# LifeStream: Token Life-Cycle Modeling for Training-Free Online Video Understanding

**Yifan Ge, Zhihang Liu, Jiannan Ge†, Chenhui Jin, Hongtao Xie**
University of Science and Technology of China
geyifan@mail.ustc.edu.cn, gejn@mail.ustc.edu.cn

## Abstract

With the rapid advancement of Video Large Language Models (VLLMs), online video understanding has emerged as a practical paradigm for real-world streaming applications. A central challenge is to build a compact, query-agnostic visual memory from continuously expanding video streams while respecting the LLM's visual-token budget. Existing token reduction approaches generally rely on static, coarse-grained importance estimates, implicitly assuming that retained tokens remain equally useful over time. However, a token's relevance may emerge, persist, or decay as new observations arrive. Ignoring these temporal dynamics leads to suboptimal token prioritization and inefficient use of long-term memory. To address this limitation, we introduce LifeStream, a training-free framework for lifecycle-aware visual token selection and hierarchical memory construction. LifeStream models the temporal evolution of visual tokens and shows that their contributions to downstream reasoning vary with their lifecycle states. Guided by this insight, it organizes tokens into a hierarchical memory structure: informative non-stable tokens are retained in an archive memory, while long-range history is progressively compressed into a global memory according to state-specific priorities. This design continuously preserves high-value information under limited token budgets, with lightweight memory updates as new frames arrive and no query-time retrieval. Extensive experiments demonstrate that LifeStream discards over 80% of visual tokens while improving performance. It improves LLaVA-OV-7B and Qwen3-VL-8B by 4.4 and 3.5 percentage points on StreamingBench, respectively, with consistent gains over both backbones on one additional online benchmark and four long-video understanding benchmarks.

## 1 Introduction

The rapid growth of streaming video applications, such as live video assistants [1–4], robotics [5], and embodied systems [6–8], has made online video understanding an increasingly important capability for Video Large Language Models (VLLMs) [9–16]. In these scenarios, a model must process visual input continuously and remain responsive to user queries at any moment, rather than waiting until the full video has been observed. This requirement distinguishes online video understanding from the standard offline setting, where the entire video is available before inference. It also introduces a key challenge: the model must continuously construct compact and efficient visual memory without knowing future user queries.

Under this setting, the central problem is how to preserve useful historical information within a limited memory budget. As the visual context grows, visual tokens accumulate rapidly, making it computationally and storage-wise impractical to retain all information. Therefore, building a strong

†Corresponding author: J. Ge (gejn@mail.ustc.edu.cn).

40th Conference on Neural Information Processing Systems (NeurIPS 2026).




Figure 1: **LifeStream vs. TimeChat-Online.** TimeChat-Online uses coarse binary keep/drop decisions, which may miss key motion cues and limit progressive compression. LifeStream assigns lifecycle states before compression and builds hierarchical memory to retain high-value tokens under limited budgets.

VLLM hinges on compressing redundant visual information while preserving as much historical evidence useful for downstream reasoning as possible. Prior works have extensively explored token selection [17–23], especially query-based methods that filter relevant visual tokens according to the given question. However, in online video understanding, memory must be constructed before user queries arrive, making such query-dependent methods difficult to directly adapt. To meet the requirements of online scenarios, existing methods can be roughly divided into two categories: KV-cache-based methods [4, 24–26], which manage and compress cached key-value states during streaming inference, and vision-token reduction methods [27, 28], which reduce the visual tokens fed into the LLM based solely on visual content. The former are often less interpretable [29–31] and highly coupled with specific LLM architectures and caching mechanisms. In contrast, token compression from a purely visual perspective is more intuitive and offers better model flexibility. A line of redundancy-based vision-token reduction methods [27, 32, 33] estimates token importance from temporal similarity between the current frame and previously observed frames. This often reduces token selection to a coarse irreversible decision, where tokens are either discarded as redundant or retained as useful evidence. However, as shown in Fig. 1, past-similarity-only pruning may directly drop object-level motion cues, such as the moving person on the boat. Meanwhile, under a fixed memory budget, uniformly prioritized retained tokens make subsequent eviction or compression prone to losing informative evidence. We argue that the root cause lies in the fact that the importance of visual tokens often depends on their temporal evolution. For example, persistently appearing visual content tends to become redundant in subsequent frames, whereas transient changes or events, though less frequent, can be crucial for downstream reasoning. More importantly, such differences in token importance cannot be reliably determined at the moment a token appears, but require assessing its temporal continuity.

To examine this temporal-evolution hypothesis, we revisit visual tokens from a lifecycle perspective. Specifically, rather than relying only on historical information, we use a short temporal window to jointly consider a token's relationship with its past and future neighborhoods, thereby characterizing its temporal behavior. This bidirectional temporal view naturally groups tokens into distinct temporal roles. Specifically, we divide visual tokens into four mutually exclusive states according to their similarity to neighboring past and future frames: `onset` denotes newly appearing tokens with low past but high future similarity; `offset` denotes disappearing tokens with high past but low future similarity; `fleeting` denotes transient evidence with low similarity to both past and future frames; and `stable` denotes persistent redundant tokens with high similarity to both sides.

To examine whether this state partition reflects the actual value of different tokens, we conduct empirical analyses on StreamingBench [34] and MLVU [35]. We first evaluate the impact of each state by removing tokens of one state while keeping the remaining tokens.

As shown in Fig. 2, `stable` tokens constitute 61.7% of visual tokens on MLVU, yet removing them causes no performance drop. Removing less frequent non-`stable` tokens causes larger drops: `fleeting` tokens account for only 5.4%, but their removal reduces accuracy by 4.5 percentage points. StreamingBench-RT shows the same pattern: negligible impact from removing `stable` tokens, but larger drops from removing non-`stable` tokens. These results indicate that the downstream value of visual tokens is not proportional to their quantity, but is closely related to their lifecycle states. We further analyze the attention distribution of tokens in different states. In the example shown in Fig. 3, the model assigns higher attention to non-`stable` tokens, particularly `fleeting` tokens, consistent with the performance changes observed in the removal experiments. Taken together, these results suggest that visual token value should be modeled with fine-grained lifecycle states. Moreover, even within non-`stable` tokens, different states exhibit clear differences in importance. Therefore, state information can be utilized to design more fine-grained token prioritization strategies, thereby enabling more effective token selection and progressive compression under limited budget constraints.

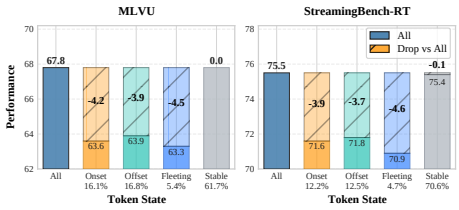



Figure 2: **Token lifecycle states have unequal downstream utility.**

Motivated by the lifecycle token utility revealed above, we propose **LifeStream**, a training-free and plug-and-play framework for fine-grained token discrimination and hierarchical memory construction. Instead of compressing each frame immediately upon arrival, LifeStream delays role assignment until the frame leaves the recent buffer, allowing token selection to exploit short-term future observations while remaining compatible with streaming inference. Built upon lifecycle states, LifeStream organizes visual memory into a recent buffer, a lifecycle-aware archive memory, and a compact global memory. The archive memory retains all non-`stable` tokens to preserve informative historical evidence, while the global memory further uses state-specific priorities to preserve highly informative `fleeting` tokens and compact boundary-related `onset` and `offset` tokens. This hierarchical design enables progressive memory compression beyond the coarse `Keep` / `Drop` decision used in prior binary methods. We conduct extensive experiments on both online and long-video understanding benchmarks. LifeStream improves both backbones across the evaluated benchmarks using preconstructed visual memory, without query-conditioned retrieval or token selection.

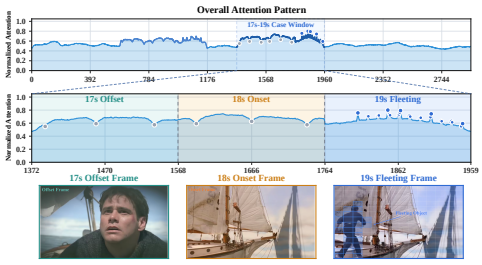



Figure 3: **Attention patterns on different lifecycle state tokens.**

In summary, the main contributions of this work are as follows:

- We show that retained tokens have unequal importance for downstream question answering, motivating finer-grained token state modeling.
- We propose LifeStream, a training-free and plug-and-play framework that models visual tokens as `onset`, `offset`, `fleeting`, and `stable`, and builds a hierarchical memory to progressively compress retained tokens under limited budgets.
- Across six online and long-video understanding benchmarks, LifeStream discards over 80% of visual tokens and improves LLaVA-OV-7B and Qwen3-VL-8B over their base models by an average of 4.2 and 2.4 percentage points, respectively, while keeping memory construction independent of incoming queries.

## 2 Related Work

**Multimodal Large Language Models.** Multimodal large language models (MLLMs) align visual encoders with language models for image understanding [14, 36–42] and video question answering [9, 10, 43–47]. Recent families, including LLaVA-OneVision [10], Qwen-VL [11–13], and InternVL [15, 16], improve multimodal perception and reasoning, but long or streaming videos still incur growing visual context costs. LifeStream complements these backbone advances with training-free visual memory construction.

**Long video understanding.** Long video understanding requires preserving key evidence over extended temporal spans, but dense visual tokens bring high context and computation costs. Existing methods mainly address this issue through context extension, visual compression, and keyframe selection. LongVA [48] transfers long-context capability from language to vision, while LongVILA [49] extends video context with long-video training and sequence parallelism. LongVU [50] reduces redundant tokens via spatiotemporal adaptive compression. Video-XL [51] builds compact visual summaries for hour-scale videos, and Video-XL-2 [52] further reduces inference cost with task-aware KV sparsification and chunk-wise inference. AKS [53] selects frames by query relevance and temporal coverage, while VISTA [54] improves long-duration and high-resolution understanding through data construction. However, these methods often assume full-video access or depend on offline, global, or query-aware selection, which limits their applicability to online video scenarios.

**Online video understanding.** Existing methods address online video understanding from different perspectives. VideoLLM-online [1] builds an online video-language model for streaming dialogue. Flash-VStream [55] introduces a memory-based framework for real-time long video streams, while Dispider [56], ViSpeak [57], and Thinking-QwenVL [58] further explore active interaction, visual instruction feedback, and evidence-aligned response timing. Another line of work focuses on KV-cache memory. ReKV [59] retrieves in-context video KV caches for streaming question answering; StreamKV [24] combines segment-based KV retrieval and compression; StreamMem [25], LiveVLM [60], and HERMES [26] construct query-agnostic or hierarchical KV memories for efficient long-stream processing. In contrast, TimeChat-Online [27] and StreamForest [28] operate more directly on visual information by reducing redundant visual tokens or maintaining persistent event memory.

## 3 Method

### 3.1 Lifecycle Streaming Video Formulation

Online video understanding requires the model to process a video stream continuously under a strict causal setting, where at time step $\tau$ only the video prefix $V_{\leq\tau}$ is available and no future visual information can be accessed. This causal constraint makes fine-grained temporal state assignment for visual tokens particularly challenging: relying only on past observations, the model often struggles to assess the utility of retained tokens and may discard critical visual evidence under the same budget. To address this issue, we introduce a recent buffer that stores the most recent $N$ frames while preserving the online input order and causal inference process. Rather than assigning token states immediately upon a frame's arrival, LifeStream delays this decision until the frame exits the recent buffer. By that time, frame $I_t$ can be analyzed with access to a limited short-term future, enabling lifecycle state assignment under a richer local temporal context.

Specifically, let the recent buffer store the latest $N_r$ frames. Once a frame reaches the boundary of the recent buffer, it becomes eligible for temporal state analysis and subsequent processing. For such a frame $I_t$, we construct a local observation window $W(t) = \{t-\delta_p,\ \ldots,\ t,\ \ldots,\ t+\delta_f\}$ to capture the short-horizon temporal context for token state assignment, where $\delta_p$ and $\delta_f$ denote the numbers of past and future frames considered, respectively. In the default setting, we use $(\delta_p, \delta_f) = (1, 2)$, which gives $W(t) = \{t-1,\ t,\ t+1,\ t+2\}$. This short-horizon formulation introduces only limited local temporal interaction, thereby preserving the efficiency required for online inference while improving the accuracy of temporal state assignment. As shown in Sec. 4.4, using two future frames achieves the best trade-off between performance and overhead.

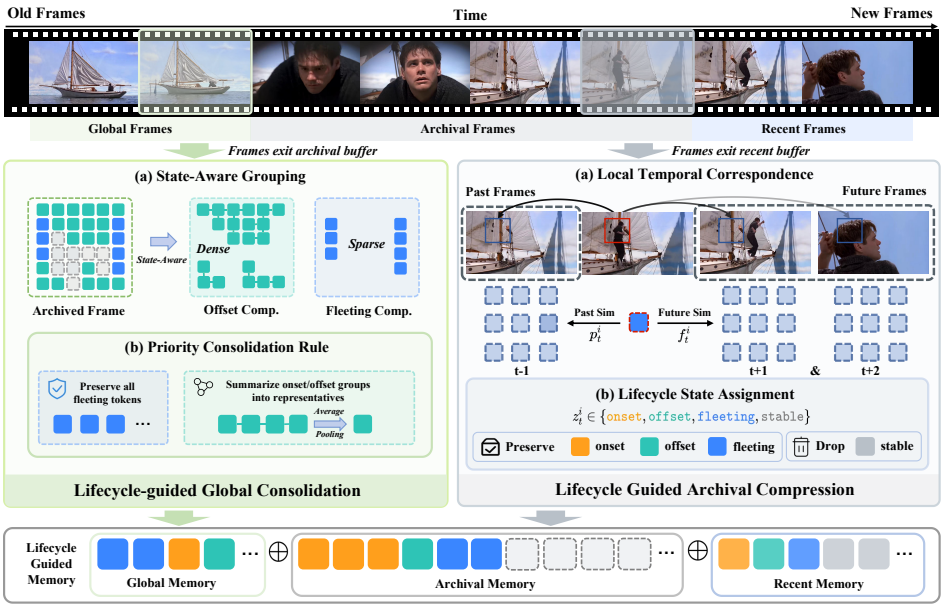



Figure 4: **Overview of LifeStream.** LifeStream maintains a recent buffer, a lifecycle-aware archive, and a compact global memory. When a frame exits the recent buffer, short-horizon lifecycle inference assigns token roles for archival compression, and lifecycle-guided global consolidation summarizes archived tokens into compact long-range memory.

### 3.2 Lifecycle Token Modeling

**Algorithm 1** Lifecycle state assignment from short-horizon temporal correspondence

**Require:** Local temporal correspondence scores $\{m^i_{t-1\to t},\ m^i_{t+1\to t},\ m^i_{t+2\to t}\}$, recency weights $\alpha_1, \alpha_2$
**Ensure:** Lifecycle state $z^i_t$ and confidence $\rho^i_t$

```
Compute normalized descriptors p_t^i and f_t^i according to Eq. 2
s_onset ← (1 − p_t^i) f_t^i
s_stable ← p_t^i f_t^i
s_offset ← p_t^i (1 − f_t^i)
s_fleeting ← (1 − p_t^i)(1 − f_t^i)
ρ_t^i ← max{s_onset, s_stable, s_offset, s_fleeting}
z_t^i ← arg max_{r∈S} s_r^i
```

To provide finer-grained guidance for memory retention under a limited memory budget, we model the lifecycle states of visual tokens based on their two-dimensional temporal characteristics: **(1) Past continuity**; **(2) Future continuity**. When frame $I_t$ is about to exit the recent buffer, we compute token-level temporal correspondence within its local temporal window $W(t)$. Let $X_t = \{x^i_t\}_{i=1}^{N_t}$ denote the visual tokens of frame $I_t$. Specifically, for each token $x^i_t$, we search within a spatially aligned $3\times3$ neighborhood in each frame of $W(t)$ for its most similar token, and define the resulting maximum similarity as its local temporal correspondence score:

$$m^i_{t'\to t} = \max_{j\in\mathcal{N}_{3\times3}(i)} \cos\left(x^i_t,\ x^j_{t'}\right), \quad t' \in \{t-1, t+1, t+2\} \tag{1}$$

For each token $x^i_t$, we summarize its short-horizon temporal behavior using two descriptors derived from $m^i_{t'\to t}$: a past descriptor $p^i_t$ and a future descriptor $f^i_t$, defined as

$$p^i_t = (m^i_{t-1\to t} + 1)/2, \qquad f^i_t = (\alpha_1 m^i_{t+1\to t} + \alpha_2 m^i_{t+2\to t} + 1)/2, \tag{2}$$

where $\alpha_1 + \alpha_2 = 1$. To account for recency bias, we assign a larger weight to the nearer future frame by setting $\alpha_1 > \alpha_2$. In practice, we use $\alpha_1 = 0.7$ and $\alpha_2 = 0.3$.

Based on the two temporal descriptors $p_t^i$ and $f_t^i$, LifeStream maps each token $x_t^i$ into a compact lifecycle state space $S$:

$$\mathcal{S} = \{\texttt{onset}, \texttt{stable}, \texttt{offset}, \texttt{fleeting}\}, \quad (\rho_t^i, z_t^i) = \mathrm{AssignRole}(p_t^i, f_t^i), \tag{3}$$

where the four states correspond to distinct short-horizon temporal patterns: `onset` is characterized by weak past continuity but clear future support; `offset` has strong past continuity but weak future support; `fleeting` lacks strong support from either side of the temporal axis, often capturing transient fine-grained cues or moving objects; and `stable` exhibits strong continuity on both temporal sides. To make the state assignment process more transparent, we adopt a simple multiplicative scoring rule, where each state is defined by its core temporal pattern. Let $s_r^i$ denote the score of assigning token $x_t^i$ to lifecycle state $r \in \mathcal{S}$. The detailed scoring rules are summarized in Algorithm 1.

### 3.3 Lifecycle Guided Memory

**Lifecycle Guided Archival Compression** A key motivation of LifeStream is that distinguishing visual tokens solely by whether they are redundant is still insufficient to satisfy the compression demands of online or long video understanding. Even after removing most temporally redundant `stable` tokens, the remaining tokens continue to accumulate as the video stream grows, and can still exceed the available memory budget under long-context settings. According to our empirical analysis in Fig. 2 and Fig. 3, tokens in different lifecycle states exhibit different levels of utility for downstream reasoning. Based on this observation, LifeStream adopts a state-aware retention strategy in archival compression, where the retention priority is defined as follows:

$$\omega_{\texttt{fleeting}} > \omega_{\texttt{onset}} = \omega_{\texttt{offset}} > \omega_{\texttt{stable}}. \tag{4}$$

In the archival memory, we retain all non-`stable` tokens and record their lifecycle state distribution, which is later used to further bias the compression process in the global memory:

$$\pi_t^r = \frac{\sum_{i=1}^{N_t} \mathbf{1}[z_t^i = r]}{\sum_{i=1}^{N_t} \mathbf{1}[z_t^i \in \mathcal{S}_{\mathrm{ns}}]}, \qquad r \in \mathcal{S}_{\mathrm{ns}}, \tag{5}$$

where $\mathcal{S}_{\mathrm{ns}} = \{\texttt{fleeting}, \texttt{onset}, \texttt{offset}\}$ denotes the set of non-`stable` lifecycle states, $N_t$ denotes the total number of visual tokens in frame $I_t$, $z_t^i$ is the lifecycle state assigned to the $i$-th token in frame $I_t$, $\mathbf{1}[\cdot]$ is the indicator function, and $\pi_t^r$ represents the proportion of tokens belonging to state $r$ among all non-`stable` tokens in frame $I_t$.

**Lifecycle Guided Global Consolidation** After archival compression has removed most `stable` tokens, the long-range historical frames written into the archival memory still contain different types of non-`stable` evidence. Considering the higher informational value of `fleeting` tokens, LifeStream preserves them during global compression. In contrast, `onset` and `offset` tokens mainly reflect semantic transitions at temporal boundaries. In practice, such frames are typically dominated by either `onset` or `offset`, rather than both.

When $\max(\pi_t^{\texttt{onset}}, \pi_t^{\texttt{offset}}) > \gamma$, LifeStream compresses the dominant boundary state tokens. Specifically, within local 3×3 neighborhoods, vision tokens that belong to the same dominant boundary state are grouped into local connected sets $\{\mathcal{G}_{t,m}\}_{m=1}^{M_t}$, and each set is merged into a single token by average pooling:

$$\bar{x}_{t,m} = \frac{1}{|\mathcal{G}_{t,m}|} \sum_{x_t^i \in \mathcal{G}_{t,m}} x_t^i, \quad m = 1, \ldots, M_t. \tag{6}$$

This produces a more compact long-range representation for transition heavy regions, while more implementation details are deferred to Appendix B.

## 4 Experiments

### 4.1 Implementation Details

In our experiments, we implement LifeStream on two strong open-source MLLMs, LLaVA-OneVision-7B [10] and Qwen3-VL-8B [13]. We sample videos at 1 fps, use a chunk size of 32 frames, and feed one chunk at a time to simulate streaming input. For LLaVA-OneVision, we follow

the official setting and encode each video frame into 196 vision tokens for fair comparison. For Qwen3-VL, since its technical report does not disclose streaming evaluation details, we conduct a series of controlled experiments under matched settings and cap the number of tokens per frame at 196. For the hierarchical memory in LifeStream, we set the recent buffer size to 8 frames, the mid-term memory to 128 frames, and store all remaining historical frames in the global memory for long-horizon lifecycle retention. For global consolidation, we set $\gamma = 0.7$ and pool the dominant boundary state only when its proportion among non-stable tokens exceeds this threshold. To efficiently evaluate offline benchmarks, we set the maximum number of sampled video frames to 1024. For all videos, the maximum total number of visual tokens submitted to the LLM at query time is 8192. If the visual token count of a video exceeds this predefined budget, we further apply average sampling within each memory level to constrain it to the specified token limit. All experiments are conducted on $8 \times$ A100 GPUs.

### 4.2 Benchmarks

To further evaluate LifeStream, we conduct extensive experiments on both online and long-video understanding benchmarks. For online video understanding, we focus on StreamingBench [34] and OVO-Bench [61], which are designed to evaluate real-time streaming capabilities under strict causal constraints. Our evaluation primarily focuses on the real-time understanding subtasks shared by both. For long video understanding, we evaluate on four authoritative benchmarks: EgoSchema [62], MLVU [35], LongVideoBench [63], and VideoMME [64]. These benchmarks cover a wide range of video durations, from short clips to hour-level videos, allowing us to assess the generality of LifeStream across diverse video scenarios.

Table 1: **Results on online video understanding benchmarks.** We report the accuracy (%) on the real-time tasks of StreamingBench and OVO-Bench. † indicates the reproduced results. The meaning of each subtasks are provided in Appendix A.

| Method | Size | Frames | OVO-Bench real-time | | | | | | | StreamingBench real-time | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | OCR | ACR | ATR | STU | FPD | OJR | Avg. | OP | CR | CS | ATP | EU | TR | PR | SU | ACP | CT | Avg. |
| **Human** | | | | | | | | | | | | | | | | | | | | |
| Human | – | – | 94.0 | 92.6 | 94.8 | 92.7 | 91.1 | 94.0 | 93.2 | 89.5 | 92.0 | 93.6 | 91.5 | 95.7 | 92.5 | 88.0 | 88.8 | 89.7 | 91.3 | 91.5 |
| **Proprietary Models** | | | | | | | | | | | | | | | | | | | | |
| Gemini 1.5 Pro [42] | – | 1 fps | 85.9 | 67.0 | 79.3 | 58.4 | 63.4 | 62.0 | 69.3 | 79.0 | 80.5 | 83.5 | 79.7 | 80.0 | 84.7 | 77.8 | 64.2 | 72.0 | 48.7 | 75.7 |
| GPT-4o [41] | – | 64 | 69.8 | 64.2 | 71.6 | 51.1 | 70.3 | 59.8 | 64.5 | 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** | | | | | | | | | | | | | | | | | | | | |
| LongVA [48] | 7B | 128 | – | – | – | – | – | – | – | 70.0 | 63.3 | 61.2 | 70.9 | 62.7 | 59.5 | 61.1 | 53.7 | 54.7 | 34.7 | 60.0 |
| LongVU [50] | 7B | 1 fps | 55.7 | 49.5 | 59.5 | 48.3 | 68.3 | 63.0 | 57.4 | – | – | – | – | – | – | – | – | – | – | – |
| InternVL2 [14] | 8B | 64/16 | 67.1 | 60.6 | 63.8 | 46.1 | 68.3 | 56.5 | 60.4 | 68.1 | 60.9 | 69.4 | 77.1 | 67.7 | 62.9 | 59.3 | 53.3 | 55.0 | 56.5 | 63.7 |
| LLaVA-OneVision [10] | 7B | 64/32 | 66.4 | 57.8 | 73.3 | 53.4 | 71.3 | 62.0 | 64.0 | 80.4 | 74.2 | 76.0 | 80.7 | 72.7 | 71.7 | 67.6 | 65.5 | 65.7 | 45.1 | 71.1 |
| Qwen2.5-VL [12] | 7B | 1 fps | 79.2 | 53.2 | 67.2 | 51.7 | 71.3 | 57.1 | 63.3 | 78.3 | 80.5 | 79.8 | 82.4 | 75.5 | 80.4 | 74.1 | 62.6 | 67.6 | 51.1 | 73.9 |
| Qwen3-VL† [13] | 8B | 1 fps | 77.9 | 69.7 | 76.7 | 54.5 | 67.3 | 66.8 | 68.8 | 80.4 | 80.5 | 85.8 | 82.8 | 74.8 | 80.4 | 78.7 | 67.5 | 70.5 | 49.7 | 75.9 |
| **Open-source Online MLLMs (Training-Based)** | | | | | | | | | | | | | | | | | | | | |
| VideoLLM-online [CVPR 24] [1] | 8B | 2 fps | 8.1 | 23.9 | 12.1 | 14.0 | 45.5 | 21.2 | 20.8 | 39.1 | 40.1 | 34.5 | 31.1 | 46.0 | 32.4 | 31.5 | 34.2 | 42.5 | 27.9 | 36.0 |
| Flash-VStream [ARXIV 24] [55] | 7B | 1 fps | 25.5 | 32.1 | 29.3 | 33.7 | 29.7 | 28.8 | 29.9 | 25.9 | 43.6 | 24.9 | 23.9 | 27.3 | 13.1 | 18.5 | 25.2 | 23.9 | 48.7 | 23.2 |
| Dispider [CVPR 25] [56] | 7B | 1 fps | 57.7 | 49.5 | 62.1 | 44.9 | 61.4 | 51.6 | 54.5 | 74.9 | 75.5 | 74.1 | 73.1 | 74.4 | 59.9 | 76.1 | 62.9 | 62.2 | 45.8 | 67.6 |
| ViSpeak [ICCV 25] [57] | 7B | 1 fps | 75.2 | 58.7 | 71.6 | 51.1 | 74.3 | 66.9 | 66.3 | 79.8 | 88.3 | 83.3 | 81.1 | 76.4 | 75.1 | 70.4 | 65.9 | 77.3 | 34.2 | 74.4 |
| TimeChat-Online [Qwen2.5-VL] [ACMMM 25] [27] | 7B | 1 fps | 74.5 | 48.6 | 68.1 | 48.3 | 69.3 | 59.8 | 61.4 | 80.8 | 79.7 | 80.8 | 83.3 | 74.8 | 78.8 | 78.7 | 64.2 | 68.8 | 58.0 | 75.3 |
| Thinking-QwenVL [Qwen2.5-VL] [ICLR 26] [58] | 7B | 1 fps | 74.1 | 57.2 | 68.1 | 55.3 | 75.0 | 58.3 | 64.7 | 70.3 | 66.7 | 80.0 | 78.0 | 79.3 | 68.7 | 78.3 | 68.2 | 72.3 | 52.4 | 71.6 |
| **Open-source Online MLLMs (Training-Free)** | | | | | | | | | | | | | | | | | | | | |
| ReKV [LLaVA-OV] [ICLR 25] [59] | 7B | 0.5 fps | – | – | – | – | – | – | – | 74.4 | 78.9 | 78.6 | 77.1 | 68.3 | 67.9 | 67.6 | 62.6 | 64.3 | 44.6 | 69.1 |
| StreamKV [LLaVA-OV] [AAAI 26] [24] | 7B | 0.5 fps | – | – | – | – | – | – | – | 74.7 | 78.1 | 87.7 | 79.4 | 70.8 | 67.6 | 70.4 | 64.6 | 64.0 | 45.1 | 71.0 |
| LiveVLM [LLaVA-OV] [ARXIV 25] [60] | 7B | 0.5 fps | – | – | – | – | – | – | – | 81.5 | 78.1 | 83.3 | 79.1 | 69.6 | 74.1 | 75.0 | 69.1 | 67.7 | 40.4 | 72.9 |
| HERMES [LLaVA-OV] [ARXIV 26] [26] | 7B | 0.5 fps | 72.5 | 62.4 | 74.1 | 50.6 | 73.3 | 65.2 | 66.3 | 79.0 | 81.3 | 87.7 | 80.2 | 69.2 | 72.0 | 73.2 | 66.3 | 69.4 | 43.5 | 73.2 |
| **LifeStream +LLaVA-OneVision** | **7B** | **1 fps** | 76.5 | 67.9 | 75.9 | 55.6 | 68.3 | 69.0 | **68.9** +4.9↑ | 79.3 | 79.7 | 85.5 | 83.5 | 77.4 | 73.5 | 75.0 | 67.5 | 72.8 | 53.4 | **75.5** +4.4↑ |
| **LifeStream +Qwen3-VL** | **8B** | **1 fps** | 79.2 | 71.6 | 79.3 | 57.3 | 69.3 | 70.1 | **71.1** +2.3↑ | 83.7 | 80.5 | 87.1 | 87.1 | 74.8 | 82.9 | 83.3 | 75.6 | 75.1 | 54.4 | **79.4** +3.5↑ |

### 4.3 Main Results

**Results on Online Video Benchmarks** Table 1 compares LifeStream on the real-time subsets of OVO-Bench [61] and StreamingBench [34]. With LLaVA-OneVision [10], LifeStream improves StreamingBench accuracy from 71.1% to 75.5% and OVO-Bench from 64.0% to 68.9%, outper-

forming the reported Qwen2.5-VL-based TimeChat-Online [27] and Thinking-QwenVL [58] results. During memory construction, it discards nearly 80% of visual tokens while maintaining a query-time visual token cost comparable to the backbone. Gains of 8.3 and 7.0 percentage points on CT and OJR, respectively, highlight its ability to retain fine-grained evidence. Consistent improvements with Qwen3-VL [13] further support the applicability of lifecycle-guided memory across backbones.

Table 2: **Results on long video understanding benchmarks.** We report accuracy (%) on EgoSchema, MLVU-dev(mc), LongVideoBench, and VideoMME (w/o subtitles). † indicates the reproduced results.

| Method | Size | Frames | EgoSchema | MLVU | LongVideo Bench | VideoMME | |
|---|---|---|---|---|---|---|---|
| | | | | | | Medium | Long |
| **Video Length** | – | – | 3min | 3~120min | 8sec~60min | 4~15min | 30~60min |
| **Open-source Offline Video LLMs** | | | | | | | |
| Video-LLaVA [9] | 7B | 8 | 38.4 | 47.3 | – | 38.0 | 36.2 |
| VideoChat2 [47] | 7B | 16 | 54.4 | 47.9 | – | 37.0 | 33.2 |
| LongVA [48] | 7B | 128 | – | 56.3 | – | 50.4 | 46.2 |
| Kangaroo [46] | 8B | 64 | – | 61.0 | – | 55.3 | 46.6 |
| LLaVA-OneVision† [10] | 7B | 32 | 58.5 | 64.7 | 53.9 | 54.7 | 46.2 |
| VideoXL [51] | 7B | 128 | – | 64.9 | – | 53.2 | 49.2 |
| Qwen3-VL† [13] | 8B | 1 fps | 64.2 | 70.1 | 55.6 | 56.6 | 52.3 |
| **Open-source Online Video LLMs** | | | | | | | |
| MovieChat [CVPR 24] [65] | 7B | 2048 | 53.5 | 25.8 | – | – | 33.4 |
| Dispider [CVPR 25] [56] | 7B | 1 fps | 55.6 | 61.7 | – | 53.7 | 49.7 |
| TimeChat-Online [Qwen2.5-VL] [ACMMM 25] [27] | 7B | 1 fps | – | 65.4 | 57.7 | – | 49.2 |
| LiveVLM [LLaVA-OV] [ARXIV 25] [60] | 7B | 0.5/0.2 fps | 59.0 | 66.3 | – | 56.4 | 48.8 |
| StreamMem [LLaVA-OV] [ARXIV 25] [25] | 7B | 4.0/0.5 fps | 63.0 | 66.9 | – | 56.6 | 50.1 |
| HERMES [LLaVA-OV] [ARXIV 26] [26] | 7B | 0.5 fps | 60.3 | – | – | 54.9 | 49.1 |
| **LifeStream +LLaVA-OneVision** | **7B** | **1 fps** | **63.6** +5.1↑ | **67.8** +3.1↑ | **58.2** +4.3↑ | **56.8** +2.1↑ | **51.1** +4.9↑ |
| **LifeStream +Qwen3-VL** | **8B** | **1 fps** | **67.1** +2.9↑ | **72.3** +2.2↑ | **57.3** +1.7↑ | **58.1** +1.5↑ | **54.6** +2.3↑ |

**Results on Long Video Benchmarks** Table 2 evaluates whether the streaming-oriented memory also preserves evidence over longer videos. With LLaVA-OneVision [10], LifeStream improves accuracy by 5.1, 3.1, and 4.3 percentage points on EgoSchema [62], MLVU [35], and LongVideoBench [63], respectively. On VideoMME [64], it improves both medium and long subsets, including a 4.9-point gain on 30–60-minute videos. In particular, accuracy on the long subset rises from 46.2% to 51.1%, suggesting that compact historical representations retain useful evidence beyond the recent observation window. These results extend the benefits of the streaming memory to offline evaluation over longer temporal spans. Qwen3-VL [13] also benefits, indicating that lifecycle-guided evidence retention generalizes across backbones and temporal scales.

### 4.4 Ablation Studies

Unless otherwise specified, all ablation studies are conducted on LLaVA-OneVision-7B [10] under the same experimental setup as Sec. 4.1. We use the observation window $W(t) = \{t-1,\ t,\ t+1,\ t+2\}$ and a maximum token budget of 8192.

**Effect of Lifecycle State Modeling** We evaluate the effectiveness of the Lifecycle states by comparing different state partitions. Baseline denotes the binary classification of `stable` and non-`stable`, following DTD [27]. `fleeting` denotes retaining the `fleeting` and `stable` states in Algorithm 1. `onset` + `offset` retains the `onset`, `offset`, and `stable` states, while `fleeting` + `onset` + `offset` represents the full Lifecycle state setting. As shown in Table 3, introducing `onset` and `offset` brings

Table 3: **Ablation results of lifecycle state modeling.**

| Lifecycle States Used | MLVU | VideoMME | StreamingBench | Score |
|---|---|---|---|---|
| | M-Avg | Long | Real-time | Avg. |
| Baseline | 66.0 | 49.8 | 73.2 | 63.0 |
| `fleeting` | 64.2 | 47.1 | 69.8 | 60.4 |
| `onset` + `offset` | 66.3 | 50.1 | 73.6 | 63.3 |
| `fleeting` + `onset` + `offset` | **67.8** | **51.1** | **75.5** | **64.8** |

consistent gains across MLVU [35], VideoMME-Long [64], and the real-time subset of StreamingBench [34], improving the average score from 63.0% to 63.3%. Further adding `fleeting` raises the average score to 64.8%, showing that transient evidence provides additional benefits beyond boundary cues. In contrast, partitioning states only into `stable` and `fleeting` leads to less stable performance, suggesting that `fleeting` tokens alone provide overly discrete evidence without explicit boundary-state cues.

**Effect of Lifecycle Guided Memory** We ablate the three components of Lifecycle Guided Memory, including recent memory $R$, archival memory $A$, and global memory $G$. As shown in Table 4, MLVU benefits more from broader temporal coverage. Using only $R$ leads to a clear drop, while adding $A$ or $G$ substantially improves performance by introducing longer-range visual evidence. StreamingBench [34] shows a stronger recency bias, where $R$ alone already achieves competitive results, and using only $A$, $G$, or $A$+$G$ performs worse due to the lack of recent context. Combining $R$ with either $A$ or $G$ recovers performance, and the full Lifecycle Guided Memory achieves the best overall results across both long video and online video understanding. These complementary trends explain the value of the hierarchy: recent memory preserves the immediate context, while archive and global memories extend temporal coverage.

Table 4: **Ablation results of lifecycle guided memory.** $R$ denotes the Recent buffer, $A$ denotes the Archive memory, and $G$ denotes the Global memory.

| Memory | | | MLVU | StreamingBench | Score |
|---|---|---|---|---|---|
| $R$ | $A$ | $G$ | M-Avg | Real-time | Avg. |
| ✓ | ✗ | ✗ | 60.4 | 74.8 | 67.6 |
| ✗ | ✓ | ✗ | 66.3 | 72.3 | 69.3 |
| ✗ | ✗ | ✓ | 66.1 | 71.8 | 69.0 |
| ✓ | ✓ | ✗ | 65.7 | 74.6 | 70.2 |
| ✓ | ✗ | ✓ | 66.2 | 75.2 | 70.7 |
| ✗ | ✓ | ✓ | 67.2 | 72.9 | 70.1 |
| ✓ | ✓ | ✓ | **67.8** | **75.5** | **71.7** |

**Effect of the Observation Window** We further ablate the future observation window by fixing $\delta_p = 1$ in $W(t) = \{t - \delta_p, \ldots, t, \ldots, t + \delta_f\}$ and varying $\delta_f$. As shown in Table 5, introducing future frames consistently improves performance, indicating that short-term future observations enable more fine-grained token retention. The best trade-off is achieved at $\delta_f = 2$, which improves the average score from 69.6% to 71.7% with moderate latency overhead. Extending the window to $\delta_f = 3$ brings higher latency but no further accuracy gain. These future observations are relative to the historical frame being compressed and are already available when it leaves Recent Memory. Queries can therefore use the observed video prefix without waiting for additional frames. Unlike retrieval-based methods [24, 59], LifeStream maintains memory as frames arrive and requires no query-time retrieval.

Table 5: **Ablation results of the future window.**

| Future Frames ($\delta_f$) | MLVU | StreamingBench | Score | Latency |
|---|---|---|---|---|
| | M-Avg | Real-time | Avg. | ms |
| $\delta_f = 0$ (Past Only) | 66.0 | 73.2 | 69.6 | **1734** |
| $\delta_f = 1$ | 66.5 | 74.6 | 70.6 | 1787 |
| $\delta_f = 2$ | **67.8** | **75.5** | **71.7** | 1826 |
| $\delta_f = 3$ | 67.4 | 75.2 | 71.3 | 1885 |

**Sensitivity to Fixed Parameters** We vary $(\alpha_1, \alpha_2)$ and $\gamma$ individually, keeping other settings unchanged. The weights control the relative contributions of the two future references, while $\gamma$ determines when a dominant boundary state is consolidated. Table 6 reports accuracy and Global tokens/frame after consolidation, before query-time budget control. The default $(\alpha_1, \alpha_2) = (0.7, 0.3)$ and $\gamma = 0.7$ performs best among the tested fixed settings. Lowering $\gamma$ to 0.5 makes consolidation more aggressive, reducing retained tokens from 11.4 to 10.8 at the cost of accuracy. Raising it to 0.9 retains 15.1 tokens but reduces accuracy on both benchmarks, indicating that retaining more tokens alone does not guarantee better performance. Using only the nearer reference also lowers accuracy. Equal reference weights yield a smaller decline, whereas favoring the distant reference causes a larger drop. Together, these results favor combining both references with greater weight on the nearer one, and support the default balance between evidence retention and compression.

Table 6: **Sensitivity to fixed parameters.**

| Setting | MLVU | StreamingBench | Global |
|---|---|---|---|
| | M-Avg | Real-time | tokens/frame |
| **Default** | **67.8** | **75.5** | 11.4 |
| $\gamma = 0.5$ | 66.9 | 74.5 | 10.8 |
| $\gamma = 0.9$ | 65.2 | 73.7 | 15.1 |
| $(\alpha_1, \alpha_2) = (1.0, 0.0)$ | 66.5 | 74.6 | **10.7** |
| $(\alpha_1, \alpha_2) = (0.5, 0.5)$ | 67.5 | 75.2 | 12.6 |
| $(\alpha_1, \alpha_2) = (0.3, 0.7)$ | 66.2 | 72.1 | 12.5 |

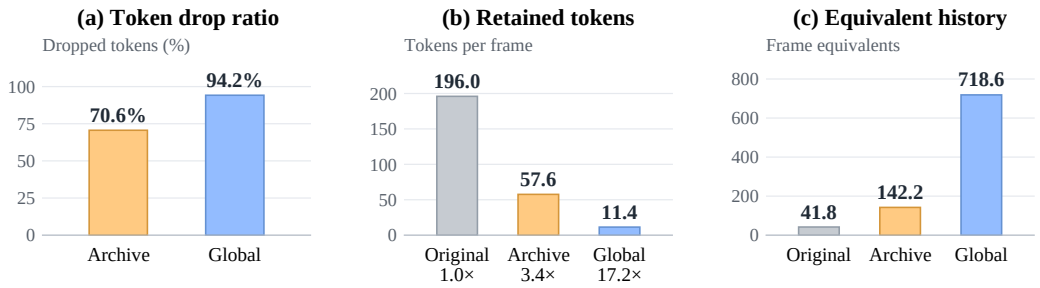



Figure 5: **Visual-token efficiency of LifeStream.** (a) Token reduction after archival compression and global consolidation. (b) Average retained tokens per frame and compression factors relative to uncompressed visual tokens. (c) Estimated equivalent history length under a fixed token budget.

### 4.5 Efficiency and Runtime Analysis

**Visual Token Efficiency** We measure visual-token efficiency on StreamingBench real-time under the main experimental settings and an 8,192-token budget. Figure 5 shows that Archive reduces the average tokens per frame from 196 to 57.6, and Global further reduces it to 11.4, removing 94.2% of the original tokens. Archive removes stable redundancy, while Global further consolidates older evidence according to lifecycle states. These counts correspond to estimated history lengths of 41.8, 142.2, and 718.6 frame equivalents, respectively, under the same budget. Thus, progressive compression allows a fixed token budget to represent a substantially longer history.

**Memory Maintenance Runtime** We spatially upsample and downsample frames of the same video to vary tokens per frame, then time memory maintenance on one NVIDIA A100 with $\delta_f = 2$, excluding visual encoding and QA inference. Table 7 shows that total update cost increases from 3.9 ms at 196 tokens/frame to 29.2 ms at 2,304 tokens/frame. Local matching grows approximately linearly, consistent with the $O(27ND) = O(ND)$ cost of three $3 \times 3$ correspondence passes for $N$ tokens of dimension $D$. The local lookup is vectorized across tokens and reuses cached past correspondences. With a fixed token count and reference window, its search cost does not grow with accumulated video history. Memory-update throughput ranges from 256.4 to 34.2 frames/s, exceeding the evaluated 1-fps input rate.

Table 7: **Memory-maintenance runtime at different visual token counts per frame.** Local passes combine three local correspondence searches. Other operations include lifecycle assignment and amortized Archive-to-Global consolidation. Total sums both components, and update rate is derived from this per-frame cost.

| Tokens/frame | Local passes (ms) | Other ops. (ms) | Total (ms) | Update rate (frames/s) |
|---|---|---|---|---|
| **196** | **2.3** | **1.6** | **3.9** | **256.4** |
| 576 | 6.4 | 2.1 | 8.5 | 117.6 |
| 1,024 | 11.3 | 2.8 | 14.1 | 70.9 |
| 2,304 | 25.1 | 4.1 | 29.2 | 34.2 |

## 5 Conclusion

LifeStream is a training-free, plug-and-play framework that uses visual token lifecycle states to guide hierarchical memory construction. Our empirical analyses show that token utility depends on temporal evolution: persistent content is often redundant, whereas transient evidence can be important for downstream reasoning. The archive retains informative non-stable tokens, while global consolidation preserves transient evidence and compresses boundary-related states. This lifecycle-aware organization enables progressive compression beyond binary keep/drop decisions. By retaining informative evidence and progressively compressing historical tokens, it improves streaming and long-video understanding across VLLM backbones under limited token budgets. The framework still relies on a predefined discrete state space and manually specified retention priorities and consolidation thresholds. Future work may explore adaptive state modeling and parameter selection to improve robustness across diverse streaming scenarios.

## Acknowledgments

This work is supported by the National Natural Science Foundation of China (62425114, 62121002, U23B2028), and the Open Research Fund from Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ), under Grant No. GML-KF-26-19.

## References

[1] 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*, pages 18407–18418, 2024.

[2] Zhenyu Yang, Kairui Zhang, Yuhang Hu, Bing Wang, Shengsheng Qian, Bin Wen, Fan Yang, Tingting Gao, Weiming Dong, and Changsheng Xu. Livestar: Live streaming assistant for real-world online video understanding. *arXiv preprint arXiv:2511.05299*, 2025.

[3] Xueyun Tian, Wei Li, Bingbing Xu, Heng Dong, Yuanzhuo Wang, and Huawei Shen. Roma: Real-time omni-multimodal assistant with interactive streaming understanding. *arXiv preprint arXiv:2601.10323*, 2026.

[4] 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*, 2025.

[5] Yang Liu, Weixing Chen, Yongjie Bai, Xiaodan Liang, Guanbin Li, Wen Gao, and Liang Lin. Aligning cyber space with physical world: A comprehensive survey on embodied ai. *IEEE/ASME Transactions on Mechatronics*, 2025.

[6] Xiangchen Wang, Weiye Zhu, Teng Wang, TianTian Geng, Zekai Zhang, Zhiyuan Qi, Jinyu Yang, and Feng Zheng. Livevln: Breaking the stop-and-go loop in vision-language navigation. *arXiv preprint arXiv:2604.19536*, 2026.

[7] Yiran Shi, Dongqi Guo, Tianchen Zhao, Feng Gao, Liangzhi Shi, Chao Yu, ZhiJian Mo, Qihua Xiao, XiaoShuai Peng, Qingmin Liao, et al. Streamingvla: Streaming vision-language-action model with action flow matching and adaptive early observation. *arXiv preprint arXiv:2603.28565*, 2026.

[8] Meng Wei, Chenyang Wan, Xiqian Yu, Tai Wang, Yuqiang Yang, Xiaohan Mao, Chenming Zhu, Wenzhe Cai, Hanqing Wang, Yilun Chen, et al. Streamvln: Streaming vision-and-language navigation via slowfast context modeling. *arXiv preprint arXiv:2507.05240*, 2025.

[9] Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pages 5971–5984, 2024.

[10] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024.

[11] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

[12] 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, 2025. URL `https://arxiv.org/abs/2502.13923`.

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

[14] OpenGVLab Team. Internvl2: Better than the best—expanding performance boundaries of open-source multimodal models with the progressive scaling strategy, 2024.

[15] Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, et al. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*, 2025.

[16] Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. Internvl3. 5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025.

[17] Kai Huang, Hao Zou, Ye Xi, BoChen Wang, Zhen Xie, and Liang Yu. Ivtp: Instruction-guided visual token pruning for large vision-language models. In *European conference on computer vision*, pages 214–230. Springer, 2024.

[18] Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, et al. Sparsevlm: Visual token sparsification for efficient vision-language model inference. *arXiv preprint arXiv:2410.04417*, 2024.

[19] Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. Visionzip: Longer is better but not necessary in vision language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19792–19802, 2025.

[20] Xiaohu Huang, Hao Zhou, and Kai Han. Prunevid: Visual token pruning for efficient video large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 19959–19973, 2025.

[21] Chenyu Yang, Xuan Dong, Xizhou Zhu, Weijie Su, Jiahao Wang, Hao Tian, Zhe Chen, Wenhai Wang, Lewei Lu, and Jifeng Dai. Pvc: Progressive visual token compression for unified image and video processing in large vision-language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 24939–24949, 2025.

[22] Yizheng Sun, Yanze Xin, Hao Li, Jingyuan Sun, Chenghua Lin, and Riza Theresa Batista-Navarro. Lvpruning: An effective yet simple language-guided vision token pruning approach for multi-modal large language models. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 4299–4308, 2025.

[23] Jiawei Guo, Feifei Zhai, Pu Jian, Qianrun Wei, and Yu Zhou. Crop: Contextual region-oriented visual token pruning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 9767–9783, 2025.

[24] Yilong Chen, Xiang Bai, Zhibin Wang, Chengyu Bai, Yuhan Dai, and Ming Lu. Streamkv: Streaming video question-answering with segment-based kv cache retrieval and compression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 3120–3128, 2026.

[25] Yanlai Yang, Zhuokai Zhao, Satya Narayan Shukla, Aashu Singh, Shlok Kumar Mishra, Lizhu Zhang, and Mengye Ren. Streammem: Query-agnostic kv cache memory for streaming video understanding. *arXiv preprint arXiv:2508.15717*, 2025.

[26] Haowei Zhang, Shudong Yang, Jinlan Fu, See-Kiong Ng, and Xipeng Qiu. Hermes: Kv cache as hierarchical memory for efficient streaming video understanding. *arXiv preprint arXiv:2601.14724*, 2026.

[27] 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*, pages 10807–10816, 2025.

[28] 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.

[29] Sarthak Jain and Byron C Wallace. Attention is not explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, 2019.

[30] Sofia Serrano and Noah A Smith. Is attention interpretable? In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 2931–2951, 2019.

[31] Xiang Liu, Zhenheng Tang, Hong Chen, Peijie Dong, Zeyu Li, Xiuze Zhou, Bo Li, Xuming Hu, and Xiaowen Chu. Can llms maintain fundamental abilities under kv cache compression? *arXiv preprint arXiv:2502.01941*, 2025.

[32] Hongzhi Zhang, Jingyuan Zhang, Xingguang Ji, Qi Wang, and Fuzheng Zhang. Dyntok: Dynamic compression of visual tokens for efficient and effective video understanding. *arXiv preprint arXiv:2506.03990*, 2025.

[33] Xinqi Jin, Hanxun Yu, Bohan Yu, Kebin Liu, Jian Liu, Keda Tao, Yixuan Pei, Huan Wang, Fan Dang, Jiangchuan Liu, et al. Streamingassistant: Efficient visual token pruning for accelerating online video understanding. *arXiv preprint arXiv:2512.12560*, 2025.

[34] 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)*, pages 12147–12151. IEEE, 2026.

[35] Junjie Zhou, Yan Shu, Bo Zhao, Boya Wu, Zhengyang Liang, Shitao Xiao, Minghao Qin, Xi Yang, Yongping Xiong, Bo Zhang, et al. Mlvu: Benchmarking multi-task long video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13691–13701, 2025.

[36] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

[37] Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai C Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. *Advances in Neural Information Processing Systems*, 37:87310–87356, 2024.

[38] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35: 23716–23736, 2022.

[39] Junke Wang, Lingchen Meng, Zejia Weng, Bo He, Zuxuan Wu, and Yu-Gang Jiang. To see is to believe: Prompting gpt-4v for better visual instruction tuning. *arXiv preprint arXiv:2311.07574*, 2023.

[40] Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, et al. Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 91–104, 2025.

[41] 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.

[42] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[43] Jiabo Ye, Haiyang Xu, Haowei Liu, Anwen Hu, Ming Yan, Qi Qian, Ji Zhang, Fei Huang, and Jingren Zhou. mplug-owl3: Towards long image-sequence understanding in multi-modal large language models. *arXiv preprint arXiv:2408.04840*, 2024.

[44] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Khan. Video-chatgpt: Towards detailed video understanding via large vision and language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12585–12602, 2024.

[45] Boqiang Zhang, Kehan Li, Zesen Cheng, Zhiqiang Hu, Yuqian Yuan, Guanzheng Chen, Sicong Leng, Yuming Jiang, Hang Zhang, Xin Li, et al. Videollama 3: Frontier multimodal foundation models for image and video understanding. *arXiv preprint arXiv:2501.13106*, 2025.

[46] Jiajun Liu, Yibing Wang, Hanghang Ma, Xiaoping Wu, Xiaoqi Ma, Xiaoming Wei, Jianbin Jiao, Enhua Wu, and Jie Hu. Kangaroo: A powerful video-language model supporting long-context video input: J. liu et al. *International Journal of Computer Vision*, 134(3):114, 2026.

[47] Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, et al. Mvbench: A comprehensive multi-modal video understanding benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22195–22206, 2024.

[48] Peiyuan Zhang, Kaichen Zhang, Bo Li, Guangtao Zeng, Jingkang Yang, Yuanhan Zhang, Ziyue Wang, Haoran Tan, Chunyuan Li, and Ziwei Liu. Long context transfer from language to vision. *arXiv preprint arXiv:2406.16852*, 2024.

[49] Yukang Chen, Fuzhao Xue, Dacheng Li, Qinghao Hu, Ligeng Zhu, Xiuyu Li, Yunhao Fang, Haotian Tang, Shang Yang, Zhijian Liu, et al. Longvila: Scaling long-context visual language models for long videos. In *The Thirteenth International Conference on Learning Representations*, 2024.

[50] Xiaoqian Shen, Yunyang Xiong, Changsheng Zhao, Lemeng Wu, Jun Chen, Chenchen Zhu, Zechun Liu, Fanyi Xiao, Balakrishnan Varadarajan, Florian Bordes, et al. Longvu: Spatiotemporal adaptive compression for long video-language understanding. *arXiv preprint arXiv:2410.17434*, 2024.

[51] Yan Shu, Zheng Liu, Peitian Zhang, Minghao Qin, Junjie Zhou, Zhengyang Liang, Tiejun Huang, and Bo Zhao. Video-xl: Extra-long vision language model for hour-scale video understanding. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 26160–26169, 2025.

[52] Minghao Qin, Xiangrui Liu, Zhengyang Liang, Yan Shu, Huaying Yuan, Juenjie Zhou, Shitao Xiao, Bo Zhao, and Zheng Liu. Video-xl-2: Towards very long-video understanding through task-aware kv sparsification. *arXiv preprint arXiv:2506.19225*, 2025.

[53] Xi Tang, Jihao Qiu, Lingxi Xie, Yunjie Tian, Jianbin Jiao, and Qixiang Ye. Adaptive keyframe sampling for long video understanding. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 29118–29128, 2025.

[54] Weiming Ren, Huan Yang, Jie Min, Cong Wei, and Wenhu Chen. Vista: Enhancing long-duration and high-resolution video understanding by video spatiotemporal augmentation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 3804–3814, 2025.

[55] Haoji Zhang, Yiqin Wang, Yansong Tang, Yong Liu, Jiashi Feng, Jifeng Dai, and Xiaojie Jin. Flash-vstream: Memory-based real-time understanding for long video streams. *arXiv preprint arXiv:2406.08085*, 2024.

[56] 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*, pages 24045–24055, 2025.

[57] Shenghao Fu, Qize Yang, Yuan-Ming Li, Yi-Xing Peng, Kun-Yu Lin, Xihan Wei, Jian-Fang Hu, Xiaohua Xie, and Wei-Shi Zheng. Vispeak: Visual instruction feedback in streaming videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 21778–21788, 2025.

[58] Kecheng Zhang, Zongxin Yang, Mingfei Han, Haihong Hao, Yunzhi Zhuge, Changlin Li, Junhan Zhao, Zhihui Li, and Xiaojun Chang. Progressive online video understanding with evidence-aligned timing and transparent decisions. *arXiv preprint arXiv:2604.18459*, 2026.

[59] Shangzhe Di, Zhelun Yu, Guanghao Zhang, Haoyuan Li, Tao Zhong, Hao Cheng, Bolin Li, Wanggui He, Fangxun Shu, and Hao Jiang. Streaming video question-answering with in-context video kv-cache retrieval. *arXiv preprint arXiv:2503.00540*, 2025.

[60] Zhenyu Ning, Guangda Liu, Qihao Jin, Wenchao Ding, Minyi Guo, and Jieru Zhao. Livevlm: Efficient online video understanding via streaming-oriented kv cache and retrieval. *arXiv preprint arXiv:2505.15269*, 2025.

[61] 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*, pages 18902–18913, 2025.

[62] Karttikeya Mangalam, Raiymbek Akshulakov, and Jitendra Malik. Egoschema: A diagnostic benchmark for very long-form video language understanding. *Advances in Neural Information Processing Systems*, 36:46212–46244, 2023.

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

[64] Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, et al. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 24108–24118, 2025.

[65] Enxin Song, Wenhao Chai, Guanhong Wang, Yucheng Zhang, Haoyang Zhou, Feiyang Wu, Haozhe Chi, Xun Guo, Tian Ye, Yanting Zhang, et al. Moviechat: From dense token to sparse memory for long video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18221–18232, 2024.

[66] Zhenpeng Huang, Xinhao Li, Jiaqi Li, Jing Wang, Xiangyu Zeng, Cheng Liang, Tao Wu, Xi Chen, Liang Li, and Limin Wang. Online video understanding: OVBench and VideoChat-Online. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. URL `https://videochat-online.github.io/`.

# Appendix

## A Subtasks of online video benchmarks

We provide detailed descriptions of the online video understanding subtasks used in Table 1 of the main paper. Our online evaluation is conducted on the real-time subsets of OVO-Bench and StreamingBench. Both benchmarks require the model to answer questions under a causal streaming protocol, where only the video prefix observed before the query timestamp can be used. This setting is consistent with the online inference scenario considered in LifeStream.

**OVO-Bench real-time subtasks.** OVO-Bench evaluates online video understanding under three temporal modes: Backward Tracing, Real-Time Visual Perception, and Forward Active Responding. In the main paper, we focus on the Real-Time Visual Perception split, which contains six subtasks. The abbreviations are summarized in Table A.1.

Table A.1: Descriptions of the OVO-Bench real-time visual perception subtasks.

| Abbr. | Subtask | Description |
|---|---|---|
| OCR | Optical Character Recognition | Recognize and interpret characters that appear within the current frame. |
| ACR | Action Recognition | Recognize and interpret the actions performed by individuals in the current frame. |
| ATR | Attribute Recognition | Identify object attributes or properties, such as color, texture, and size, in nearby frames. |
| STU | Spatial Understanding | Reason over spatial relationships between objects occurring in nearby frames. |
| FPD | Future Prediction | Forecast the most probable subsequent phase of the current scene, including changes in object states, actions, or other dynamic elements. |
| OJR | Object Recognition | Recognize objects appearing in the current frames. |

**StreamingBench real-time subtasks.** StreamingBench groups streaming video understanding into three categories: Real-Time Visual Understanding, Omni-Source Understanding, and Contextual Understanding. In the main paper, we report the Real-Time Visual Understanding results, which include ten subtasks. The abbreviations are summarized in Table A.2.

Table A.2: Descriptions of the StreamingBench real-time visual understanding subtasks.

| Abbr. | Subtask | Description |
|---|---|---|
| OP | Object Perception | Detect and identify specific objects within the video. |
| CR | Causal Reasoning | Analyze cause-and-effect relationships among events in the video stream. |
| CS | Clips Summarization | Summarize the main content of specific video clips. |
| ATP | Attribute Perception | Identify and categorize attributes of objects or individuals. |
| EU | Event Understanding | Recognize and describe sequences of events in the video stream. |
| TR | Text-Rich Understanding | Interpret and explain text-rich content within the video. |
| PR | Prospective Reasoning | Predict future events based on the current video context. |
| SU | Spatial Understanding | Understand and describe spatial relationships and locations. |
| ACP | Action Perception | Identify and describe specific actions in the video. |
| CT | Counting | Count occurrences of specific objects or actions. |

## B Lifecycle guided global consolidation

**Notation.** For a sampled frame $I_t$, let

$$X_t = \{x_t^i\}_{i=1}^{N_t} \tag{B.1}$$

denote its visual tokens encoded by the vision encoder, where $N_t$ is the number of tokens in frame $I_t$. We denote the three memory banks by $\mathcal{M}_R$, $\mathcal{M}_A$, and $\mathcal{M}_G$, corresponding to recent memory, archive memory, and global memory, respectively. The recent and archive memories have frame capacities $N_R$ and $N_A$, while $B$ denotes the maximum number of visual tokens submitted to the LLM.

The lifecycle state space is

$$\mathcal{S} = \{\texttt{onset}, \texttt{stable}, \texttt{offset}, \texttt{fleeting}\}, \qquad \mathcal{S}_{ns} = \mathcal{S} \setminus \{\texttt{stable}\}. \tag{B.2}$$

Here, $\mathcal{S}_{ns}$ denotes the non-stable state set. For frame $I_t$, $\mathcal{A}_t$ denotes its archived representation after discarding stable tokens, and $\mathcal{G}_t$ denotes its consolidated representation written into the global memory. The default memory construction uses only the discrete lifecycle state $z_t^i$; the confidence score $\rho_t^i$ produced by lifecycle assignment is not required by the global consolidation algorithm.

For a visual token grid of size $H_t \times W_t$, we use $q(i) = (u_i, v_i)$ to denote the spatial coordinate of the original token index $i$. The local $3 \times 3$ neighborhood of token $i$ is defined as

$$\mathcal{N}_{3\times3}(i) = \{j \mid \|q(j) - q(i)\|_\infty \le 1\}. \tag{B.3}$$

For an index set $\mathcal{I}$, we use $\mathrm{CC}(\mathcal{I}, \mathcal{N}_{3\times3})$ to denote the connected-component partitioning of token indices in $\mathcal{I}$ under the above $3 \times 3$ spatial adjacency relation.

**Frame-level global consolidation.** After archival compression, each archived frame contains only non-stable visual tokens. When an archived frame is promoted into the global memory, LifeStream further consolidates its boundary-related tokens while preserving highly informative transient evidence. Importantly, the archive representation keeps the original spatial token index, which is used for local grouping and order restoration. Formally, let

$$\mathcal{I}_t^{ns} = \{i \in \{1, \dots, N_t\} \mid z_t^i \in \mathcal{S}_{ns}\}, \qquad N_t^a = |\mathcal{I}_t^{ns}|, \tag{B.4}$$

and define the archived representation as

$$\mathcal{A}_t = \{(x_t^i, z_t^i, i) \mid i \in \mathcal{I}_t^{ns}\}. \tag{B.5}$$

For each state $r \in \mathcal{S}_{ns}$, we define

$$\mathcal{I}_t^r = \{i \mid (x_t^i, z_t^i, i) \in \mathcal{A}_t,\ z_t^i = r\}. \tag{B.6}$$

If $\mathcal{A}_t = \emptyset$, the frame contains no non-stable tokens and is skipped during global consolidation. Otherwise, the non-stable state distribution is computed as

$$\pi_t^r = \frac{|\mathcal{I}_t^r|}{\sum_{r' \in \mathcal{S}_{ns}} |\mathcal{I}_t^{r'}|}, \qquad r \in \mathcal{S}_{ns}. \tag{B.7}$$

`fleeting` tokens are directly preserved, since they usually correspond to transient fine-grained evidence. For boundary states, we identify the dominant boundary state

$$b_t = \arg \max_{r \in \{\texttt{onset}, \texttt{offset}\}} \pi_t^r \tag{B.8}$$

and its dominance score

$$\eta_t = \max(\pi_t^{\texttt{onset}}, \pi_t^{\texttt{offset}}). \tag{B.9}$$

Boundary consolidation is performed only when

$$\eta_t > \gamma, \tag{B.10}$$

where $\gamma$ is the boundary dominance threshold. This conservative rule avoids over-compressing temporally ambiguous frames where onset and offset evidence are mixed.

**Algorithm B.1** Lifecycle-guided global consolidation

**Require:** Archived frame representation $\mathcal{A}_t = \{(x_t^i, z_t^i, i) \mid i \in \mathcal{I}_t^{ns}\}$; spatial coordinate mapping $q(\cdot)$; boundary dominance threshold $\gamma$.
**Ensure:** Global representation $\mathcal{G}_t$ of frame $I_t$.

**if** $\mathcal{A}_t = \emptyset$ **then**
  **return** $\emptyset$
**end if**
Define $\mathcal{S}_{ns} = \{\texttt{fleeting}, \texttt{onset}, \texttt{offset}\}$.
**for** each $r \in \mathcal{S}_{ns}$ **do**
  $\mathcal{I}_t^r \leftarrow \{i \mid (x_t^i, z_t^i, i) \in \mathcal{A}_t,\ z_t^i = r\}$.
  $\pi_t^r \leftarrow |\mathcal{I}_t^r| / \sum_{r' \in \mathcal{S}_{ns}} |\mathcal{I}_t^{r'}|$.
**end for**
Initialize temporary list $\mathcal{Y}_t \leftarrow [\,]$.
Append all $(x_t^i, i)$ with $i \in \mathcal{I}_t^{\texttt{fleeting}}$ to $\mathcal{Y}_t$.
$b_t \leftarrow \arg\max_{r \in \{\texttt{onset}, \texttt{offset}\}} \pi_t^r$.
$\eta_t \leftarrow \max(\pi_t^{\texttt{onset}}, \pi_t^{\texttt{offset}})$.
**if** $\eta_t > \gamma$ **then**
  $\{G_{t,m}\}_{m=1}^{M_t} \leftarrow \mathrm{CC}(\mathcal{I}_t^{b_t}, \mathcal{N}_{3\times3})$.
  **for** each component $G_{t,m}$ **do**
    $\bar{x}_{t,m} \leftarrow |G_{t,m}|^{-1} \sum_{i \in G_{t,m}} x_t^i$.
    $a_{t,m} \leftarrow \min_{i \in G_{t,m}} i$.
    Append $(\bar{x}_{t,m}, a_{t,m})$ to $\mathcal{Y}_t$.
  **end for**
  Let $b_t'$ be the other state in $\{\texttt{onset}, \texttt{offset}\} \setminus \{b_t\}$.
  Append all $(x_t^i, i)$ with $i \in \mathcal{I}_t^{b_t'}$ to $\mathcal{Y}_t$.
**else**
  Append all $(x_t^i, i)$ with $i \in \mathcal{I}_t^{\texttt{onset}} \cup \mathcal{I}_t^{\texttt{offset}}$ to $\mathcal{Y}_t$.
**end if**
Sort $\mathcal{Y}_t$ by the spatial anchor index in ascending order.
Remove anchor indices from $\mathcal{Y}_t$ to obtain the ordered token sequence $\mathcal{G}_t$.
**return** $\mathcal{G}_t$.

Given the dominant boundary state $b_t$, we compute connected components over $\mathcal{I}_t^{b_t}$:

$$\{G_{t,m}\}_{m=1}^{M_t} = \mathrm{CC}(\mathcal{I}_t^{b_t}, \mathcal{N}_{3\times3}). \tag{B.11}$$

Each connected component is summarized into one representative token by average pooling:

$$\bar{x}_{t,m} = \frac{1}{|G_{t,m}|} \sum_{i \in G_{t,m}} x_t^i, \qquad m = 1, \ldots, M_t. \tag{B.12}$$

For each pooled component, we use the smallest original token index as its spatial anchor:

$$a_{t,m} = \min_{i \in G_{t,m}} i. \tag{B.13}$$

The anchor index is used only to restore the flattened spatial order after pooling.

The final global representation of frame $I_t$ consists of all `fleeting` tokens, the pooled representatives of the dominant boundary state, and the non-dominant boundary tokens kept unchanged. Non-dominant boundary tokens are preserved because they are usually sparse but may correspond to local counter-transitions or small moving objects. This conservative design avoids losing rare boundary evidence while still compressing transition-heavy regions. If no boundary state satisfies the dominance condition, the archived tokens of this frame are written into the global memory without boundary pooling.

## C Full lifecycle guided memory algorithm

**End-to-end streaming procedure.** Algorithm C.1 summarizes the complete streaming pipeline of LifeStream. The method maintains three memory banks: recent memory $\mathcal{M}_R$, archive memory $\mathcal{M}_A$, and global memory $\mathcal{M}_G$. The recent memory stores full visual tokens for the latest frames, the archive memory stores non-stable tokens after lifecycle state assignment, and the global memory stores compact long-range representations after lifecycle-guided global consolidation.

To preserve online causality without storing full historical tokens, LifeStream computes and caches the past correspondence between adjacent frames when the newer frame arrives. When a frame leaves the recent memory, its cached past correspondence is combined with future correspondences computed from frames still in the recent memory. At query time, LifeStream directly concatenates the pre-constructed memories in temporal order and applies a query-agnostic token budget cap, without additional query-dependent retrieval.

**Local temporal correspondence.** For compact notation, we use $\textsc{LocalCorr}(X_t, X_{t'})$ to denote the token-level local correspondence from reference frame $I_{t'}$ to target frame $I_t$. It returns

$$\textsc{LocalCorr}(X_t, X_{t'}) = \mathbf{m}_{t'\to t} = \{m^i_{t'\to t}\}_{i=1}^{N_t}, \tag{C.1}$$

where

$$m^i_{t'\to t} = \max_{j\in\mathcal{N}_{3\times 3}(i)} \cos(x^i_t, x^j_{t'}). \tag{C.2}$$

LIFECYCLESTATEASSIGN follows the lifecycle scoring rule in the main paper. Given past correspondence and future correspondences, it computes normalized past and future descriptors and outputs lifecycle states $\{z^i_t\}_{i=1}^{N_t}$. For configurations with $\delta_f \geq 1$, the future descriptor uses weights $\{\alpha_k\}_{k=1}^{\delta_f}$ satisfying

$$\alpha_k \geq 0, \qquad \sum_{k=1}^{\delta_f} \alpha_k = 1. \tag{C.3}$$

The $\delta_f = 0$ setting corresponds to the past-only ablation in Table 5 and does not use future-reference weights. In the default setting, $\delta_f = 2$, $\alpha_1 = 0.7$, and $\alpha_2 = 0.3$. GLOBALCONSOLIDATE follows Algorithm B.1. CONCAT denotes temporal concatenation of the visual tokens stored in $\mathcal{M}_G$, $\mathcal{M}_A$, and $\mathcal{M}_R$, where auxiliary metadata such as lifecycle states and spatial indices is removed before feeding tokens into the VLLM. QUERYAGNOSTICBUDGETCAP denotes deterministic budget control under token budget $B$, without computing token-query relevance.

**Boundary cases.** For the first frame, the past correspondence is unavailable and is omitted in lifecycle state assignment. For the latest frames that do not yet have sufficient future observations, LifeStream keeps them in the recent memory as full visual tokens. Therefore, the algorithm never accesses frames beyond the observed video prefix. When a video stream ends and the remaining recent frames need to be flushed, we use the available temporal neighbors and renormalize the corresponding future weights.

**Algorithm C.1** Full lifecycle guided memory algorithm

**Require:** Video stream $\{I_\tau\}_{\tau=1}^{T}$; vision encoder $E_v$; recent capacity $N_R$; archive capacity $N_A$; future window $\delta_f$ with $N_R \geq \delta_f$; future weights $\{\alpha_k\}_{k=1}^{\delta_f}$; dominance threshold $\gamma$; token budget $B$.

**Ensure:** Lifecycle-guided visual memory for online video understanding.

Initialize recent memory $\mathcal{M}_R \leftarrow \emptyset$, archive memory $\mathcal{M}_A \leftarrow \emptyset$, global memory $\mathcal{M}_G \leftarrow \emptyset$.
Initialize past correspondence cache $\mathcal{C}_P \leftarrow \emptyset$.
**for** each incoming frame $I_\tau$ **do**
  Encode visual tokens $X_\tau = \{x_\tau^i\}_{i=1}^{N_\tau} \leftarrow E_v(I_\tau)$.
  **if** $\tau > 1$ **then**
    $\mathcal{C}_P[\tau] \leftarrow \textsc{LocalCorr}(X_\tau, X_{\tau-1})$.
  **end if**
  Append $(\tau, X_\tau)$ to $\mathcal{M}_R$.
  **if** $|\mathcal{M}_R| > N_R$ **then**
    Let $(t, X_t)$ be the oldest frame-token pair in $\mathcal{M}_R$.
    Retrieve cached past correspondence $\mathbf{m}_{t-1\to t} \leftarrow \mathcal{C}_P[t]$, if it exists.
    Compute future correspondences $\{\mathbf{m}_{t+k\to t}\}_{k=1}^{\delta_f} \leftarrow \{\textsc{LocalCorr}(X_t, X_{t+k})\}_{k=1}^{\delta_f}$.
    Assign lifecycle states

$$\{z_t^i\}_{i=1}^{N_t} \leftarrow \textsc{LifecycleStateAssign}(\mathbf{m}_{t-1\to t}, \{\mathbf{m}_{t+k\to t}\}_{k=1}^{\delta_f}, \{\alpha_k\}_{k=1}^{\delta_f}).$$

    Form archived representation:

$$\mathcal{A}_t \leftarrow \{(x_t^i, z_t^i, i) \mid z_t^i \neq \texttt{stable}\}.$$

    **if** $\mathcal{A}_t \neq \emptyset$ **then**
      Append $(t, \mathcal{A}_t)$ to $\mathcal{M}_A$.
    **end if**
    Remove $(t, X_t)$ from $\mathcal{M}_R$, and remove $\mathcal{C}_P[t]$ if it exists.
  **end if**
  **if** $|\mathcal{M}_A| > N_A$ **then**
    Let $(s, \mathcal{A}_s)$ be the oldest archived frame in $\mathcal{M}_A$.
    $\mathcal{G}_s \leftarrow \textsc{GlobalConsolidate}(\mathcal{A}_s, \gamma)$.
    Append $(s, \mathcal{G}_s)$ to $\mathcal{M}_G$.
    Remove $(s, \mathcal{A}_s)$ from $\mathcal{M}_A$.
  **end if**
  **if** a user query $Q_\tau$ arrives **then**
    $\mathcal{V}_\tau \leftarrow \textsc{Concat}(\mathcal{M}_G, \mathcal{M}_A, \mathcal{M}_R)$.
    $\widetilde{\mathcal{V}}_\tau \leftarrow \textsc{QueryAgnosticBudgetCap}(\mathcal{V}_\tau, B)$.
    Feed $(\widetilde{\mathcal{V}}_\tau, Q_\tau)$ into the VLLM.
  **end if**
**end for**

**return** $(\mathcal{M}_G, \mathcal{M}_A, \mathcal{M}_R)$.

# D Causal memory updates and query inference

Under the chunked input protocol in Section 4.1, memory construction and query inference use only frames at or before the query timestamp, including when the timestamp falls within a 32-frame chunk.

We measure per-frame memory maintenance at 196 tokens/frame on one NVIDIA A100, varying $\delta_f$ while keeping the other settings fixed.

Table D: **Memory-update cost with different historical-frame reference windows.**

| $\delta_f$ | Wait for new observations (ms) | Memory update (ms/frame) | Update throughput (frames/s) |
|---|---|---|---|
| 0 | 0 | 2.7 | 370.4 |
| 1 | 0 | 3.3 | 303.0 |
| 2 | 0 | 3.9 | 256.4 |

Table D shows that the update cost increases from 2.7 to 3.9 ms/frame as $\delta_f$ increases from 0 to 2, without requiring queries to wait for additional video observations. These measurements exclude visual encoding and QA inference, whereas Table 5 reports the entire QA process. The update throughput therefore characterizes memory maintenance rather than the answer-generation rate of the full VLLM.

# E Additional comparisons

We average the main paper's accuracy gains equally across the six benchmarks. For VideoMME, we first average the gains on the reported medium and long subsets.

## E.1 Comparisons at matched frame rates

To examine whether LifeStream benefits primarily from observing more frames, we compare methods at 0.5 and 1 fps using LLaVA-OneVision-7B. At 0.5 fps, we use the baseline results reported in Table 1 and additionally evaluate LifeStream at the same sampling rate. The reported HERMES results correspond to its LLaVA-OneVision-7B configuration with a 4K-token budget. Missing baseline results remain unavailable. At 1 fps, ReKV, StreamKV, LiveVLM, and HERMES are re-evaluated with identical benchmark splits, causal prefixes, query timestamps, prompts, decoding settings, and answer parsing. Each method retains its released memory configuration while the external stream is standardized to 1 fps. Thus, the first comparison aligns sampling rates with reported results, while the second additionally standardizes the evaluation protocol for the re-evaluated methods.

As shown in Table E.1, LifeStream exceeds HERMES by 1.2/1.5 percentage points on OVO-RT/SB-RT at 0.5 fps and by 2.1/1.9 points at 1 fps. These comparisons support a benefit from lifecycle-guided memory beyond the difference in sampling rate. We retain 1 fps as the main LifeStream setting; the 0.5-fps result is a supplementary evaluation.

Table E.1: **Frame-rate comparisons with LLaVA-OneVision-7B.** We report accuracy (%) on the real-time subsets of OVO-Bench and StreamingBench. † indicates re-evaluation with the released memory configuration and a standardized 1-fps stream; – denotes an unavailable result.

| Method | Frames | OVO-Bench | StreamingBench |
|---|---|---|---|
| | | Real-time | Real-time |
| **Reported baselines at 0.5 fps** | | | |
| ReKV | 0.5 fps | – | 69.1 |
| StreamKV | 0.5 fps | – | 71.0 |
| LiveVLM | 0.5 fps | – | 72.9 |
| HERMES | 0.5 fps | 66.3 | 73.2 |
| **LifeStream** $_{\text{+LLaVA-OneVision-7B}}$ | **0.5 fps** | **67.5** | **74.7** |
| **Re-evaluated baselines at 1 fps** | | | |
| ReKV† | 1 fps | 61.2 | 70.3 |
| StreamKV† | 1 fps | 64.7 | 71.2 |
| LiveVLM† | 1 fps | 65.2 | 72.7 |
| HERMES† | 1 fps | 66.8 | 73.6 |
| **LifeStream** $_{\text{+LLaVA-OneVision-7B}}$ | **1 fps** | **68.9** | **75.5** |

### E.2 Additional streaming baselines

We include VideoChat-Online [66] and StreamForest [28] to broaden the comparison with streaming-specific memory systems. VideoChat-Online-4B is re-evaluated from its official checkpoint under our standardized 1-fps protocol, preserving the released Pyramid Memory Bank. For StreamForest, we use its officially reported 1-fps results. Its OVO-RT score is the real-time visual perception aggregate, rather than the aggregate over all OVO-Bench task families.

Table E.2: **Additional comparisons with streaming video models at 1 fps.** We report accuracy (%) on the real-time subsets of OVO-Bench and StreamingBench. † indicates re-evaluation from the released checkpoint; ‡ indicates officially reported results.

| Method | Streaming-specific training | OVO-Bench | StreamingBench |
|---|---|---|---|
| | | Real-time | Real-time |
| VideoChat-Online-4B† [66] | Yes | 60.4 | 72.5 |
| StreamForest-7B‡ [28] | Yes | 61.2 | 77.3 |
| **LifeStream** $_{\text{+LLaVA-OneVision-7B}}$ | **No** | **68.9** | **75.5** |
| **LifeStream** $_{\text{+Qwen3-VL-8B}}$ | **No** | **71.1** | **79.4** |

Table E.2 places LifeStream alongside methods with different backbones, parameter counts, and streaming-specific training. LifeStream with LLaVA-OneVision-7B attains 68.9/75.5 on OVO-RT/SB-RT, while its Qwen3-VL-8B variant attains 71.1/79.4. These results complement the controlled same-backbone comparisons; they do not isolate the effect of the memory mechanism from backbone or training differences.

## F Additional ablations and robustness

Unless otherwise stated, this section follows the LLaVA-OneVision-7B ablation setup in Section 4.4.

### F.1 Separating temporal evidence from lifecycle-guided compression

We separate three sources of improvement: additional temporal evidence, an explicit four-state lifecycle representation, and state-conditioned consolidation. The past-only binary variant uses

no future-relative evidence and no four-state representation. The bidirectional binary variant adds future-relative evidence while retaining the binary representation. Lifecycle + uniform introduces the four lifecycle states but uses uniform consolidation, and full LifeStream conditions consolidation on lifecycle state. Global tokens/frame is measured after consolidation and before query-time budget control. This intervention differs from the state-partition ablation in Table 3, which compares retained lifecycle categories.

Table F.1: **Separating future evidence, lifecycle modeling, and state-conditioned consolidation.**

| Variant | Future evidence | Four states | State-guided consolidation | StreamingBench | MLVU | Global |
|---|---|---|---|---|---|---|
| | | | | Real-time | M-Avg | tokens/frame |
| Past-only binary | ✗ | ✗ | ✗ | 72.5 | 64.2 | 11.9 |
| Bidirectional binary | ✓ | ✗ | ✗ | 73.1 | 64.9 | 11.7 |
| Lifecycle + uniform | ✓ | ✓ | ✗ | 74.4 | 66.5 | 11.6 |
| **Full LifeStream** | ✓ | ✓ | ✓ | **75.5** | **67.8** | **11.4** |

Adding future-relative evidence to the binary variant improves SB-RT/MLVU by 0.6/0.7 percentage points. Introducing the four-state representation then adds 1.3/1.6 points, and state-conditioned consolidation adds a further 1.1/1.3 points. Meanwhile, the mean Global representation decreases from 11.9 to 11.4 tokens/frame across the four variants. The progression in Table F.1 indicates that the improvement is not explained solely by using additional temporal references: lifecycle modeling and its use in memory consolidation both contribute under these controlled settings.

### F.2 Robustness to scene and camera motion

For each valid adjacent frame pair, let $u_t(p)$ denote dense optical flow at pixel $p$ and let $H, W$ be the image height and width. Define

$$m_t = P_{90}\left(\left\{\frac{\|u_t(p)\|_2}{\sqrt{H^2+W^2}}\right\}_p\right), \qquad M(V) = P_{75}(\{m_t\}_{t\in V}), \tag{F.1}$$

where $P_q$ is the $q$-th percentile. Hard cuts and frame pairs with insufficient valid flow are excluded. The bottom and top 30% of videos form the low- and high-motion subsets, respectively, retaining all questions from each selected video. Each SB-RT subset contains 150 videos and 750 questions. The MLVU low- and high-motion subsets each contain 336 videos, with 648 and 629 questions, respectively.

Within the high-motion subset, we fit a RANSAC homography to dense-flow correspondences using a fixed resolution-normalized reprojection threshold. Camera explainability for an adjacent pair is $e_t = N_t^{\text{in}}/N_t^{\text{valid}}$, where the numerator counts homography inliers and the denominator counts valid correspondences. The video score is $C(V) = P_{75}(\{e_t\}_{t\in V})$. Selecting the top 50% yields 75 SB-RT and 168 MLVU camera-dominant videos. These subset definitions are independent of LifeStream and are used only for evaluation, not during inference. All results below use the original fixed configuration, including $\delta_f = 2$.

The base model is LLaVA-OneVision-7B. Onset+offset is measured before Global consolidation, while Global tokens/frame is measured afterward, before query-time budget control; – denotes an unreported measurement.

Table F.2 shows consistent gains over the base model in both low- and high-motion subsets. On SB-RT, the gains are 4.2 and 4.5 percentage points, respectively; on MLVU they are 2.9 and 3.2 points. On camera-dominant videos, LifeStream improves SB-RT/MLVU by 4.5/3.1 points. Relative to the high-motion subsets, the onset+offset fraction increases by 1.5/1.7 percentage points and the Global representation increases by 0.5/0.5 tokens/frame. Thus, camera-dominant motion makes retention moderately more conservative in these evaluations while preserving a performance advantage.

The 8,192-token cap applies to the visual context submitted to the LLM. It should not be interpreted as a bound on all resident memory or total GPU usage; the retained-token measurements above describe the evaluated Global representations before that query-time cap.

Table F.2: **Robustness of fixed-parameter LifeStream across motion regimes.**

| Benchmark | Subset | Accuracy (%) | | | Onset+offset | Global |
|---|---|---|---|---|---|---|
| | | Base | LifeStream | Δ (pp) | (%) | tokens/frame |
| StreamingBench Real-time | Full | 71.1 | 75.5 | +4.4 | 24.7 | 11.4 |
| | Low motion | 72.6 | 76.8 | +4.2 | – | – |
| | High motion | 69.2 | 73.7 | +4.5 | 28.6 | 12.3 |
| | Camera-dominant | 68.5 | 73.0 | +4.5 | 30.1 | 12.8 |
| MLVU M-Avg | Full | 64.7 | 67.8 | +3.1 | 32.9 | 11.4 |
| | Low motion | 66.1 | 69.0 | +2.9 | – | – |
| | High motion | 62.8 | 66.0 | +3.2 | 36.4 | 12.5 |
| | Camera-dominant | 62.3 | 65.4 | +3.1 | 38.1 | 13.0 |

# G Exploratory adaptive variants

The fixed-parameter LifeStream configuration remains the main method throughout the paper. Here we examine three optional adaptations separately, using LLaVA-OneVision-7B and the same evaluation settings. They are not combined, and their results do not replace the main-paper results. Global tokens/frame is measured after consolidation and before query-time budget control; – denotes an unreported measurement or an inapplicable difference.

## G.1 Correspondence-aware lifecycle weights

For token $i$ and future reference $k \in \{1, 2\}$, let $q_{i,k}$ be the margin between the top-one and top-two local similarity scores, and let $\bar{q}_{t,k} = N^{-1} \sum_i q_{i,k}$. With prior $(\bar{\alpha}_1, \bar{\alpha}_2) = (0.7, 0.3)$, we set

$$\alpha_{t,k} = \frac{\bar{\alpha}_k(\bar{q}_{t,k} + \epsilon)}{\sum_{j=1}^{2} \bar{\alpha}_j(\bar{q}_{t,j} + \epsilon)}, \tag{G.1}$$

where $\epsilon > 0$ stabilizes the normalization. This retains temporal proximity as the prior while giving more weight to the reference with clearer local correspondence.

## G.2 Uncertainty-aware consolidation threshold

Normalize the lifecycle scores from Algorithm 1 as $\widetilde{p}_i^r = s_r^i / (\sum_{r' \in \mathcal{S}} s_{r'}^i + \epsilon)$, and define $c_i = \widetilde{p}_i^{(1)} - \widetilde{p}_i^{(2)}$, the gap between the largest and second-largest normalized scores. For a candidate dominant boundary region $D_t$, let $\bar{c}_t = |D_t|^{-1} \sum_{i \in D_t} c_i$ and set

$$\gamma_t = \gamma_0 + (1 - \bar{c}_t)(1 - \gamma_0), \qquad \gamma_0 = 0.7. \tag{G.2}$$

We retain the dominance measure from the main method, $\eta_t = \max(\pi_t^{\text{onset}}, \pi_t^{\text{offset}})$, where each state proportion is normalized over all non-stable tokens. Adaptive consolidation applies when $\eta_t > \gamma_t$. Lower confidence therefore requires stronger boundary-state dominance before pooling.

Table G.1: **Exploratory calibration of lifecycle weights and the consolidation threshold.**

| Configuration | StreamingBench | | MLVU | | Global |
|---|---|---|---|---|---|
| | Real-time | Δ (pp) | M-Avg | Δ (pp) | tokens/frame |
| **Fixed defaults** | 75.5 | – | 67.8 | – | **11.4** |
| Adaptive $\alpha$ | 75.3 | -0.2 | **68.1** | +0.3 | – |
| Adaptive $\gamma_t$ | **75.6** | +0.1 | 67.9 | +0.1 | 12.3 |

Table G.1 shows that adaptive weights improve MLVU by 0.3 points but reduce SB-RT by 0.2 points. The adaptive threshold improves both scores by 0.1 points while increasing Global tokens/frame from 11.4 to 12.3. These small changes illustrate a distribution-dependent benefit and an accuracy–memory trade-off; they do not establish a universal advantage over the fixed defaults.

### G.3 Motion-adaptive reference windows

A fixed frame count need not correspond to a fixed amount of visual change. Fast motion can cause correspondence drift over a long span, while slow motion can require more frames to accumulate useful evidence. We therefore explore choosing the shortest span with sufficient accumulated motion. For adjacent frames, define

$$d_j(p) = \frac{\|u_j(p)\|_2}{\sqrt{H^2 + W^2}}, \quad a_j = P_{90}(\{d_j(p)\}_p), \quad A_t(k) = \sum_{r=0}^{k-1} a_{t+r}, \tag{G.3}$$

using valid pixels. We select the smallest $k \in \{1, 2, 3\}$ such that $A_t(k) \geq \tau_m$, and otherwise use 3. The same dimensionless threshold $\tau_m = 0.02$ is used for both benchmarks. Fast motion tends to select a shorter span, whereas slow motion selects a longer one. All selected references are already in the observed causal prefix before $I_t$ is compressed; the rule does not wait for additional frames to answer a query.

We compare the adaptive rule with the default $\delta_f = 2$ configuration; the complete fixed-window sweep is reported in Table 5.

Table G.2: **Exploratory motion-adaptive reference windows.**

| Future window | StreamingBench | | MLVU | |
|---|---|---|---|---|
| | Real-time | Δ (pp) | M-Avg | Δ (pp) |
| $\boldsymbol{\delta_f = 2}$ | 75.5 | +0.0 | 67.8 | +0.0 |
| Adaptive $\delta_f(t)$ | **75.8** | +0.3 | **68.2** | +0.4 |

The adaptive window improves SB-RT/MLVU by 0.3/0.4 percentage points over the default fixed window in Table G.2. This suggests that accumulated visual change can be useful for selecting temporal support. The variant requires no additional LifeStream training, but its motion estimation and window selection introduce computation not measured by the fixed-configuration timing results in Table 7. We retain the fixed window as the main setting and report this variant as an exploratory extension.

## H Computational resources

The full evaluation across the main benchmarks requires approximately 150 GPU-hours per backbone on NVIDIA A100 GPUs, using the settings in Section 4.1. LifeStream updates no model parameters and requires no additional training.

## I Licenses for existing assets

We use public models and benchmarks for research evaluation. The following summary identifies their official sources and published licenses or usage terms.

| Asset | License or usage terms | Official source |
|---|---|---|
| LLaVA-OneVision-7B | Apache 2.0 | Model card |
| Qwen3-VL-8B | Apache 2.0 | Model card |
| StreamingBench | MIT (repository code) | Repository |
| OVO-Bench | CC BY-NC-SA 4.0 (dataset) | Repository |
| EgoSchema | Ego4D license | Project page |
| MLVU | CC BY-NC-SA 4.0 (dataset) | Repository |
| LongVideoBench | CC BY-NC-SA 4.0 (dataset) | Dataset card |
| VideoMME | Custom academic-research-only terms | Repository |

Code licenses do not extend to source videos, whose original rights and dataset terms still apply. We do not redistribute third-party assets.

## J Case studies

We provide qualitative examples to illustrate how LifeStream improves online video understanding under a compact visual token budget. The cases cover fine-grained action recognition, localized object counting, and long-range event reasoning, where preserving informative lifecycle tokens is critical for answering user queries.

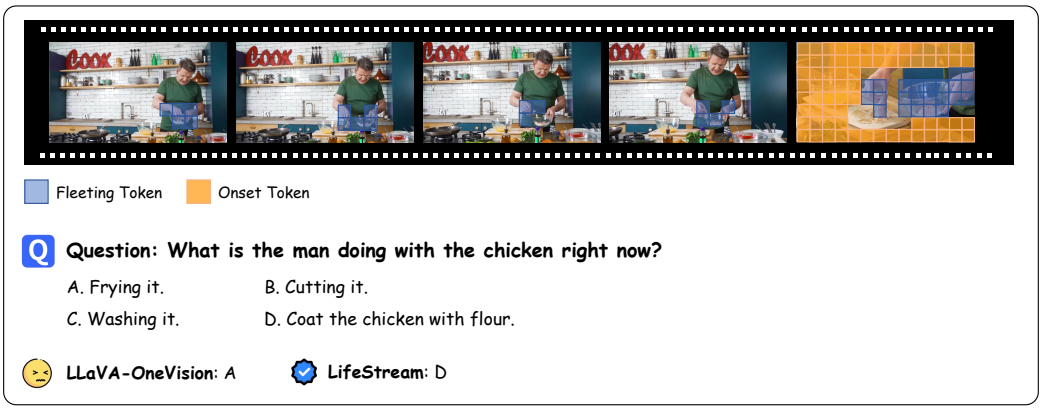



Figure J.1: **Case study on fine-grained action recognition.** The question asks what the man is doing with the chicken at the current moment. The base VLLM predicts "frying", while LifeStream correctly identifies that the chicken is being coated with flour. The visualized lifecycle tokens highlight transient and newly appearing evidence around the hands, chicken, and flour, showing that LifeStream preserves localized action cues that are easily missed under streaming context compression.

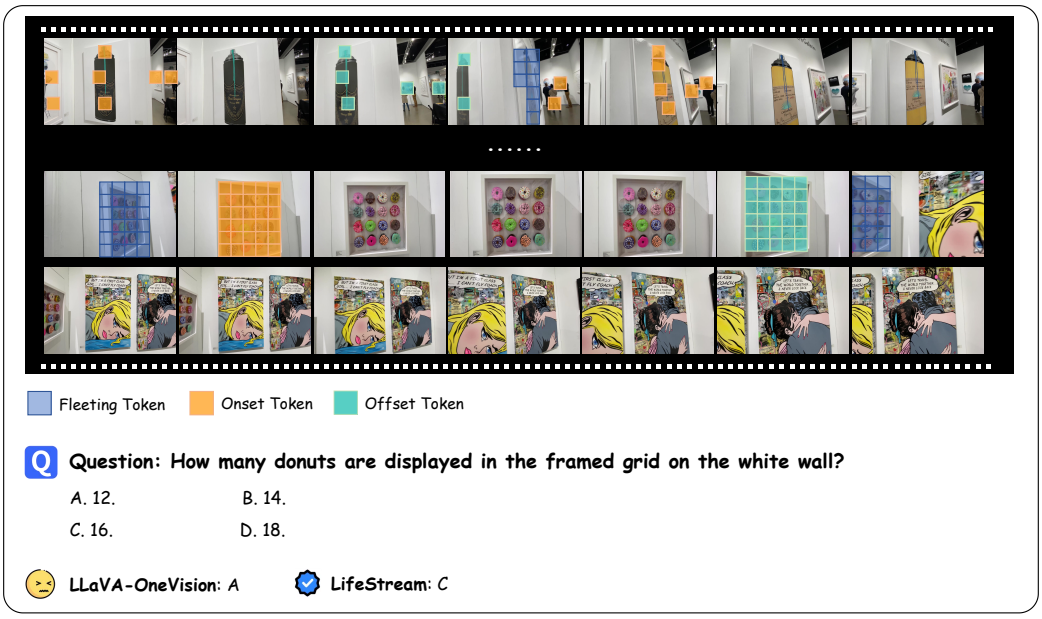



Figure J.2: **Case study on localized object counting.** The question requires counting the donuts in the framed grid on the white wall. The base VLLM predicts 12, while LifeStream correctly answers 16. By retaining non-stable tokens around the framed grid and suppressing redundant background regions, LifeStream better preserves the fine-grained visual evidence needed for accurate counting.

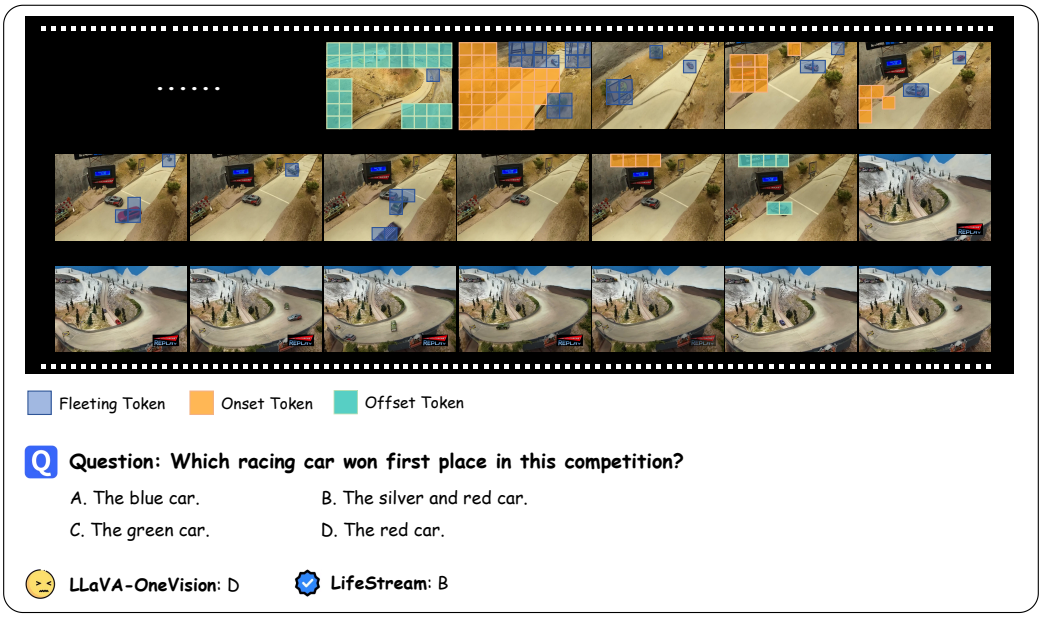



Figure J.3: **Case study on long-range event reasoning.** The question asks which racing car won first place. The base VLLM answers the red car, while LifeStream correctly predicts the silver and red car. The retained onset, offset, and fleeting tokens cover key racing moments and transition evidence across the stream, enabling LifeStream to recover the competition outcome from compact historical memory.

## NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims in the abstract and introduction accurately reflect the proposed training-free lifecycle-aware visual memory framework and its evaluated scope. The reported improvements and token reduction claims are supported by the experimental results on online and long-video understanding benchmarks.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 5 discusses the reliance on a predefined discrete lifecycle state space, manually specified retention priorities, and consolidation thresholds, and identifies adaptive state modeling and parameter selection as future directions.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper does not present formal theoretical results or claims that require theorem statements or proofs. The proposed method is an empirical, training-free framework for visual token selection and memory construction.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The main paper describes lifecycle state assignment and reports the evaluation settings in Section 4. Appendices B and C detail global consolidation and the complete streaming memory algorithm, respectively. Appendix D clarifies causal memory updates and query inference.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The paper provides algorithmic details, implementation settings, and hyperparameters, and the evaluated benchmarks and backbone models are publicly available under their respective licenses. The complete codebase, documentation, environment setup, data preparation instructions, and reproduction scripts will be open-sourced soon.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4.1 specifies the backbone models, frame sampling rate, chunk size, memory capacities, visual token budget, and hardware configuration. Section 4.2 describes the benchmarks, and Section 4.4 specifies the ablation settings. Further algorithmic details are provided in Appendices B and C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

Justification: The paper does not report error bars or statistical significance tests. The experiments are conducted on fixed benchmark splits with deterministic evaluation settings, and repeated full-scale evaluations across all backbones and benchmarks would require substantial compute.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: This paper reports the computing environment used for all experiments, as well as the estimated execution time (or GPU hours) for the main experiments. Furthermore, we note that no additional training was required, as LifeStream is a training-free framework that operates solely during the inference phase. For further details, please refer to Appendix H.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The authors have reviewed the NeurIPS Code of Ethics and the research conforms to it. The work uses public benchmarks and open-source models, and does not involve private data collection or human-subject experiments.

   Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: This work is primarily a training-free efficiency method for visual memory construction in existing video-language models. It does not introduce new data collection, new deployment scenarios, or application-specific systems beyond evaluation on standard public benchmarks.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release a new pretrained model, image generator, scraped dataset, or other high-risk asset. The proposed method is a training-free inference-time algorithm evaluated on existing public models and benchmarks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.

- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We use existing public benchmarks and open-source VLLM backbones only for research evaluation. All models, datasets, and benchmark assets used in our experiments are cited in the paper, and their official sources, licenses, or terms of use are summarized in the supplementary material; we do not redistribute these assets. See appendix I.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: No new datasets, pretrained models, or code assets are released with the current version. The implementation is described in the paper and supplementary material. The complete codebase, documentation, environment setup, data preparation instructions, and reproduction scripts will be open-sourced soon.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing experiments, participant studies, or newly collected human-subject data. It evaluates on existing public video understanding benchmarks.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects, and therefore IRB approval or equivalent review is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: The paper describes the use of LLM/VLLM backbones in the core experiments, including LLaVA-OneVision-7B and Qwen3-VL-8B. These models are used as frozen backbone models to evaluate the proposed training-free visual memory construction method.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.