Rethinking Long-Video Efficiency: A Joint Allocation Perspective on Frames, Pixels, and Front-End Latency

Sixun Dong[1] Wei Li[1] Andong Deng[1] Qi Qian[2] Victor Zhu[3] Zhengping Ji[3] Chen Chen[1]✉

[1]University of Central Florida [2]Meta Reality Labs [3]Axon

{sixundong,wei-li,chen.chen}@ucf.edu

Abstract

Efficient long-video understanding with vision-language models (VLMs) has largely been treated as informative token selection at fixed native resolution: which frames or which visual tokens to retain under a token budget. We argue this framing leaves two axes unused — per-frame resolution itself can be traded for denser temporal coverage, and front-end decoding latency scales with the candidate pool, not the token budget. Through an empirical study across multiple VLMs and long-video benchmarks, we distill three lessons: (i) dense low-resolution sampling outperforms sparse native-resolution sampling at matched token budgets; (ii) some tasks are resolution-sensitive and benefit from high-resolution frames; and (iii) front-end decoding dominates wall time on hour-scale clips. These lessons motivate LoHi, a training-free, single-pass framework that pairs a dense low-resolution video stream with a sparse set of high-resolution image streams, processed through the VLM's native video and image pathways. Two lightweight selectors choose Hi-I frames: LoHi-Anchor uses codec I-frame metadata, and LoHi-SemDiv uses a query-relevance and visual-diversity DPP over CLIP features. On three long-video benchmarks, LoHi improves over the vanilla native-resolution baseline by $+10.6\%$ on average at matched token budget and over the strongest prior efficiency methods by $+5.2\%$, while reducing front-end decoding latency by up to $7\times$ on hour-scale clips. Project page: https://sixundong.com/projects/lohi.

1 Introduction

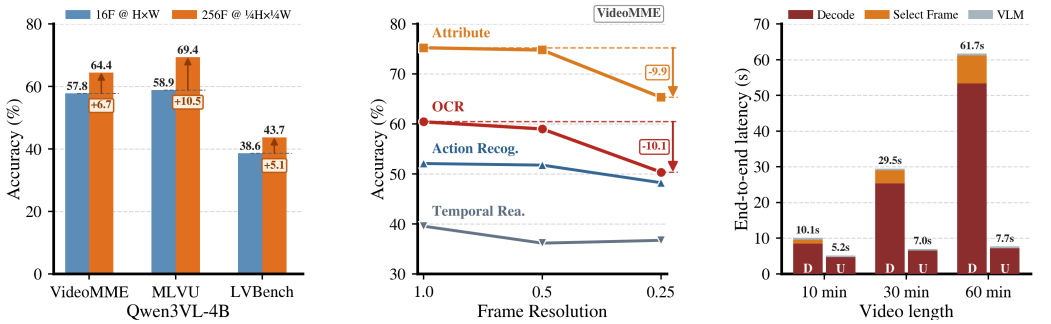
Recent vision-language models (VLMs) have made strong progress on static images and short videos [45, 16, 41, 1, 34]. Research is increasingly shifting toward long-video understanding, where task-relevant evidence is scattered across long temporal spans while decisive visual cues can be fleeting, resolution-sensitive, or dependent on long-term temporal relationships [29, 17, 8, 26, 25, 4].

Under a fixed visual-token budget, a long-video VLM can either sample more frames to achieve temporal coverage or use higher per-frame resolution to achieve spatial detail. However, existing efficient long-video VLMs largely treat this as an *informative token selection* problem at fixed native resolution: which frames to select [21, 11, 29, 43, 28, 47] or which visual tokens to retain [24, 6, 38, 8, 44] under the budget. This framing leaves two axes unused. First, per-frame resolution itself can be traded for temporal coverage — lowering resolution frees budget for more frames — yet resolution stays at native scale by default. Second, front-end latency, i.e., the time spent decoding the video and preprocessing frames before they reach the VLM, grows with video length and can exceed VLM inference itself, yet is rarely accounted for.

The central question is therefore not which informative tokens to keep, but how to jointly allocate an end-to-end budget across frame count, per-frame resolution, and front-end latency. We revisit this

✉ Corresponding author.

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



(a) A new simple recipe (b) Resolution sensitivity (c) The overlooked decode latency

Figure 1: **Three practical lessons on long-video understanding. (a)** At matched token budgets, dense low-resolution sampling substantially outperforms the default baseline. **(b)** Resolution-sensitive tasks (OCR, Attribute Perception) significantly drop under aggressive down-scaling. Resolution-insensitive tasks (Action Recognition, Temporal Reasoning) remain robust. **(c)** Decode-then-select(D) scale linearly with video length; while uniform fixed-budget(U) keeps front-end latency constant.

sparse native-resolution recipe through an empirical study across multiple VLMs and long-video benchmarks (Sec. 3), distilling three lessons.

- **L1. Dense low-resolution sampling is the new recipe.** At a fixed token budget, trading per-frame resolution for denser temporal coverage consistently outperforms the sparse native-resolution recipe on long-video benchmarks (Fig. 1a).
- **L2. Resolution sensitivity is task-dependent.** Moderate downscaling preserves most accuracy, but aggressive downscaling fails on resolution-sensitive tasks such as OCR and attribute perception (Fig. 1b). The remaining deficit on these tasks calls for a small amount of high-resolution evidence.
- **L3. Front-end latency dominates on long videos.** Decode-then-select pipelines build their candidate pool by densely decoding the video, with latency scaling linearly with duration; on hour-scale videos this front-end latency dominates wall time, while uniform fixed-budget sampling stays constant (Fig. 1c).

Together, these lessons point to a simple paradigm for long-video understanding. Start from dense low-resolution temporal coverage, then add only a small amount of high-resolution image evidence needed to recover the remaining spatial gap, while keeping front-end latency low. We instantiate this idea in **LoHi** (**Lo**w-resolution video + **Hi**gh-resolution images), a training-free framework that separates temporal coverage from spatial detail. LoHi uses a dense **Lo**w resolution **V**ideo **(Lo-V)** base to cover the full timeline and adds only a sparse set of **Hi**gh resolution **I**mages **(Hi-I)** to recover fine spatial evidence. These two streams are processed through the VLM's native *video* and *image* pathways and linked by an inline text marker, requiring no fine-tuning or architectural changes. Beyond this initial instantiation, LoHi is highly extensible and exposes a modular design space over Hi-I selection, integration, and adaptive triggering. We make three contributions.

- We frame long-video VLM efficiency as a joint allocation problem over frame count, per-frame resolution, and front-end latency, supported by three empirical lessons across multiple VLMs and benchmarks.
- We propose LoHi, a training-free framework that addresses this allocation problem by composing a dense low-resolution video stream with a sparse high-resolution image stream through the VLM's native video and image pathways.
- LoHi improves accuracy over the strongest keyframe-selection and token-pruning baselines at matched token budget, while reducing front-end decoding latency by up to 7× on hour-scale videos.

2 Related Work

The frame–resolution allocation problem under strict token budgets remains the central challenge in long-video understanding. Existing approaches can be broadly categorized into fixed-resolution methods and resolution-aware methods. Table 1 provides a qualitative comparison against LoHi.

Fixed-resolution paradigms. To manage strict token budgets, existing efficient long-video approaches generally adopt one of two paradigms. The first is pre-encoder keyframe selection, which filters frames based on semantic relevance to the query (e.g., AKS [29] adaptively partitions the timeline by query–frame similarity scores, and BOLT [17] uses inverse transform sampling). The second is post-encoder token pruning [38, 8, 42], which compresses visual features downstream of the vision encoder. By treating native per-frame resolution as a default, both paradigms severely limit the input frame count and sacrifice temporal coverage. Keyframe methods additionally incur front-end decode latency that scales with video length. LoHi breaks this assumption by decoupling temporal coverage (low-resolution video) from spatial detail (sparse high-resolution images).

Resolution-aware exploration. A small set of recent works explores downscaling per-frame resolution. VisionThink [39] lets the VLM decide whether high resolution is needed, adding autoregressive decoding overhead whose importance is highlighted by [5]. MMTok [6] pairs image-level downscaling with token pruning, but only on single images, leaving the temporal axis untouched. VideoLLaMA3 [41] introduces inter-frame compression, yet the temporal redundancy it exploits diminishes rapidly under sparse sampling. Q-frame [43] dynamically adapts resolution for query-relevant frames, but still relies on dense semantic scoring per candidate frame. LoHi instead keeps the Lo-V stream at a fixed resolution for temporal coverage and supplies spatial detail through sparse Hi-I frames. Both streams are processed in a single forward pass, without per-frame resolution decisions within Lo-V.

Table 1: **Positioning LoHi in long-video efficiency.** Per-frame cost and Front-end latency are graded as low (●), medium (●), or high (●).

	Token-efficiency at fixed resolution					*Resolution-aware*		Frames-Res
Axis	FastV [3]	VisionZip [38]	VideoLLaMA3 [41]	AKS [29]	BOLT [17]	MMTok [6]	VisionThink [39]	**LoHi (ours)**
Stage	Post-encoder	Post-encoder	Post-encoder	Post-decode	Post-decode	Post-encoder	Multi-turn	**Pre-decode**
Selection signal	Attn.	Sim.	Training.	Sim	Sim.	Coverage	LLM judgment	**Uniform**
Per-frame cost	●	●	●	●	●	●	●	●
Front-end latency	●	●	●	●	●	●	●	●
Modality	Video	Video	Video	Video	Video	Image	Image	**V+I**
Resolution Tunes	✗	✗	✗	✗	✗	✓	✓	✓
Train-free	✓	✓	✓	✓	✓	✓	✗	✓

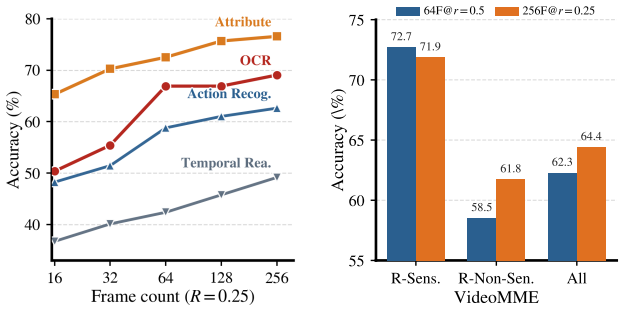
3 Lessons on Frames, Resolution, and Front-End Latency

Existing long-video VLMs allocate a fixed visual-token budget B to a single, uniform sampling configuration (N, r), where N is the frame count and $r \in (0, 1]$ is the per-frame resolution scale relative to the native resolution ($r = 1$). Let $\tau(r)$ denote the number of visual tokens generated per frame at resolution scale r. Under the strict budget constraint $N \cdot \tau(r) \leq B$, models are forced into a resource-constrained trade-off between temporal density and spatial fidelity. We comprehensively evaluate this two-dimensional configuration space across multiple VLMs and benchmarks. Our empirical analysis yields three lessons that motivate a new paradigm for long-video understanding.

3.1 Lesson 1 – Dense, low-resolution sampling is the new recipe

Tab. 2 evaluates three frame–resolution allocations (N, r) under the same fixed token budget ($\sim$5,760 tokens): $(16, 1)$, $(64, 0.5)$, and $(256, 0.25)$. The advantage of dense, low-resolution sampling persists across settings: for Qwen3-VL-4B, $(256, 0.25)$ improves over the default baseline by $+6.66\%$ on VideoMME, $+10.47\%$ on MLVU, and $+5.10\%$ on LVBench, with the same trend on Qwen3-VL-8B, and on VideoLLaMA3-7B for MLVU and LVBench. Fig. 2a isolates the source of this gain: with resolution fixed at r=0.25, accuracy increases with N from 16 to 256 for every VideoMME task category. The gains are largest when the default recipe critically under-samples the timeline, and smaller when 16 frames already provide sufficient coverage. This establishes $(256, 0.25)$ as a strong global baseline; whether it generalizes across task types is the question of Sec. 3.2.

> **Takeaway (L1)** Under a strict token budget, trading per-frame resolution for dense temporal coverage consistently outperforms the default sparse, native-resolution recipe. Dense, low-resolution sampling should serve as the new foundational baseline for long-video VLMs.



(a) Frame density monotone gain. (b) Frame–Resolution Allocation.

Video Length	Frame Sample	GoP Seeks	Decoded Frames	Decode Latency
10 min	Dense	600	∼18 K	8.5 s
	Uniform	256	∼15 K	4.7 s
30 min	Dense	1,800	∼54 K	25.3 s
	Uniform	256	∼27 K	6.5 s
60 min	Dense	3,600	∼108 K	53.3 s
	Uniform	256	∼27 K	7.2 s
VideoMME	Dense	∼272	∼24 K	14.6 s
	Uniform	∼136	∼9 K	3.9 s

(c) Front-end decode cost.

Figure 2: **Three lessons across frames, resolution, and front-end latency. (a)** At fixed low resolution (r=0.25), accuracy on VideoMME rises monotonically with frame count N grows from 16 to 256 for all four task categories (Sec. 3.1). **(b)** Under a matched token budget, resolution-sensitive tasks (R-Sens.) favor higher resolution over more frames, while resolution-insensitive tasks (R-Non-Sens.) prefer denser low-resolution sampling (Sec. 3.2). **(c)** Video decoder requires group-of-pictures (GoP) seeking, so constructing a dense candidate pool costs far more than uniform sampling(Sec. 3.3).

Table 2: Comparison of diverse frontier VLMs on different long-video benchmarks.

Setting	Qwen3-VL-4B			Qwen3-VL-8B			VideoLLaMA3-7B		
	VideoMME	MLVU	LVBench	VideoMME	MLVU	LVBench	VideoMME	MLVU†	LVBench
16F@1.0	57.78	58.89	38.61	60.04	58.95	38.22	46.81	35.88	32.09
64F@0.5	62.30	65.40	40.93	65.96	66.41	41.83	**64.37**	38.97	33.83
256F@0.25	**64.44**	**69.36**	**43.71**	**67.04**	**71.29**	**44.80**	50.04	**40.30**	**36.99**

† We report raw MLVU scores here; results after answer-format normalization are provided in Sec. C.7.

3.2 Lesson 2 – Optimal frame-resolution allocation is task-dependent

Although L1's baseline performs well on average, the budget couples N and r, forcing every operating point to trade temporal density against spatial fidelity. We refer to choosing along this trade-off as the *frame–resolution allocation problem*, and ask whether a single point on it can serve all task types. To answer this empirically, we rank VideoMME categories by their accuracy drop from r=1.0 to r=0.25 at N=16, and take the two categories with the largest drops (OCR and Attribute Perception) as the resolution-sensitive subset (R), with the remaining categories as its complement (¬R) (Fig. 1b). Under the same ∼5,760-token budget, the two operating points $(64, 0.5)$ and $(256, 0.25)$ swap dominance between the subsets (Fig. 2 (**b**)): the higher-resolution configuration wins on R, while the higher-frame configuration wins on ¬R. The aggregate winner therefore reflects benchmark subset proportions rather than a uniform task-level optimum, and the L1 baseline, despite its average advantage, is suboptimal on R. No single (N, r) pair can satisfy both subsets at the same budget.

> **Takeaway (L2)** Under a fixed token budget, no single (N, r) optimally serves all task types; any globally fixed allocation is suboptimal on at least one subset.

3.3 Lesson 3 – Front-end latency cannot be overlooked in long videos

Modern video codecs [36, 19] organize frames into Groups of Pictures (GoPs), where each GoP consists of a leading I-frame followed by a sequence of P/B-frames. An I-frame is a self-contained image and typically marks an abrupt motion or scene change, which is also where the codec inserts a new GoP boundary. For simplicity we focus on P-frames: a P-frame stores only the motion residual relative to its previous frame, ultimately chaining back to the I-frame. Sampling a P-frame inside a GoP therefore requires first decoding the I-frame and then sequentially decoding the preceding P-frames up to the target. For a policy that touches G GoPs and samples K frames at in-GoP depths $\{d_k\}_{k=1}^{K}$, the front-end decoding latency is approximately

$$T_{\text{dec}} \approx G \cdot t_I + \Big(\sum_{k=1}^{K} d_k\Big) \cdot t_P, \tag{1}$$

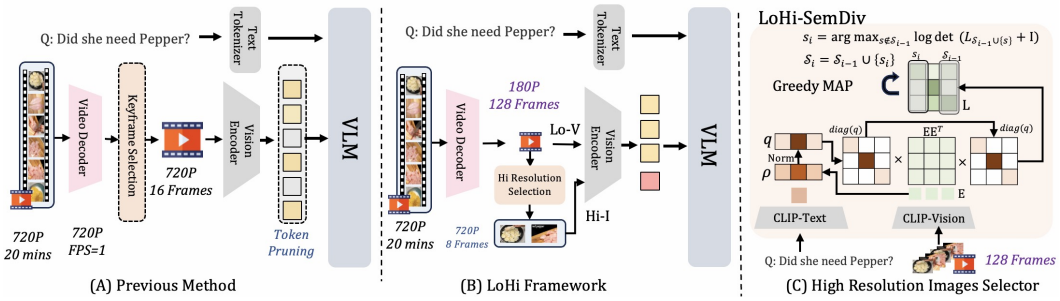


Figure 3: **Overview of LoHi framework.** Compared with previous work (a) which prunes or selects within a native resolution pathway, LoHi (b) decomposes one decoded video into two complementary streams: dense low-resolution *Lo-V* via the video pathway and sparse high-resolution *Hi-I* via the image pathway. (c) shows the LoHi-SemDiv selector, which picks Hi-I indices via greedy MAP.

where t_I and t_P denote the per-frame decoding latency of an I-frame and a P-frame, respectively. Decoding within each GoP is strictly sequential; including B-frames, which additionally reference future frames, only worsens this picture.

The overlooked decode latency. We profile Eq. (1) on videos of three lengths (10/30/60 min) and on VideoMME (Fig. 2 (**c**)). Dense sampling (FPS=1) drives both G and $\sum_k d_k$ linearly with duration, whereas uniform sampling at N=256 caps both. As a result, the latency gap widens monotonically with video length, reaching 7.4× on 60-min videos, and remains substantial on VideoMME itself, confirming this is not a stress-test artifact. This is precisely the cost paid by decode-then-select methods such as keyframe selection [29, 17], which build their candidate pool by densely decoding the video at FPS=1 in advance — a substantial front-end latency consistently overlooked by prior work. Method design and latency evaluation should therefore account for this front-end latency rather than report only the model-side token budget.

> **Takeaway (L3)** Under dense candidate-pool decoding such as FPS=1, front-end latency grows with video length and eventually dominates end-to-end runtime. Long-video efficiency methods must therefore bound the size of the decoded candidate pool, not just the visual-token budget passed to the VLM.

4 The LoHi Framework

We unify the lessons from Sec. 3 into a single problem statement: *frame–resolution allocation under a front-end latency constraint*. We first formalize it via a cross-resolution decomposition (Sec. 4.1), then instantiate it as the training-free single-pass framework **LoHi** (Sec. 4.2), enhance its high-resolution stream with a diversity-aware selector (Sec. 4.3), and finally add an entropy-triggered adaptive variant (Sec. 4.4).

4.1 Cross-Resolution Decomposition

To address the frame–resolution allocation problem (Sec. 3.2), we propose a *cross-resolution decomposition*: split the single configuration (N, r) into two complementary streams sharing one budget. A dense low-resolution stream (Lo-V) of N frames at scale r_ℓ, sampled uniformly throughout the video, provides temporal coverage (L1); a sparse high-resolution stream (Hi-I) of $K \ll N$ frames at scale $r_h > r_\ell$ supplies spatial detail (L2). The K Hi-I indices are drawn as a subset of the Lo-V grid $[N]$, so each frame is decoded once at native resolution and reused at both scales. The two streams jointly satisfy a token budget and a decode budget:

$$\underbrace{N \cdot \tau(r_\ell) \;+\; K \cdot \tau(r_h) \;\le\; B}_{\text{token budget (L1, L2)}}, \qquad \underbrace{N_{\text{dec}} \;=\; N}_{\text{decode budget (L3)}} . \tag{2}$$

VLM compute stays bounded by B, identical to a single-stream baseline. Front-end decode count N_{dec} depends only on the chosen N, not on video length: each of the N frames is decoded once at native resolution, then resized to r_ℓ for Lo-V and additionally kept at r_h for the K selected Hi-I. Compared with a single-stream baseline at $(N_{\text{base}}, r_{\text{base}})$ that decodes $N_{\text{base}} > N$ frames, e.g.,(256F,0.25), our decomposition saves $(N_{\text{base}} - N)$ decoded frames at the same token budget.

4.2 LoHi: Instantiating the Decomposition via Video and Image Pathways

Modern unified video-image VLMs (e.g., Qwen-VL series [1]) expose two input pathways sharing the same vision tower but differing in positional encoding: a *video pathway* with 3D temporal-spatial mRoPE [27, 32], and an *image pathway* treating each input as an independent 2D image. We instantiate the decomposition of Sec. 4.1 by routing the low-resolution stream through the video pathway and the high-resolution stream through the image pathway. mRoPE on the video pathway assigns each frame a globally consistent temporal position so motion is preserved at its natural rate; the image pathway gives each high-resolution image its own 2D positional encoding, leaving the video timeline undisturbed. This yields **LoHi**, a two-stream framework that combines a low-resolution video stream (**Lo-V**) with a high-resolution image stream (**Hi-I**).

We then bridge these two streams via a natural-language marker to tell the VLM the relationship between a high-resolution image and a low-resolution video.

> *[Video] The above low-resolution video provides temporal context. The following K high-resolution images show selected key frames: [img$_1$], ..., [img$_K$].*

LoHi-Uniform. Serving as the fundamental baseline of our method, LoHi-Uniform simply selects the K Hi-I frames at regular intervals directly from the Lo-V grid. By relying solely on mathematical indexing, this strategy incurs strictly zero additional computational overhead.

Codec-guided Hi-I selector. As established in Sec. 3.3, video encoders naturally anchor GoPs with I-frames, which can align with abrupt scene changes or significant motion. We propose a codec-aware selector, **LoHi-Anchor**, which explicitly exploits these pre-computed boundaries as a free semantic signal. It reads pre-computed I-frame indices without additional pixel decoding. For each uniformly spaced position u_m, it finds the nearest I-frame and selects the Lo-V grid frame closest to it. All Hi-I frames therefore come from the same decoded candidate pool (see Sec. A.1 for details).

While LoHi-Uniform and LoHi-Anchor both achieve negligible-cost selection by avoiding pixel-level scoring, they do not explicitly guarantee the visual distinctiveness of the sampled frames. To address this, we introduce a third selector that explicitly optimizes for semantic diversity.

4.3 LoHi-SemDiv: Diversity-Aware Selection via Quality-Similarity DPP

Inspired by the subset-selection framing of [6], we cast Hi-I selection as choosing K frames from the Lo-V grid $[N]$ given a query Q. Unlike MMTok, whose objective is to *cover* the original token sets, temporal coverage is already inherently supplied by our Lo-V base. Therefore, the K sparse Hi-I slots must instead be (i) *mutually diverse*, contributing distinct spatial evidence, and (ii) *query-relevant*, allocating the strict budget exclusively to content informative for Q.

The quality–similarity decomposition of determinantal point processes (DPPs) [13] elegantly unites these two criteria within a single submodular objective: it promotes relevance via the quality term while penalizing redundancy via the similarity kernel. While recent work [42] applies DPPs to token-level pruning within a single static image, we elevate this formulation to the temporal dimension, operating at frame granularity across two complementary resolution streams. We instantiate this diversity-aware, cross-stream selector on the Lo-V grid as **LoHi-SemDiv**. A detailed complexity analysis is provided in Sec. A.2.

DPP kernel. Let $e_j \in \mathbb{R}^d$ denote the L2-normalized CLIP [21] embedding of frame j, and let $\rho_j = \cos\big(\mathrm{CLIP}_{\mathrm{txt}}(Q),\, e_j\big)$ be the raw query–frame cosine similarity. We min-max normalize and power-sharpen ρ_j into a per-frame quality score q_j, then couple per-frame relevance with pairwise visual similarity via the quality–similarity decomposition[13]:

$$\tilde{\rho}_j = \frac{\rho_j - \min_m \rho_m}{\max_m \rho_m - \min_m \rho_m}, \qquad q_j = \tilde{\rho}_j^{\,\alpha}, \qquad L_{ij} = q_i\, q_j\, e_i^{\top} e_j. \tag{3}$$

Diagonals $L_{jj} = q_j^2$ encode per-frame relevance, off-diagonals couple relevance with pairwise visual similarity, so subsets of similar frames yield small $\det(L_{\mathcal{S}})$. Stacking the embeddings as $E = [e_1, \ldots, e_N]^{\top} \in \mathbb{R}^{N \times d}$ and qualities as $\mathbf{q} = (q_1, \ldots, q_N)^{\top}$, the kernel admits the equivalent matrix form

$$L = \mathrm{diag}(\mathbf{q})\, E E^{\top}\, \mathrm{diag}(\mathbf{q}), \tag{4}$$

which matches the construction in Fig. 3 (c) and is positive semi-definite. We set $\alpha = 5$.

Selection objective. For a theoretically grounded greedy approximation, we adopt the regularized log-determinant:

$$\mathcal{S}^* = \arg\max_{|\mathcal{S}|=K} g(\mathcal{S}), \qquad g(\mathcal{S}) = \log\det\big(L_{\mathcal{S}} + \mathbf{I}\big). \tag{5}$$

The identity regularization ensures all eigenvalues of $L_{\mathcal{S}} + \mathbf{I}$ are at least one, so g is non-negative, monotone non-decreasing, and submodular [13]. This admits a near-optimal greedy approximation:

Proposition 1 ([18]). *Let $\mathcal{S}$ be the subset obtained by greedy maximization of g. Then*

$$g(\mathcal{S}) \;\geq\; (1 - 1/e) \max_{|\mathcal{A}|=K} g(\mathcal{A}).$$

4.4 Adaptive LoHi via Entropy Triggering

While the Hi-I stream provides crucial spatial details, processing K high-resolution frames incurs additional computational overhead. However, because many queries can be answered correctly using the Lo-V base alone, our decoupled two-stream architecture naturally supports an adaptive, per-question early-exit mechanism. Specifically, we first evaluate the query using the lightweight Lo-V base and compute the entropy of the generated answer distribution. The Hi-I stream is triggered only if the model's uncertainty exceeds a pre-defined threshold θ:

$$\text{Trigger Hi-I} \iff H\big(p_{\text{Lo-V}}(\cdot \mid Q)\big) > \theta. \tag{6}$$

This entropy-triggered protocol reduces the average computational cost by bypassing the Hi-I stream for simple queries. As detailed in Sec. C.6, this adaptive variant nearly matches the performance of the full LoHi pipeline while invoking the high-resolution stream on only one-third of the queries.

5 Experiments

Setup. We evaluate LoHi on three long-video benchmarks—VideoMME (without subtitles) [10], MLVU [46], and LVBench [33]. Additional results with subtitles are provided in Sec. C.11. We employ Qwen3-VL-4B [1] as the primary backbone and evaluate generalization on Qwen3-VL-8B (Sec. C.1); Qwen3.5-4B [20] (Sec. C.2); and VideoLLaMA3-7B [41], VideoChat3-4B [15], and Qwen2.5-VL-7B [2] (Sec. C.3). We compare against representative keyframe-selection methods (AKS [29], BOLT [17]) and two token-pruning methods (FlashVid [8] for video VLMs and VisionZip [38] for image VLMs). We also include CLIP-Topk as a keyframe-selection baseline.

Implementation details. Following L3, all methods cap the decoded frames per video at $N_{\text{dec}} \leq 256$ and share the visual-token budget of the default $(16, 1.0)$ recipe ($\sim$5,760 tokens). Key-frame selection baselines decode 256 frames at r=1.0 and pick 16; token-pruning baselines decode 256 at r=1.0 and prune 93.75% of the tokens. LoHi uses N=128 at r_ℓ=0.25 for Lo-V and K=8 Hi-I indices drawn from these 128 at r_h=1.0, giving N_{dec}=128 — half of the baselines at the same token budget. We denote this as 128F@0.25+Hi-I. Full details are in Sec. B.

5.1 Performance Comparison across different benchmarks

Comparison on Qwen3-VL-4B. Tab. 3 reports accuracy at a default token budget. LoHi-SemDiv achieves the best performance on all three benchmarks, reaching 66.81 / 74.38 / 45.90% on VideoMME / MLVU / LVBench, an average gain of +10.60% over the vanilla baseline (16F,1.0), and +5.23% over the strongest prior baseline (BOLT[17]) at identical token cost. Notably, **Low-Res-Base**, the dense low resolution video stream (256F@0.25) already outperforms every prior keyframe-selection and token-pruning baseline on average, supporting L1 that varying r alone provides a competitive recipe at fixed budget without any selection logic. Adding Hi-I extends the lead, with LoHi-SemDiv > LoHi-Anchor > LoHi-Uniform on every benchmark and the largest semantic-diversity contribution on MLVU (+6.76% over LoHi-Uniform). Paired significance tests and task-level analyses are provided in Sec. C.10. Crucially, LoHi decodes only 128 frames per video—half the 256-frame pool consumed by every prior efficiency baseline. Specifically, LoHi-Uniform and LoHi-Anchor rely entirely on simple frame indices or free metadata, which is almost zero cost. LoHi-SemDiv adds one CLIP forward pass over the 128 low-resolution frames and has lower end-to-end latency than token pruning and keyframe-selection methods (see Tab. 5 for details). We also evaluate our method on Qwen3-VL-8B in Sec. C.1.

Table 3: Performance Comparison on Qwen3-VL-4B. Avg. is the mean over three benchmarks.

Method	#Frames (Decoded)	Setting ($NF@r$)	Tokens (Budget)	VideoMME (%) Short	Med.	Long	All	MLVU (%)	LVBench (%)	Avg. (%)
Qwen3-VL-4B	16	16F@1.0	5,760	68.44	55.67	49.22	57.78	58.89	38.61	51.76
The new recipe: dense low-resolution video										
Low-Res-Base	256	256F@0.25	5,760	75.78	62.33	55.22	64.44	69.36	43.71	59.17
Pre-encoder keyframe selection										
CLIP-TopK	256 → 16	16F@1.0	5,760	65.67	56.33	49.56	57.19	69.36	44.74	57.10
AKS [29]	256 → 16	16F@1.0	5,760	65.89	56.22	49.67	57.26	69.22	44.67	57.05
BOLT [17]	256 → 16	16F@1.0	5,760	71.44	59.00	52.00	60.81	68.11	42.48	57.13
Post-encoder token pruning. Pruning Ratio:93.75% to keep same token budget										
VisionZip[38]	256	256F@1.0	5,760	64.78	55.67	50.56	57.00	62.46	35.13	51.53
FlashVid[8]	256	256F@1.0	5,760	69.56	58.11	50.22	59.30	63.70	34.09	52.36
LoHi (ours)										
LoHi-Uniform	128	128F@0.25+Hi-I	5,760	75.67	**66.44**	55.22	65.78	67.62	44.09	59.16
LoHi-Anchor	128	128F@0.25+Hi-I	5,760	76.00	66.22	55.44	65.89	68.40	44.42	59.57
LoHi-SemDiv	128	128F@0.25+Hi-I	5,760	**77.11**	65.78	**57.56**	**66.81**	**74.38**	**45.90**	**62.36**

Table 4: Frame-Resolution Allocation.

Qwen3-VL-4B	Tokens	R (%)	¬R (%)	All (%)
Under Single Resolution and Frames setting (N, r).				
64F@0.5	5,760	<u>72.73</u>	58.54	62.30
256F@0.25	5,760	71.89	<u>61.76</u>	<u>64.44</u>
LoHi with Hi-I Selector				
LoHi-Uniform	5,760	76.64	61.86	65.78
LoHi-Anchor	5,760	75.38	62.47	65.89
LoHi-SemDiv	5,760	**77.06**	**63.12**	**66.81**

Table 5: Comparison of Inference Efficiency.

Method	Decode s	Sel. ms	ViT ms	TTFT ms	TFLOPs	Mem. +8.3GB	VideoMME ACC
256F@0.25	5.4	50	216	5,963	82.2	1.3	64.44
Key-frame	5.4	130	151	5,978	89.3	1.3	60.81
VisionZip	5.4	4	2,486	8,186	464.3	15.4	57.00
FlashVid	5.4	808	2,486	8,990	464.3	15.4	59.30
LoHi-Uniform	3.3	51	195	**3,836**	84.6	1.3	65.78
LoHi-Anchor	3.3	51	195	**3,836**	84.6	1.3	65.89
LoHi-SemDiv	3.3	120	195	**3,906**	85.8	1.3	**66.81**

LoHi resolves the trade-off between resolution-sensitive (R) and insensitive (¬R) tasks. As demonstrated in L2 and Tab. 4, R and ¬R tasks inherently demand divergent configurations. By leveraging complementary representations from both the Lo-V and Hi-I streams, all three proposed LoHi variants simultaneously outperform the single-configuration baselines on both subsets. This consistent gain stems from maintaining dual operating points across the video and image pathways, rather than relying on any specific Hi-I selector.

End-to-End Inference Efficiency. Tab. 5 profiles the Time-to-First-Token (TTFT) wall time. Unlike prior baselines that decode 256 frames, LoHi decodes only 128 frames (L3), immediately reducing front-end latency from 5.4 s to 3.3 s. The Key-frame row represents AKS[29], BOLT[17] and CLIP-TopK. More crucially, token pruning methods (e.g., VisionZip, FlashVid) exhibit a severe Vision Tower bottleneck (464.3 TFLOPs; 2,486 ms) because they encode native-resolution frames before pruning. By avoiding dense native-resolution encoding, LoHi reduces ViT latency by approximately 12× (from 2,486 to 195 ms), with a measured GPU memory footprint of 9.6 GB. Compared with the dense low-resolution baseline, LoHi-SemDiv improves VideoMME accuracy from 64.44% to 66.81% while using half as many sampled frames (128 vs. 256) and reducing TTFT from 5,963 to 3,906 ms. This result illustrates how joint allocation improves both accuracy and efficiency.

5.2 Rethinking Token Pruning for Video-VLMs

Table 6: Token pruning vs Resize.

Method	Setting	VideoMME	LVBench	MLVU	Avg.
Qwen3-VL-4B	16F@1.0	57.78	38.61	58.89	51.76
25% default budget (∼1,440 tokens)					
VisionZip [38]	16F@1.0	55.56	34.47	53.75	47.93
MMTok [6]	16F@1.0	55.19	34.80	54.82	48.27
FlashVid [8]	16F@1.0	55.11	35.44	54.54	48.36
Resize	16F@0.5	**56.26**	**35.51**	**57.95**	**49.91**
50% default budget (∼2,880 tokens)					
VisionZip [38]	128F@1.0	54.44	35.51	58.99	49.65
MMTok [6]	128F@1.0	59.56	36.86	62.24	52.89
FlashVid [8]	128F@1.0	58.19	35.77	60.67	51.54
Resize	128F@0.25	**63.00**	**39.25**	**66.34**	**56.20**

Token pruning falls short of simple low-resolution baselines. To analyze the limitations of token pruning exhibited in Tab. 3, we further benchmark pruning methods against a naive resize baseline under fixed token budgets (25% and 50%; Tab. 6), including MMTok [6], which is explicitly optimized for extreme low-retention regimes. Despite feeding identical frame counts and tokens to the LLM, their pipelines differ fundamentally: resizing downsamples frames upfront (drastically reducing Vision Tower FLOPs), whereas pruning relies on costly native-resolution processing before discarding tokens. Strikingly, simple resizing consistently matches or outperforms all dedicated

pruning methods. This exposes a fundamental inefficiency in current paradigms, proving that resizing is both computationally cheaper and empirically superior.

Spatiotemporal consistency is critical for Video-VLMs. Although effective for static images [38, 6], token pruning degrades temporal position modeling in videos. Discarding irregular patches across frames destroys both spatial continuity and temporal consistency [8]. Consequently, preserving a complete, lower-resolution global context is strictly more critical for dynamic video comprehension than retaining fragmented, high-resolution details.

5.3 Ablations

We ablate the three core design axes of LoHi on VideoMME using Qwen3-VL-4B: the number of high-resolution images (Hi-I$_K$), the high resolution image selection strategy, and the necessity of dual pathways.

Table 7: K of Hi-I ablation across selectors.

Selector	K=2	K=4	K=8
LoHi-Uniform	65.44	65.52	65.78
LoHi-Anchor	65.22	65.85	65.89
LoHi-CLIP Top-K	65.37	66.00	66.44
LoHi-SemDiv	**65.59**	**66.22**	**66.81**

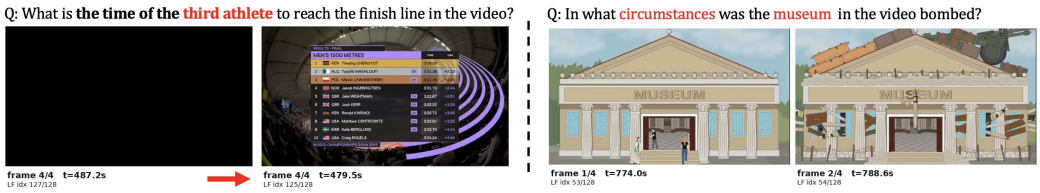
Table 8: Lo-V / Hi-I necessity ablation.

Lo-V	Hi-I	Tokens	Uniform	SemDiv
256F	-	5,760	64.44	–
-	8	2,880	55.19	60.85
-	16	5,760	60.41	63.74
128F	8	5,760	65.78	66.81

Robustness across frame counts and selectors. To evaluate the architectural stability of LoHi, we investigate its sensitivity to the high-resolution frame count ($K \in \{2, 4, 8\}$) and four training-free selection strategies (Tab. 7). All 12 configurations achieve 65.22–66.81% accuracy on VideoMME, exceeding the 128-frame Lo-V-only baseline (63.00%). This consistency suggests that the gains are not specific to a single selector or Hi-I budget. SemDiv further improves over Uniform at every budget, highlighting the benefit of query-aware, diversity-aware selection. Beyond these small Hi-I budgets, we also evaluate LoHi on EgoLongQA [30] with Qwen3.5-27B, where it matches the accuracy of a higher-resolution single-stream baseline using 29% fewer visual tokens (Sec. C.5).

Lo-V and Hi-I are strictly complementary. To prove the necessity of both pathways (Lo-V and Hi-I), we isolate their individual contributions under a strictly controlled budget of 5,760 tokens (Tab. 8). Relying exclusively on dense low-resolution video (256 Lo-V frames) or sparse high-resolution images (Hi-I$_{K=16}$, SemDiv) yields suboptimal performance at 64.44% and 63.74%, respectively. In stark contrast, integrating both pathways—allocating tokens to Lo-V with 128 frames and Hi-I with 8 images—boosts accuracy to 66.81%. This confirms our core hypothesis: dense temporal context (Lo-V) and precise spatial clarity (Hi-I) provide orthogonal, non-replaceable benefits.

Qualitative analysis. Fig. 4 visualizes Hi-I frames chosen by the two selectors. (a) LoHi-Anchor replaces a black frame with the Lo-V grid frame closest to the nearest I-frame, recovering the scoreboard. (b) LoHi-SemDiv picks two query-relevant frames spanning the critical state transition of the museum before and after the bombing. More cases are provided in Sec. F.



(A) LoHi-Anchor replaces the 4th Hi-I: 127 → 125, guided by I-frames.

(B) LoHi-SemDiv selects the query-aware frames

Figure 4: Visualization of selected high-resolution images by our proposed Hi-I Selector.

6 Conclusion

We revisit long-video VLM efficiency as a joint allocation problem over frame count, per-frame resolution, and front-end latency. This perspective motivates LoHi, a training-free, single-pass framework combining dense low-resolution video with sparse high-resolution images. At the same visual-token budget, LoHi improves average accuracy by 10.6% over the native-resolution baseline and reduces front-end latency by up to 7× on hour-scale clips. Its plug-and-play selectors support different cost–accuracy trade-offs. Future work can explore richer selectors and adaptive allocation.

Acknowledgments and Disclosure of Funding

This work received no third-party funding. The authors declare no competing interests.

References

[1] 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.

[2] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[3] Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *European Conference on Computer Vision*, pages 19–35. Springer, 2024.

[4] Sixun Dong, Huazhang Hu, Dongze Lian, Weixin Luo, Yicheng Qian, and Shenghua Gao. Weakly supervised video representation learning with unaligned text for sequential videos. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2437–2447. IEEE, 2023.

[5] Sixun Dong, Juhua Hu, Steven Li, Wei Wen, and Qi Qian. Rethinking model efficiency: Multi-agent inference with large models. *arXiv preprint arXiv:2604.04929*, 2026.

[6] Sixun Dong, Juhua Hu, Mian Zhang, Ming Yin, Yanjie Fu, and Qi Qian. MMTok: Multimodal coverage maximization for efficient inference of VLMs. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=GvPdSWZT31.

[7] Xiaoyi Dong, Pan Zhang, Yuhang Zang, Yuhang Cao, Bin Wang, Linke Ouyang, Songyang Zhang, Haodong Duan, Wenwei Zhang, Yining Li, et al. Internlm-xcomposer2-4khd: A pioneering large vision-language model handling resolutions from 336 pixels to 4k hd. *Advances in Neural Information Processing Systems*, 37:42566–42592, 2024.

[8] Ziyang Fan, Keyu Chen, Ruilong Xing, Yulin Li, Li Jiang, and Zhuotao Tian. Flashvid: Efficient video large language models via training-free tree-based spatiotemporal token merging. *arXiv preprint arXiv:2602.08024*, 2026.

[9] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6202–6211, 2019.

[10] 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.

[11] Bo He, Hengduo Li, Young Kyun Jang, Menglin Jia, Xuefei Cao, Ashish Shah, Abhinav Shrivastava, and Ser-Nam Lim. Ma-lmm: Memory-augmented large multimodal model for long-term video understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13504–13514, 2024.

[12] Huazhang Hu, Sixun Dong, Yiqun Zhao, Dongze Lian, Zhengxin Li, and Shenghua Gao. Transrac: Encoding multi-scale temporal correlation with transformers for repetitive action counting. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18991–19000. IEEE, 2022.

[13] Alex Kulesza and Ben Taskar. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2-3):123–286, 2012.

[14] Bo Li, Peiyuan Zhang, Kaichen Zhang, Fanyi Pu, Xinrun Du, Yuhao Dong, Haotian Liu, Yuanhan Zhang, Ge Zhang, Chunyuan Li, and Ziwei Liu. LMMs-Eval: Accelerating the development of large multimodal models, March 2024. URL https://github.com/EvolvingLMMs-Lab/lmms-eval.

[15] Xinhao Li, Yuhan Zhu, Xiangyu Zeng, Yuhao Dong, Haoning Wu, Zhiqiu Zhang, Yuandong Yang, Changlian Ma, Qingyu Zhang, Yansong Shi, et al. Videochat3: Fully open video mllm for efficient and generalist video understanding. *arXiv preprint arXiv:2607.14935*, 2026.

[16] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge, January 2024. URL https://llava-vl.github.io/blog/2024-01-30-llava-next/.

[17] Shuming Liu, Chen Zhao, Tianqi Xu, and Bernard Ghanem. Bolt: Boost large vision-language model without training for long-form video understanding. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 3318–3327, 2025.

[18] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14(1):265–294, 1978.

[19] Amal Punchihewa and Donald Bailey. A review of emerging video codecs: Challenges and opportunities. In *2020 35th International Conference on Image and Vision Computing New Zealand (IVCNZ)*, pages 1–6. IEEE, 2020.

[20] Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

[21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[22] Sayan Deb Sarkar, Rémi Pautrat, Ondrej Miksik, Marc Pollefeys, Iro Armeni, Mahdi Rad, and Mihai Dusmanu. Cope-videolm: Leveraging codec primitives for efficient video language modeling. *arXiv preprint arXiv:2602.13191*, 2026.

[23] Benjamin Schneider, Dongfu Jiang, Chao Du, Tianyu Pang, and Wenhu Chen. Quick-video: Real-time long video understanding with system algorithm co-design. *arXiv preprint arXiv:2505.16175*, 2025.

[24] Leqi Shen, Guoqiang Gong, Tao He, Yifeng Zhang, Pengzhang Liu, Sicheng Zhao, and Guiguang Ding. Fastvid: Dynamic density pruning for fast video large language models. *arXiv preprint arXiv:2503.11187*, 2025.

[25] 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.

[26] 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.

[27] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[28] Guangyu Sun, Archit Singhal, Burak Uzkent, Mubarak Shah, Chen Chen, and Garin Kessler. From frames to clips: Efficient key clip selection for long-form video understanding. *arXiv e-prints*, pages arXiv–2510, 2025.

[29] 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.

[30] Tuyen (Harry) Tran, Maxim Arap, Seungwhan Moon, Raffay Hamid, Alessandro Suglia, Zsolt Kira, Pascale Fung, and Mubarak Shah. Egowearbench: A benchmark suite for long-context, conversational, and proactive egocentric ai. https://wearable-ai-workshop.github.io/, 2026. Workshop at the European Conference on Computer Vision (ECCV) 2026.

[31] Chenyu Wang, Weixin Luo, Sixun Dong, Xiaohua Xuan, Zhengxin Li, Lin Ma, and Shenghua Gao. Mllm-tool: A multimodal large language model for tool agent learning. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 6678–6687. IEEE, 2025.

[32] 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.

[33] Weihan Wang, Zehai He, Wenyi Hong, Yean Cheng, Xiaohan Zhang, Ji Qi, Ming Ding, Xiaotao Gu, Shiyu Huang, Bin Xu, et al. Lvbench: An extreme long video understanding benchmark. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22958–22967, 2025.

[34] 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.

[35] Haoran Wei, Yaofeng Sun, and Yukun Li. Deepseek-ocr: Contexts optical compression. *arXiv preprint arXiv:2510.18234*, 2025.

[36] Thomas Wiegand, Gary J Sullivan, Gisle Bjontegaard, and Ajay Luthra. Overview of the h. 264/avc video coding standard. *IEEE Transactions on circuits and systems for video technology*, 13(7):560–576, 2003.

[37] Chao-Yuan Wu, Manzil Zaheer, Hexiang Hu, R Manmatha, Alexander J Smola, and Philipp Krähenbühl. Compressed video action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6026–6035, 2018.

[38] 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.

[39] Senqiao Yang, Junyi Li, Xin Lai, Bei Yu, Hengshuang Zhao, and Jiaya Jia. Visionthink: Smart and efficient vision language model via reinforcement learning. *arXiv preprint arXiv:2507.13348*, 2025.

[40] Ming Yin, Dinghan Shen, Silei Xu, Sixun Dong, Mian Zhang, Yebowen Hu, Shujian Liu, Jianbing Han, Simin Ma, Song Wang, et al. Livemcp-101: Stress testing and diagnosing mcp-enabled agents on challenging queries. *arXiv preprint arXiv:2508.15760*, 2025.

[41] 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.

[42] Qizhe Zhang, Mengzhen Liu, Lichen Li, Ming Lu, Yuan Zhang, Junwen Pan, Qi She, and Shanghang Zhang. Beyond attention or similarity: Maximizing conditional diversity for token pruning in mllms. *arXiv preprint arXiv:2506.10967*, 2025.

[43] Shaojie Zhang, Jiahui Yang, Jianqin Yin, Zhenbo Luo, and Jian Luan. Q-frame: Query-aware frame selection and multi-resolution adaptation for video-llms. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22056–22065, 2025.

[44] Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis A Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, et al. Sparsevlm: Visual token sparsification for efficient vision-language model inference. In *International Conference on Machine Learning*, pages 74840–74857. PMLR, 2025.

[45] Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. Llava-video: Video instruction tuning with synthetic data. *arXiv preprint arXiv:2410.02713*, 2024.

[46] 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.

[47] Yuanhao Zou, Shengji Jin, Andong Deng, Youpeng Zhao, Jun Wang, and Chen Chen. A.i.r.: Enabling adaptive, iterative, and reasoning-based frame selection for video question answering. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=SZVpOKw0YD.

[48] Yulin Zou, Yan Chen, Wenyan Chen, JooYoung Park, Shivaraman Nitin, Luo Tao, Francisco Romero, and Dmitrii Ustiugov. Costream: Codec-guided resource-efficient system for video streaming analytics. *arXiv preprint arXiv:2604.06036*, 2026.

A Extra Method and Experiment Details

This appendix supplements Sec. 4. It gives the formal definition of LoHi-Anchor, the algorithm and complexity of LoHi-SemDiv, and two exploratory variants that extend the SemDiv selector: LoHi-SemDiv-Anchor, which combines the codec and semantic signals, and LoHi-SemDiv++, which conditions the selection on the Lo-V stream.

A.1 LoHi-Anchor

LoHi-Anchor starts from the K uniformly spaced indices $\{u_1, \dots, u_K\}$ used by LoHi-Uniform. For each position, it considers the five nearest I-frame candidates to avoid duplicate selections, then maps the selected I-frame back to the nearest Lo-V grid index:

$$h_m = \arg \min_{i \in \{1,\dots,N\}} |p_i - j_m|, \qquad \mathcal{H}_{\text{Anchor}} = \{h_m\}_{m=1}^{K}, \tag{7}$$

where p_i is the original-video frame index of the i-th Lo-V grid frame, and j_m is the I-frame index selected for u_m. I-frame indices are retrieved using `decord.VideoReader.get_key_indices()` (Sec. B.4). No temporal-distance threshold is applied. If keyframe retrieval fails or returns no keyframes, the selector falls back to LoHi-Uniform.

All Hi-I frames are selected from the same decoded Lo-V candidate pool and resized at high resolution, requiring no additional frame decoding. The selector runs no CLIP forward pass and does not use the question.

A.2 LoHi-SemDiv

Algorithm 1 shows the greedy MAP procedure for the regularized log-determinant objective $g(\mathcal{S}) = \log\det(L_{\mathcal{S}} + \mathbf{I})$ in Eq. (5). At each step, the algorithm picks the frame that increases g the most given the frames already selected. By Proposition 1, the resulting set is within a $(1{-}1/e)$ factor of the optimum.

Algorithm 1 LoHi-SemDiv: greedy MAP for the regularized log-det objective.

Input: kernel $L \in \mathbb{R}^{N\times N}$, budget K, objective g (Eq. (5)).
Initialize $\mathcal{S} \leftarrow \emptyset$.
for $i = 1, \dots, K$ **do**
  $s_i \leftarrow \arg\max_{s \in [N]\setminus\mathcal{S}} \; g\big(\mathcal{S} \cup \{s\}\big)$
  $\mathcal{S} \leftarrow \mathcal{S} \cup \{s_i\}$
end for
return $\mathcal{S}$ sorted in temporal order.

With rank-one Cholesky updates on $L_{\mathcal{S}} + \mathbf{I}$, the procedure runs in $\mathcal{O}(NK^2)$ time. On top of the Lo-V decoding that LoHi performs anyway, SemDiv adds only one CLIP forward pass over the N low-resolution frames. Both costs are negligible compared with VLM prefill.

A.3 LoHi-SemDiv-Anchor

LoHi-SemDiv-Anchor combines the two selectors above. It runs the SemDiv greedy search on snapped positions, so the diversity kernel scores exactly the frames that the VLM will receive. At each greedy step, the candidate index is snapped to the nearest Lo-V grid index associated with a nearby I-frame, subject to a radius $t(D, K)$, and the kernel is then evaluated at the snapped position. The radius is measured in Lo-V grid steps and depends on the clip duration D and the budget K:

$$t(D, K) = \begin{cases} 0 & D < 180\,\text{s}, \\ \lfloor 0.10\, N/K \rfloor & D \geq 180\,\text{s}, \end{cases} \tag{8}$$

which gives $t = 3$ at $K = 4$ and $t = 1$ at $K = 8$ for $N = 128$ on long clips. On short clips, the snap is disabled because moving an index could shift it beyond the short time span that a typical

action occupies; the variant then reduces to plain SemDiv. To avoid duplicate grid selections, the snap checks the four nearest I-frames in order of distance and accepts the first whose mapped Lo-V grid index lies within the radius and has not been selected; if none qualifies, the index is left unchanged. Algorithm 2 shows the full loop.

Algorithm 2 LoHi-SemDiv-Anchor: snap-aware greedy MAP.

Input: kernel $L \in \mathbb{R}^{N\times N}$, budget K, I-frame set $\mathcal{I}$, duration D, threshold $t(D, K)$.
Initialize $\mathcal{S}_{\text{snap}} \leftarrow \emptyset$.
for $i = 1, \ldots, K$ **do**
  $j^\star \leftarrow \arg\max_{j\in[N]\setminus\mathcal{S}_{\text{snap}}} \log\det\big(L_{\mathcal{S}_{\text{snap}}\cup\{\tau(j;\,\mathcal{S}_{\text{snap}})\}} + \mathbf{I}\big)$
  $\mathcal{S}_{\text{snap}} \leftarrow \mathcal{S}_{\text{snap}} \cup \{\tau(j^\star;\,\mathcal{S}_{\text{snap}})\}$
end for
return $\mathcal{S}_{\text{snap}}$ sorted in temporal order.

Here, $\tau(j; S)$ denotes the snapped Lo-V grid index returned by the procedure above, or j if no valid candidate is found.

We set the constants (the 0.10 fraction, the 180 s threshold, and the four snap candidates) by rough reasoning rather than tuning. The procedure stays $\mathcal{O}(NK^2)$ and requires only keyframe-index lookup in addition to the CLIP inference used by SemDiv. On VideoMME (Tab. 9), the variant improves over plain SemDiv by $+0.19\%$ at $K{=}4$ and $+0.08\%$ at $K{=}8$, mostly on medium-length clips, while at $K{=}2$ the two selectors perform on par. Given the small gains, we regard this variant as exploratory.

Table 9: LoHi-SemDiv-Anchor on VideoMME (Qwen3-VL-4B, 128F Lo-V, interleaved cue).

Selector	K	Short	Medium	Long	All
LoHi-SemDiv	2	75.67	65.78	55.33	65.59
LoHi-SemDiv-Anchor	2	75.67	65.67	55.22	65.52
LoHi-SemDiv	4	76.22	65.22	57.22	66.22
LoHi-SemDiv-Anchor	4	76.22	66.00	57.00	**66.41**
LoHi-SemDiv	8	77.11	65.78	57.56	66.81
LoHi-SemDiv-Anchor	8	77.11	66.00	57.56	**66.89**

A.4 LoHi-SemDiv++: Lo-V-Conditioned Hi-I Selection

SemDiv in Sec. 4.3 rests on a simple premise: the Lo-V base already covers the timeline, so the K Hi-I frames only need to be relevant to the query and different from each other. Its quality–similarity DPP is built from the CLIP features alone and never refers to the Lo-V base. A natural question is whether an explicit reference to Lo-V would help, that is, whether Hi-I should target the visual content that Lo-V does *not* already capture.

LoHi-SemDiv++ does this with a PCA projection. Let $\{e_j^\ell\}_{j=1}^N \subset \mathbb{R}^d$ be the L2-normalized CLIP embeddings of the N Lo-V frames, and $U_r \in \mathbb{R}^{d\times r}$ the top-r PCA components fitted on them. We project each Hi-I candidate embedding e_j onto the subspace orthogonal to these components and renormalize:

$$\hat{e}_j \;=\; \frac{(\mathbf{I} - U_r U_r^\top)\, e_j}{\big\|(\mathbf{I} - U_r U_r^\top)\, e_j\big\|_2}. \tag{9}$$

The SemDiv kernel in Eq. (3) is then computed on $\{\hat{e}_j\}$ instead of $\{e_j\}$; the quality term and the greedy procedure in Algorithm 1 stay unchanged.

LoHi-SemDiv++ improves slightly over plain SemDiv (Tab. 10), which suggests that modeling Lo-V as an explicit information base is a promising direction. Since the current gain is small, we keep plain SemDiv as the default and leave a deeper study of Lo-V conditioning to future work.

Table 10: LoHi-SemDiv++ on Qwen3-VL-4B (128F@0.25, K=8).

	VideoMME					
Variant	Short	Medium	Long	All	MLVU	LVBench
LoHi-SemDiv (reference)	77.11	65.78	57.56	66.81	74.38	**45.90**
LoHi-SemDiv++ (frame-PCA)	77.22	66.22	57.33	**66.93**	**75.02**	45.77
Δ	+0.11	+0.44	−0.23	+0.12	**+0.64**	−0.13

B Extra Implementation Details

B.1 Hardware and software configuration

All measurements use a single node with four Intel Xeon Gold 6548Y CPUs and 512 GB DDR5 memory, together with one NVIDIA H100 80 GB PCIe GPU. Front-end decoding uses `decord` built against FFmpeg/libavcodec. The GPU NVDEC backend is invoked through the same `decord` interface. VLM inference runs through HuggingFace `transformers` 4.51.3 with PyTorch 2.5 in bf16. Models (Qwen3-VL-4B/8B, VideoLLaMA3-7B, Qwen3.5-4B, Qwen2.5-VL-7B, VideoChat3-4B) are pulled from their official checkpoints and rsynced to local NVMe before benchmarking to remove network jitter. All multiple-choice benchmarks are evaluated with `lmms-eval`[14] using single-turn inference and a maximum of eight generated tokens.

B.2 Front-end decoding latency profiling

We profile decode wall time per video on the full VideoMME set of 900 clips (85.3% at 1280×720 and 13.1% at 640×360, all H.264; median duration about 8.1 min, 95th percentile about 56 min). For each protocol, we record the wall time from constructing the `VideoReader` to the return of the final `get_batch` call, excluding any later preprocessing or transfer to the GPU. When reporting latency at the reference durations of 10, 30, and 60 minutes, we average over clips within ±2 min of each target, so that clips near a duration boundary do not blur the comparison.

CPU vs. GPU NVDEC. We compare uniform sampling with two backends on the same hardware: CPU `decord` and GPU NVDEC. Tab. 11 reports the mean decode time at each frame count. Decode times differ from those used in Tab. 5, possibly due to different CPU/GPU workloads across runs, but the relative savings from 256 to 128 frames remain simila. NVDEC is 1.7× slower than CPU `decord` at N=16 on the full 900-clip set, and 2.3× slower at N=64 on a 45-clip subsample. There are two reasons. First, the default video pipeline resizes and normalizes frames on the CPU (through PIL or torchvision), so decoded frames must travel back across PCIe before reaching the VLM, and the GPU decoding hardware sits idle in the meantime. Second, each PCIe transfer adds roughly 0.1–0.3 s of overhead that a CPU-only path never pays. We therefore report all latency numbers with CPU `decord`, which is also the default backend in common open-source evaluation stacks (`lmms-eval`, the VideoLLaMA3 utilities, and the HuggingFace video processors). Our main observation, that front-end decoding dominates VLM compute on long clips, holds for both backends.

Table 11: Decode wall time on VideoMME with uniform sampling. CPU `decord` is profiled on the full 900-clip set at every frame count; NVDEC is profiled at N=16 on the same set and at N=64 on a 45-clip subsample (†).

Frames	N_{CPU}	CPU mean (s)	N_{GPU}	GPU mean (s)	GPU / CPU
16	900	0.60	900	1.04	1.73×
64	900	1.47	45†	3.38†	-
128	900	2.45	—	—	—
256	900	3.94	—	—	—

Decode at native resolution, resize downstream. Our pipeline decodes every requested frame at native resolution and resizes it afterwards with `smart_resize`; the Lo-V and Hi-I streams share this single decode pass and differ only in the resize branch. This design costs little, because H.264 decode

time depends on the encoded content (the macroblock and motion-vector count at native resolution) rather than on the output resolution. In our profile, IDCT and motion compensation take roughly 70% of the decode wall time, PCIe and memory copies take about 25%, and the resize step takes only about 5%. Two options appear to bypass the full decode but do not. The `width/height` arguments of `decord` still run the full IDCT and skip only the resize step, and the hardware downscaler of NVDEC suffers from the same random-seek stalls as the GPU pipeline above. Either option saves at most about 5% of decode time at 256 frames. The decode workload of LoHi is therefore one shared native-resolution pass that produces the 128 Lo-V frames together with the K Hi-I frames.

B.3 TTFT pipeline profile

Tab. 5 breaks the TTFT into four parts: decode, selector, ViT, and LLM prefilling. All numbers are measured on a dedicated H100 80 GB PCIe node. Each component is the median wall-clock latency over 100 runs after 30 warm-up runs, with `torch.cuda.synchronize()` called between samples. To compare methods at the same token budget, we decode 256 uniformly sampled frames once, store them in a fixed tensor, and reuse this tensor across all runs; the methods then differ only in how they transform these frames (subsampling, CLIP scoring, or token pruning) and in the resulting ViT and LLM prefill shapes. Decode time is not re-measured on this path, since the fixed tensor skips the codec; we instead use CPU `decord` timings measured on the TTFT node (5.4 s for 256 native frames, 3.3 s for 128). These are higher than the full-set profile in Tab. 11 because of different machine load, but the 256-to-128 ratio is similar (1.64× vs. 1.61×). For the keyframe baselines (CLIP-TopK, AKS, BOLT), the selector latency comes from a shared CLIP-encode-and-select pipeline, about 130 ms for 256 frames at 720×1280 with CLIP-B/32 on the H100. SemDiv encodes only the 128 Lo-V candidates, about 65 ms, and its greedy DPP solve adds a negligible amount. ViT and LLM prefill are measured directly on Qwen3-VL-4B with the token shape of each method.

B.4 Codec metadata extraction

We obtain the keyframe index set $\mathcal{I}$ using `decord.VideoReader.get_key_indices()` and use these indices to guide LoHi-Anchor (Eq. (7)). The index lookup requires no additional pixel decoding. Each selected index is mapped to the nearest Lo-V grid frame, so Hi-I frames reuse the decoded candidate pool. VideoReader initialization is included in the reported front-end latency.

C Extra Experiments and Analysis

C.1 LoHi on Qwen3-VL-8B

To check that the cross-resolution decomposition transfers across model scales, we apply LoHi to Qwen3-VL-8B without retuning any hyperparameter. We keep the same three benchmarks and the same 128F Lo-V at r_ℓ=0.25, but reduce the Hi-I budget to K=4, which lowers the total token budget from ∼5,760 to ∼4,320.

All three LoHi variants outperform the prior efficiency baselines on average while using approximately 25% fewer tokens (Tab. 12). LoHi-Anchor reaches 67.70% on VideoMME, and LoHi-SemDiv reaches 75.34% on MLVU and 46.80% on LVBench, all above the strongest prior baseline despite the smaller token budget. We draw two conclusions. The decomposition transfers to the larger backbone. Moreover, once the backbone has accuracy headroom, K becomes one more efficiency lever: it can be reduced to save tokens while the two-stream design still outperforms Low-Res-Base.

C.2 Generalization to Qwen3.5

Qwen3.5 is a recently released native multimodal model. We repeat the two main experiments of Sec. 5 on Qwen3.5-4B, again without retuning any selector hyperparameter, to check that the cross-resolution decomposition depends on the input format rather than on a specific model family.

Setup. By default, the chat template of Qwen3.5 prepends an implicit thinking prefix. We disable thinking and use greedy decoding, matching the inference protocol of Sec. 5.

Table 12: **Cross-backbone transfer of LoHi**. All LoHi rows use a 128F Lo-V at r_ℓ=0.25 paired with K=4 Hi-I frames at r_h=1.0. Avg. is the mean over VideoMME-All, MLVU, and LVBench.

Method	#Frames (Decoded)	Setting $NF@r$	Tokens (Budget)	VideoMME (%) Short	Med.	Long	All	MLVU (%)	LVBench (%)	Avg. (%)
Qwen3-VL-8B										
Vanilla	16	16F@1.0	5,760	69.89	57.56	52.67	60.04	58.95	38.22	52.40
CLIP-TopK	256 → 16	16F@1.0	5,760	68.44	58.44	53.33	60.07	69.73	44.09	57.97
AKS [29]	256 → 16	16F@1.0	5,760	68.56	58.22	53.33	60.04	69.88	44.09	58.00
BOLT [17]	256 → 16	16F@1.0	5,760	72.89	59.67	54.00	62.19	69.34	43.83	58.45
Low-Res-Base	256	256F@0.25	5,760	78.56	64.67	57.89	67.04	71.29	44.80	61.04
LoHi-Uniform	128	128F@0.25+Hi-I	4,320	77.89	66.89	56.00	66.93	72.02	43.32	60.76
LoHi-Anchor	128	128F@0.25+Hi-I	4,320	78.89	66.78	57.44	**67.70**	72.04	44.03	61.26
LoHi-SemDiv	128	128F@0.25+Hi-I	4,320	78.56	65.11	57.67	67.11	**75.34**	**46.80**	**63.08**

The fixed-budget trade-off transfers. At the same token budget, we compare the few-frame, high-resolution setting (16F@1.0) with the many-frame, low-resolution setting (256F@0.25) on the three benchmarks. The dense low-resolution recipe wins on every benchmark (Tab. 13).

Table 13: Frame–resolution trade-off on Qwen3.5-4B at the same token budget.

Setting	VideoMME	MLVU	LVBench
Qwen3.5-4B			
16F@1.0	60.74	60.48	38.54
256F@0.25	**67.41**	**69.87**	**41.45**
Δ	+6.67	+9.39	+2.91

LoHi transfers. We then run LoHi on Qwen3.5-4B with a 128F Lo-V at r_ℓ=0.25 and K=8 Hi-I frames, and compare the three selectors (Tab. 14). SemDiv is the best selector on all three benchmarks.It also improves over the 256F@0.25 Low-Res-Base on every benchmark at the same token budget, with the largest gain on LVBench (+5.23%), whose long clips leave the most room for diversity-aware selection. SemDiv also Both findings therefore carry over to this newer natively multimodal backbone without any modification.

Table 14: LoHi selectors on Qwen3.5-4B (128F Lo-V at r_ℓ=0.25, K=8 Hi-I).

Configuration	VideoMME	MLVU	LVBench
Qwen3.5-4B			
128F@0.25	64.93	66.80	39.51
256F@0.25	67.41	69.87	41.45
LoHi-Uniform	67.52	67.48	44.61
LoHi-Anchor	67.70	66.70	44.74
LoHi-SemDiv	**68.15**	**70.42**	**46.68**

C.3 Generalization to Additional VLM Backbones

To further evaluate the generality of LoHi, we extend our experiments to three additional VLMs: Qwen2.5-VL-7B[2], VideoChat3-4B[15], and VideoLLaMA3-7B[41].

Across these backbones, the results further support our three lessons. L1: the dense low-resolution base consistently outperforms the 16-frame vanilla setting at the same token budget. L2: LoHi-SemDiv improves over the dense base in seven of nine model–benchmark combinations, with small decreases in the remaining two (-0.26% and -0.13%). L3: LoHi requires only 128 sampled frames to be decoded instead of 256, reducing the front-end decoding workload.

Table 15: Generalization to additional VLM backbones. Both LoHi variants use 128 Lo-V frames at $r_\ell = 0.25$ and Hi-I frames at $r_h = 1.0$. †For VideoLLaMA3, MLVU scores use the output normalization described in Sec. C.7.

Method	VideoMME	MLVU	LVBench
Qwen2.5-VL-7B			
Vanilla 16F@1.0	58.78	58.00	37.83
Low-Res-Base 256F@0.25	63.78	64.77	41.06
LoHi-Uniform K=8	63.26	63.80	40.35
LoHi-SemDiv K=8	**64.41**	**68.03**	**43.51**
VideoChat3-4B			
Vanilla 16F@1.0	59.78	63.43	40.09
Low-Res-Base 256F@0.25	64.81	70.29	**46.29**
LoHi-Uniform K=8	63.56	69.18	42.80
LoHi-SemDiv K=8	**64.96**	**73.00**	46.03
VideoLLaMA3-7B†			
Vanilla 16F@1.0	46.81	52.38	32.09
Low-Res-Base 256F@0.25	50.04	57.73	36.99
LoHi-Uniform K=8	**51.37**	58.14	35.57
LoHi-SemDiv K=8	50.67	**58.59**	36.86

C.4 Pushing the Lo-V to N=256

The main paper fixes the Lo-V at N=128 so that all methods compare at the same token budget. Here we relax this constraint: we keep r_ℓ=0.25 and r_h=1.0, increase the Lo-V to N=256, and add K=4 Hi-I frames. Tab. 16 reports VideoMME results on Qwen3-VL-4B. The results extend both lessons. Densifying the Lo-V improves accuracy by supplying richer temporal coverage (L1), and adding Hi-I frames on top still brings a clear gain, since the high-resolution evidence they supply (L2) is not something more low-resolution frames can provide. The two streams therefore remain complementary as the Lo-V grows denser.

Table 16: Pushing the Lo-V to N=256 on VideoMME (Qwen3-VL-4B). All rows use r_ℓ=0.25 for Lo-V and r_h=1.0 for Hi-I.

Configuration	Lo-V	Hi-I	Short	Medium	Long	All
Qwen3-VL-4B						
256F@0.25	256F	—	75.78	62.33	55.22	64.44
LoHi-Uniform	128F	K=8	75.67	66.44	55.22	65.78
LoHi-SemDiv	128F	K=8	77.11	65.78	57.56	**66.81**
LoHi-Uniform	256F	K=4	77.67	65.33	56.78	66.59
LoHi-Anchor	256F	K=4	77.00	65.89	57.78	66.89
LoHi-SemDiv	256F	K=4	76.78	66.44	58.11	**67.11**

C.5 Evaluation under Larger Token Budgets

To cover a substantially wider budget range, we evaluate Qwen3.5-27B on EgoLongQA, the long-video QA task of EgoWearBench [30]. We use 700 egocentric clips of approximately 10 minutes each. To examine the token allocation motivated by L1 and L2 in this regime, we relax the decoded-frame constraint and use 512 sampled frames for all configurations. For LoHi, we fix the Lo-V resolution at $r = 0.25$ and increase the Hi-I budget from $K = 0$ to $K = 128$, scaling the visual-token count from approximately 25K to 117K. We use SemDiv for Hi-I selection. We also compare against a single-stream baseline at $r = 0.5$. Results are reported in Tab. 17.

(i) Adding more Hi-I frames consistently helps. Accuracy improves monotonically from 82.14% to 87.57% as K increases from 0 to 128. **(ii) At matched accuracy, LoHi uses less visual computation.** 512F@0.25 with $K = 64$ attains the same accuracy as 512F@0.5 (85.86%) while using **29% fewer**

Table 17: Evaluation under larger visual-token budgets on EgoLongQA with Qwen3.5-27B. All configurations use 512 sampled frames.

Lo-V + Hi-I	Visual tokens	Acc. (%)
Qwen3.5-27B		
512F@0.25, $K = 0$	25,088	82.14
512F@0.25, $K = 8$	30,848	82.86
512F@0.25, $K = 16$	36,608	83.71
512F@0.25, $K = 32$	48,128	85.00
512F@0.25, $K = 64$	**71,168**	**85.86**
512F@0.25, $K = 128$	117,248	**87.57**
512F@0.5, $K = 0$	100,352	**85.86**

visual tokens (71K vs. 100K) and **29% fewer ViT patches** (285K vs. 401K). These results further support our view of long-video efficiency as a joint allocation problem.

C.6 Adaptive cost control via multi-turn triggering

A benefit of the decoupled two-stream design is that it naturally supports multi-turn inference: the Lo-V base can answer first, and the Hi-I frames can be added in a second pass only when needed. Since the Lo-V base alone already answers most questions correctly (Tab. 18), this offers a chance to reduce vision tokens and prefill latency further. We explore this with the entropy-triggered mechanism in Eq. (6): a question enters the second pass only when the entropy of its Lo-V answer distribution exceeds a threshold. We evaluate on VideoMME with Qwen3-VL-4B and a 128F Lo-V, and tune the threshold on a held-out 10% split. As Tab. 18 shows, only about a third of the questions enter the second pass, while the accuracy nearly matches always running Hi-I. The two-stream design thus supports adaptive inference well. The threshold is tuned per benchmark, so we regard this as an exploratory validation; threshold selection without benchmark-specific tuning is left to future work.

Table 18: Two-pass triggers on VideoMME (Qwen3-VL-4B, 128F Lo-V).

Trigger	2nd-pass rate	Acc. (%)	Δ vs. Lo
Lo-V only	0%	63.00	0.00
Entropy (θ tuned)	37%	65.71	+2.71
Always-Hi	100%	65.78	+2.78

C.7 Answer-Format Normalization for VideoLLaMA3 on MLVU

This is a formatting issue in answer extraction, not an accuracy effect. MLVU expects answers ending in ‘‘`A)`’’. VideoLLaMA3-7B instead writes ‘‘`A.`’’ in about 30–38% of its outputs, and the MLVU extractor then falls back to a literal string comparison that fails to match the gold letter. We fix this by stripping the trailing period whenever the prediction is exactly two characters of the form ‘‘`X.`’’ with $X \in \{\text{A,B,C,D}\}$. Every output without a closing parenthesis in the three runs has exactly this form, so the rule covers all such cases. VideoMME and LVBench use a generic `[ABCD]` regex and are not affected. The normalization raises each VideoLLaMA3 MLVU number by 16–17% and does not change the trend across frame counts (Tab. 19). In Tab. 2, the MLVU scores for VideoLLaMA3 are reported before answer-format normalization and marked with †.

Table 19: VideoLLaMA3-7B on the MLVU dev split, with and without output normalization. “No-paren rate” is the fraction of outputs lacking a closing parenthesis.

Configuration	Raw (%)	Normalized (%)	No-paren rate
16F@1.0	35.88	52.38	38.5%
64F@0.5	38.97	55.27	30.9%
256F@0.25	40.30	57.73	30.8%
Δ (256F − 16F)	+4.42	+5.35	—

C.8 The inter-frame token compressor in VideoLLaMA3 removes almost no tokens

VideoLLaMA3-7B compresses visual tokens with a module called DiffFP. For each patch, DiffFP computes the pixel difference between adjacent frames and drops the corresponding token when this difference falls below a threshold. To check how much the module actually removes, we measure its keep ratio, defined as the number of tokens after DiffFP divided by the number before. We evaluate 45 configurations in total, covering five frame counts (16 to 256), three resolution scales (1.0, 0.5, and 0.25), and three benchmarks, with 60 videos sampled per configuration.

The result is consistent. The mean keep ratio of every configuration stays above 0.959 (Tab. 20), so on average DiffFP removes at most $\sim$4% of tokens in every long-video setting we test. Individual videos can go lower (down to 0.639 on MLVU), but such cases are rare. The cause is saturation. When many frames are sampled at reduced resolution, each merged patch covers a large image region, and almost any motion between frames pushes its pixel difference above the threshold, so the mask keeps nearly every token. For all token-budget calculations in this paper, we therefore treat VideoLLaMA3 as having no inter-frame compression.

Table 20: Keep ratio of DiffFP in VideoLLaMA3-7B, 45 configurations, 60 videos each. The lowest mean is 0.959 (VideoMME, 256F@1.0).

Benchmark	Mean (range)	Std (max)	Min (per video)
LVBench	0.990 – 0.998	0.032	0.806
MLVU	0.977 – 0.999	0.069	0.639
VideoMME	0.959 – 0.991	0.064	0.674

C.9 Extra ablations

C.9.1 Prompt template: appended cue vs. interleaved cue

The Lo-V and Hi-I streams are joined by a natural-language cue (Sec. 4.2). We compare two placements of the cue:

Appended cue. `[Lo-V tokens] [Hi-I tokens] "You are watching a video. The video above provides low-resolution temporal context. The` K `images above are high-resolution views of selected key frames. {question}"`

Interleaved cue. `[Lo-V tokens] "The above low-resolution video provides temporal context. The following` K `high-resolution images show selected key frames:" [Hi-I tokens] "{question}"`

In the Appended variant, the model reads the description only *after* it has seen the Hi-I tokens. In the Interleaved variant, the description sits *between* the Lo-V block and the Hi-I block, so the model is told that the following tokens are high-resolution frames before attending to them. Tab. 21 reports the comparison; all other settings match Tab. 7.

We find two things. First, LoHi is robust to the prompt wording. Every selector at every K benefits from the cross-resolution decomposition under either prompt, and the gap between the two prompts never exceeds 0.8%. Second, a prompt that separates the two modalities more clearly helps further. The Interleaved cue adds +0.44 to +0.74% for Uniform and SemDiv. This suggests that explicitly linking Hi-I images to the video timeline can help. Anchor does not show the same benefit in this ablation. We use the Interleaved cue by default. Richer delimiters, such as per-frame timestamps or role tokens, may extend this trend further.

Table 21: Prompt template ablation on VideoMME (Qwen3-VL-4B, 128F Lo-V at r=0.25).

Selector	K=4 (%)		K=8 (%)	
	Appended	Interleaved	Appended	Interleaved
LoHi-Uniform	64.78	**65.52** (+0.74)	65.30	**65.78** (+0.48)
LoHi-SemDiv	65.70	**66.22** (+0.52)	66.37	**66.81** (+0.44)
LoHi-Anchor	65.96	65.85 (−0.11)	65.85	65.89 (+0.04)

C.9.2 SemDiv sharpening coefficient α

The quality term in the SemDiv kernel (Eq. (3)) sharpens the normalized query–frame similarity with an exponent α. We sweep $\alpha \in \{1, 5, 10\}$ on VideoMME (Qwen3-VL-4B, 128F Lo-V at r=0.25, K=8, interleaved cue) and obtain 66.67%, **66.81**%, and 66.48%. Accuracy ranges from 66.48% to 66.81%, suggesting limited sensitivity to α over the tested range. We use $\alpha = 5$ in all LoHi-SemDiv runs without per-task tuning.

C.10 Further Analysis of Hi-I Selection

All analyses use Qwen3-VL-4B. We conduct exact McNemar tests and video-level paired bootstrap with 50,000 resamples. The two procedures yield consistent significance conclusions, and the confidence intervals are stable across bootstrap seeds.

Statistical significance. As shown in Tab. 22, SemDiv significantly improves over Uniform on MLVU (+6.76%, $p < 10^{-16}$), while the differences on VideoMME and LVBench are not statistically significant.

Table 22: Paired comparison of LoHi-SemDiv and LoHi-Uniform. Confidence intervals are computed using video-level paired bootstrap. W/L denotes questions answered correctly only by SemDiv/Uniform, respectively.

SemDiv − Uniform	Δ (%)	95% CI	McNemar p	W/L
VideoMME	+1.04	[−0.15, +2.22]	0.094	144/116
MLVU	**+6.76**	[+5.17, +8.41]	$\mathbf{< 10^{-16}}$	233/86
LVBench	+1.81	[−0.13, +3.76]	0.063	120/92

Benchmark dependence. The same benchmark-dependent pattern appears in CLIP-TopK and AKS: gains over uniform selection are largest on MLVU and small or negative on VideoMME (Tab. 23). Each comparison uses a matched budget: 16 frames for the keyframe baselines and $K = 8$ Hi-I frames for LoHi. The gain magnitudes are not directly comparable, since keyframe methods select the sole visual input, whereas SemDiv selects additional detail on top of Lo-V. The shared pattern shows that benchmark dependence is not unique to SemDiv.

Table 23: Accuracy gains of query-aware selection over the corresponding uniform baselines at matched budgets.

Query-aware − uniform selection (%)	VideoMME	MLVU	LVBench
CLIP-TopK − Vanilla (16F@1.0)	−0.6	+10.5	+6.1
AKS − Vanilla (16F@1.0)	−0.5	+10.3	+6.1
LoHi-SemDiv − LoHi-Uniform	+1.0	+6.8	+1.8

Task-level benefits. Repetitive action counting requires modeling temporal structure across action instances [12].MLVU action counting provides a clear example (Tab. 24): Uniform falls below the Lo-V-only baseline (29.13% vs. 41.75%), whereas SemDiv reaches 53.40%, improving over Uniform by 24.27% (95% CI: [+16.5, +32.0]). This highlights the importance of Hi-I placement for questions requiring evidence from multiple action instances. The MLVU gain is not driven by this category alone: leave-one-category-out differences range from +4.93 to +8.06%, all with $p < 10^{-9}$.

Table 24: Effect of Hi-I selection on MLVU action counting with Qwen3-VL-4B ($n = 206$).

MLVU action-count	Lo-V only	LoHi-Uniform	LoHi-SemDiv
Accuracy (%)	41.75	29.13	**53.40**

Variation across query types. The qualitative examples in Figs. 5 and 7 illustrate two different needs: resolving fine spatial details and placing high-resolution frames at the right moment. We further examine this distinction through a query-type breakdown. Within MLVU's ego category, LoHi-Uniform matches Low-Res-Base at 59.66%, while LoHi-SemDiv reaches 69.03%. Uniform and SemDiv use the same Lo-V stream and the same Hi-I budget ($K = 8$), differing only in which frames are selected. To examine how this benefit varies with the query, we split the 352 ego questions by template. Tab. 25 reports the five templates with $n \geq 20$; the remaining 78 questions belong to templates with at most 17 questions each. Complete results and evaluation logs for all datasets are available in our GitHub repository, linked from the project page[1].

Table 25: Accuracy (%) by question template within MLVU's ego category, using Qwen3-VL-4B. Base denotes Low-Res-Base; Uniform and SemDiv use $K = 8$ Hi-I frames.

Query type	n	Base	Uniform	SemDiv
Static location (*where was X*)	111	68.47	70.27	**73.87**
Transient action (*put in*)	48	64.58	66.67	**83.33**
Action destination	45	71.11	75.56	**77.78**
Yes/no (*did I . . .*)	43	58.14	48.84	**65.12**
Static entity count	27	18.52	**37.04**	**37.04**

Static entity counting improves from 18.52% to 37.04% with either selector, whereas transient-action accuracy increases from 66.67% with Uniform to 83.33% with SemDiv. These results suggest that high-resolution evidence alone can help some queries, while others benefit substantially from query-aware placement. We leave query-aware adaptive frame–resolution allocation to future work.

Selection overhead. SemDiv scores only the 128 Lo-V candidates, adding approximately 70 ms over Uniform (1.8% of TTFT). Its total selector-stage latency is 120 ms, below the 130 ms of the keyframe-selection baselines (Tab. 5).

Performance across benchmarks and task categories. SemDiv is never significantly worse than Uniform on any evaluated benchmark or task category; the decrease on topic reasoning (-2.66%) is not significant. A stratified exact test across all three benchmarks confirms an overall advantage ($p = 5 \times 10^{-13}$). These results and the small additional overhead motivate SemDiv as our default, while Uniform remains an option without semantic-selection overhead.

C.11 Evaluation with Subtitles

Motivation for the subtitle-free protocol. Our main evaluation excludes subtitles to isolate the effect of visual input allocation. To examine how much information subtitles provide, we evaluate Qwen3-VL-4B using only the transcript and question, without visual input. Tab. 26 reports results on VideoMME questions with and without available subtitles; the visual configurations do not use subtitles. With zero visual input, the transcript alone reaches 61.56% on subtitled questions, close to the 256-frame visual pipeline (63.13%). Subtitles also consume a substantial context budget. With the Qwen3-VL tokenizer, VideoMME subtitle lengths are heavily right-skewed (Tab. 27).

Table 26: Accuracy (%) of visual-only and text-only inputs on VideoMME using Qwen3-VL-4B. Subsets are defined by subtitle availability. Visual inputs exclude subtitles; text-only inputs contain the transcript and question.

Qwen3-VL-4B	Subtitled (2,232 Qs)	No subtitle (468 Qs)
16F@1.0	56.18	65.38
256F@0.25	63.13	70.73
Text-only	**61.56**	–
Question-only	–	30.56

[1] https://sixundong.com/projects/lohi

Table 27: VideoMME subtitle lengths measured with the Qwen3-VL tokenizer, relative to the visual-token budget of 5,760 tokens.

Subtitle length	Tokens	% of visual budget
Median	1,539	27%
Mean	3,018	52%
P90	7,799	**135%**

Table 28: Accuracy (%) with and without subtitles on the 2,232 subtitle-available questions in VideoMME, using Qwen3-VL-4B. Reported p-values compare LoHi-SemDiv with Low-Res-Base in each setting.

Subtitled subset	w/o subs	w/ subs	Subtitle gain (%)
Low-Res-Base 256F@0.25	63.13	73.12	+9.99
LoHi-SemDiv $K = 8$	65.28	72.13	+6.85
Gap (LoHi − Base)	+2.15	−0.99	–
p-value	0.011	0.117	–

At the 90th percentile, the transcript alone exceeds our visual-token budget of 5,760 tokens, bringing the combined visual and subtitle input to more than 13,000 tokens. Including subtitles therefore introduces both a substantial source of non-visual information and additional context cost.

Performance with subtitles. We evaluate Low-Res-Base and LoHi-SemDiv with and without subtitles on the same subtitled subset ($n = 2{,}232$) in Tab. 28.

Without subtitles, LoHi-SemDiv is significantly more accurate than the dense Low-Res-Base on this subset ($+2.15\%$, $p = 0.011$) while using half as many sampled frames. Adding subtitles improves both methods, but the gain is larger for Low-Res-Base ($+9.99$ vs. $+6.85\%$), and the resulting difference between the methods is not statistically significant (-0.99%, $p = 0.117$). Together with the text-only result, this suggests that subtitles can mask the benefit of improved visual input allocation, motivating our subtitle-free main evaluation. With subtitles, LoHi still uses only 128 sampled frames instead of 256, although its accuracy advantage is no longer observed.

D Extended Related Work

D.1 Codec-Aware and System-Level Video Processing

One line of work reduces front-end cost by operating directly on codec-level signals, such as the motion vectors and residuals stored in the compressed bitstream, instead of first decoding to RGB [22, 48]. The idea originates in compressed-domain action recognition [37], which uses motion vectors as task features without full decoding. The most recent VLM-side instance, CoPE-VideoLM [22], learns codec-aware token representations and removes the RGB-decode bottleneck end to end. System-level work such as QuickVideo [23] targets the same problem from another angle, optimizing decoder and prefill throughput at runtime.

These approaches share two properties. They need substantial model-side or system-side training and engineering to make compressed-domain inputs work with downstream VLMs, and their savings come from reducing the cost of decoding each frame. LoHi takes a training-free view one level higher. Instead of redesigning how a single frame is decoded, we observe that long videos need not be decoded densely in the first place: uniformly sampling a fixed budget of N frames already preserves temporal coverage, at a front-end cost that does not grow with duration (L3). Uniform sampling also produces evenly spaced timestamps, which unified-stream VLMs need for multimodal RoPE-based temporal modeling. LoHi is therefore complementary to codec-aware tokenizers and runtime systems rather than a replacement. A future system could combine codec-aware decoding with our cross-resolution decomposition to compound the gains.

D.2 Anyres and Slow–Fast: Two Single-Axis Decompositions

Two paradigms decompose the visual budget along a single axis: anyres tiling along space, and Slow–Fast along time. LoHi differs from both, for opposite reasons: anyres is its enabling foundation, while Slow–Fast conflicts with how modern unified-stream VLMs encode time.

Anyres with token merging makes Lo-V viable. Anyres-style methods such as LLaVA-NeXT [16] and InternLM-XComposer2-4KHD [7] split a single high-resolution image into spatial patches, encode the patches independently, and concatenate the results. Modern frontier VLMs (Qwen2.5/3-VL [1], InternVL3.5 [34]) inherit this design and add a vision-side merge layer that aggregates adjacent patches into fewer tokens. As a result, at $r{=}0.25$ on a 720p source, a Qwen3-VL frame occupies only about 20–30 merged tokens (Tab. 2 realizes $\sim$5,760 tokens at 256 frames). This is exactly the regime in which L1 and L2 operate: the experiments in Secs. 3.1 and 3.2 confirm that moderate downscaling preserves most accuracy, and that the loss concentrates on a small set of resolution-sensitive tasks, which a few high-resolution frames can recover. Tiling is therefore not a competitor to LoHi but the property that makes the low-resolution side of our decomposition cheap and accurate. The two also differ in scope: tiling decomposes within a single static image to improve spatial coverage, while LoHi composes the video and image modalities in one forward pass to balance temporal density and spatial detail under the decode and token constraints of long clips.

Slow–Fast and per-frame variable token allocation. The Slow–Fast paradigm [9] processes video with two pathways at different temporal and spatial rates, originally for action recognition. Video VLMs have imported the same pattern. LLaVA-Video [45] allocates different token counts to different frames, and query-aware frame selectors such as F2C [28] and Q-frame [43] follow the same spirit: keyframes form a slow path with a larger per-frame budget, and the remaining frames form a fast path with a smaller one. The intent of spending budget where motion or query relevance concentrates is reasonable. The realization, however, conflicts with how unified-stream VLMs encode time. Heterogeneous token counts across frames disrupt the multimodal RoPE position assignment used for temporal reasoning, and recovering this typically requires dedicated training, which rules out plug-and-play deployment on existing backbones.

LoHi keeps the temporal stream uniform and routes detail through the image pathway. LoHi keeps a uniform per-frame token allocation in the Lo-V stream, so mRoPE stays consistent across all N frames, and routes additional spatial detail through the image pathway (Hi-I) instead of along the temporal axis. This separation matters because the image pathway is where modern VLMs concentrate their fine-detail capability, such as small-object and text recognition, as seen in DeepSeek-OCR [35] and the OCR strength of Qwen3-VL [1]. Allocating the high-resolution budget there reuses this learned capability, which is what allows LoHi to be training-free: Lo-V matches the video-side training distribution (a uniform low-resolution timeline), and Hi-I matches the image-side training distribution (high-resolution spatial detail).

E Limitations and Outlook

Limitations. We validate LoHi on frontier VLMs (Qwen3-VL-4B/8B, VideoLLaMA3-7B, Qwen3.5-4B, Qwen2.5-VL-7B and VideoChat3-4B) whose dynamic-resolution architectures, such as AnyRes, stay accurate under aggressive downscaling. Older or smaller models such as LLaVA-NeXT [16] may degrade more sharply at very low resolutions, and would need a higher base resolution r_ℓ or a larger Hi-I budget K to compensate. Our findings therefore apply most directly to modern and future architectures, where low-resolution inputs are highly token-efficient. Our training-free integration also relies on a textual cue to bridge the Lo-V and Hi-I streams, introducing mild sensitivity to prompt structure, as shown in the prompt ablation Sec. C.9.1. In addition, this work focuses on front-end and prefill latency, while the decoding latency of the VLM also matters [5]. At a matched visual-token budget, LoHi reduces front-end cost without increasing the visual context length used during generation. When fewer visual tokens are desired to reduce prefill and generation costs, the lower-budget results in Tab. 6 also support resolution-based allocation. Beyond this controlled setting, agentic keyframe selection can incur additional generation overhead by producing many reasoning tokens to decide which tools and frames to use. Existing studies rarely count this thinking time. We leave its evaluation to future work.

Outlook. The decoupled two-stream design opens three directions. First, combining with prior efficiency methods. Once the input is split into Lo-V and Hi-I streams, existing keyframe-selection and token-pruning algorithms are no longer competitors; they can serve as drop-in modules that further compress the Hi-I subset. Second, broader adaptive computation. Our entropy-triggered early exit (Sec. C.6) targets multiple-choice QA, but the same Lo-V-first, Hi-I-on-demand pattern extends naturally to open-ended tasks such as video grounding, summarization, and dialogue. High-resolution inspection could further serve as an on-demand tool for multimodal agents [31, 40]. Third, reusing intermediate frames. As L3 shows, decoding a target P-frame forces the decoder to also process the preceding I-frame and P-frames in the same GoP. These frames come for free, and reusing them as extra Hi-I candidates or as motion-magnitude priors could enrich Hi-I selection without increasing the decode budget N_{dec}.

F Qualitative Results

We provide additional qualitative examples to illustrate the roles of the Hi-I pathway and the two selectors.

In Fig. 5, the Lo-V-only prediction is wrong because the answer depends on a small detail on the interviewee's chin, which is blurred at low resolution. The selected Hi-I frames restore this detail and lead to the correct answer. The Hi-I pathway is thus most useful when the question needs fine spatial evidence.

Fig. 6 and Fig. 7 compare LoHi-Uniform against LoHi-Anchor and LoHi-SemDiv, respectively. In Fig. 6, I-frame-guided selection lets LoHi-Anchor replace a weakly informative frame with the scoreboard that answers the question. In Fig. 7, LoHi-SemDiv helps in a different way: it removes redundant selections and picks a set of complementary, action-centric frames, which makes the key evidence easier to find.

These examples illustrate two ways uniform Hi-I selection can miss useful evidence: selecting an uninformative frame or missing a brief, task-relevant event. The Lo-V stream preserves low-resolution coverage at all sampled timestamps, but recovering fine details still depends on selecting informative Hi-I frames.

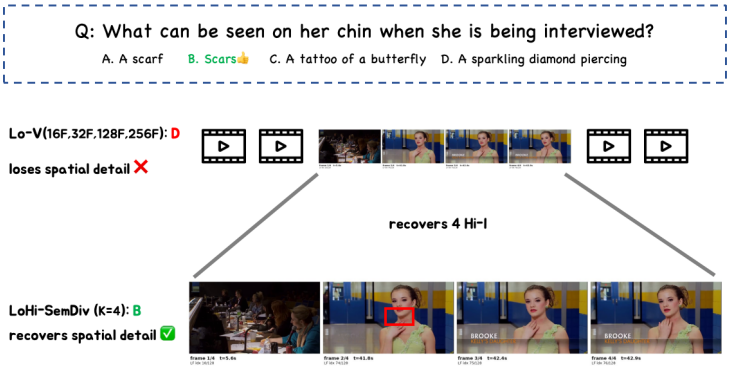


Figure 5: Recovered Hi-I frames restore the fine-grained detail needed for the question.



Figure 6: LoHi-Anchor selects more informative Hi-I frames than uniform sampling.

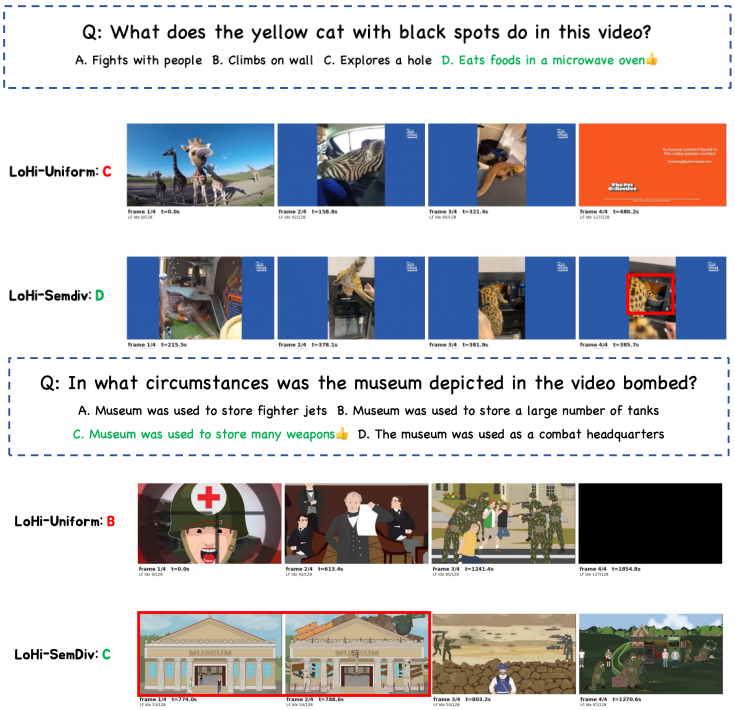


Figure 7: LoHi-SemDiv improves Hi-I selection by favoring semantically complementary action frames.

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 main claims made in the abstract and introduction accurately reflect the paper's contributions and scope, and are supported by our comprehensive experiments Sec. 5.

   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: We discussed the limitations of our work in Section E.

   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: [Yes]

   Justification: We provided the theoretical proofs in the main paper.

   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: Sec. 5 specifies benchmarks, backbones, and other details; the algorithm specified by Eq. (7) (Anchor) and Algorithm 1 (SemDiv) in App. A; hardware, software versions, and decoding protocols are listed in App. B.1, B.2, and B.3.

   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: [Yes]

   Justification: The benchmarks and backbones are publicly released; our implementation is built on the open-source `lmms-eval` framework and the selector code, configs, and run scripts have been released. Hardware, software versions, and protocol details for the reported numbers are in App. B.

   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: We provided all the necessary implementation details in Sec. 5 and App. B.

   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: [Yes]

   Justification: We report 95% confidence intervals and p-values based on video-level paired bootstrap (50,000 resamples) and McNemar tests to rigorously evaluate statistical significance.

   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: We report the computer resources in App. B.

   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 work uses publicly released benchmarks and pretrained models for evaluation only, involves no human subjects or new data collection, and complies with the NeurIPS Code of Ethics.

   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 an inference-efficiency study for existing long-video VLMs; it introduces no new model or dataset, and any downstream impact follows directly from that of the underlying VLMs and benchmarks. We discuss no novel application path beyond efficiency improvements for existing tasks.

    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 releases no pretrained models, generative models, or scraped datasets. The only released artifact is the training-free LoHi framework code, which adds no capability beyond that of the open-source VLMs it is built on top of and therefore introduces no additional misuse risk.

    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: All third-party assets are cited at first use—benchmarks (VideoMME, MLVU, LVBench), backbones (Qwen3-VL, VideoLLaMA3), baselines (AKS, BOLT, FlashVid, VisionZip, MMTok), and the `lmms-eval` framework—and used in accordance with their respective licenses (e.g., Apache-2.0 for `lmms-eval` and Qwen3-VL); details are in App. B.

    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: [Yes]

    Justification: The paper introduces the LoHi framework, which will be released upon acceptance as a plug-in module on top of `lmms-eval`, including the Lo-V/Hi-I composition pipeline, the LoHi-Anchor and LoHi-SemDiv selectors, configuration files for all experiments in Sec. 5, and a README with installation and usage instructions for reproducing the main results. The code will be released under an open-source license. No new datasets or pretrained models are introduced.

    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 involves no crowdsourcing and no research with human subjects; all experiments are conducted on publicly released video 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 human subjects, so IRB approval 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: [N/A]

    Justification: LLMs were not used as a component of the core methodology, experimental design, or analysis. Any use was limited to prose polishing, which per the question description does not require declaration.

    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.