXiv:2601.14724.

Zhang, K.; Yang, Z.; Wang, B.; Qian, S.; and Xu, C. 2026b. QueryStream: Advancing Streaming Video Understanding with Query-Aware Pruning and Proactive Response. In *The Fourteenth International Conference on Learning Representations*.

Zhang, Y.; Wu, J.; Li, W.; Li, B.; Ma, Z.; Liu, Z.; and Li, C. 2025. LLaVA-Video: Video Instruction Tuning With Synthetic Data. arXiv:2410.02713.