FORCING-KV: HYBRID KV CACHE COMPRESSION FOR EFFICIENT AUTOREGRESSIVE VIDEO DIFFUSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Autoregressive (AR) video diffusion models adopt a streaming generation framework, enabling long-horizon video generation with real-time responsiveness, as exemplified by the *Self Forcing* training paradigm. However, existing autoregressive video diffusion models still incur substantial attention and memory overhead because they retain largely redundant key–value (KV) caches for historical frames, limiting their scalability. In this paper, we tackle this challenge by introducing KV cache compression into autoregressive video diffusion. We observe that attention heads in mainstream AR diffusion models exhibit markedly distinct attention patterns and functional roles that remain stable across samples and denoising steps. Building on our empirical study of head-wise functional specialization, we divide the attention heads into two categories: *static heads*, which focus on transitions across autoregressive chunks and intra-frame fidelity, and *dynamic heads*, which govern inter-frame motion and consistency. We then propose FORCING-KV, a hybrid KV cache compression strategy that performs structural static pruning for static heads and dynamic pruning for dynamic heads based on segment-wise similarity. While maintaining output quality, FORCING-KV compresses **53%-73%** of the KV cache, achieves **29** FPS on a single NVIDIA H200 GPU with **30%** peak cache memory reduction, and delivers up to **1.35**× and **1.50**× speedups on LongLive and Self Forcing at 480P resolution. The efficiency gains further scale with resolution and sliding window size. Code and demo videos are provided in the supplementary material.

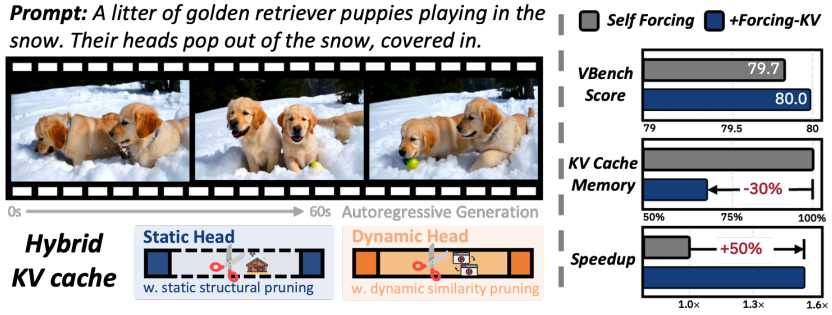

Figure 1: Overview of FORCING-KV. We apply static structural pruning and dynamic similarity pruning to different heads, accelerating inference, reducing cache memory while preserving quality.

1 INTRODUCTION

Autoregressive (AR) video diffusion (Huang et al., 2025a; Yin et al., 2025b; Teng et al., 2025; Chen et al., 2025; Zhang et al., 2025b; Yang et al., 2026; Gao et al., 2026) has recently emerged as a compelling paradigm for efficient, streaming text-to-video generation. Unlike conventional bidirectional video diffusion models that denoise all frames simultaneously (Peebles & Xie, 2023; Lin et al., 2024; Kong et al., 2025; Wan et al., 2025), AR video diffusion models synthesis video chunk by chunk, with each denoising chunk conditioned on previously generated video content via a key-value (KV) cache. This paradigm enables long-horizon, variable-length video generation with interactive

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

inputs, while reducing both attention complexity and the latency to the first generated content. Mainstream approaches build upon the *Self Forcing* (Huang et al., 2025a; Cui et al., 2026a) training paradigm, performing self-rollout during training to mitigate error accumulation, as exemplified by the broader family of "forcing" methods (Huang et al., 2025a; Yi et al., 2025; Yesiltepe et al., 2025; Xiao et al., 2025; Huang et al., 2025b; Li et al., 2026; Cui et al., 2026b; Yang et al., 2026; Cui et al., 2026a; Lu et al., 2026; Liu et al., 2026) that have shown strong long-horizon visual quality.

However, existing mainstream AR video diffusion models still suffer from substantial attention complexity and memory overhead due to the largely redundant KV cache (Yang et al., 2026; Chen et al., 2026a). The denoising chunk underutilizes the video cache across historical frames, causing unnecessary computation and memory footprint especially for long-horizon and high-resolution video generation. For instance, generating a 30-second video with Self Forcing at 1080P resolution takes over 2 minutes on a NVIDIA H200 GPU, corresponding to a generation speed of 1.71 FPS in the diffusion transformer (DiT), with over 60 GB GPU memory occupied by the KV cache. This poses a major obstacle to real-time deployment and to memory-constrained scenarios.

To achieve real-time inference, studies have explored sparse attention (Lv et al., 2026; Agarwal et al., 2026) and feature caching (Ma et al., 2026; Samuel et al., 2026) for AR video diffusion models. Although effective, such methods neither reduce memory consumption nor compress the KV cache, which is a distinctive component of AR video models. Recently, Dummy Forcing (Guo et al., 2026) observes that certain heads in AR diffusion models concentrate primarily on the local chunk, and accordingly discards the historical context for compression. However, it lacks a detailed analysis of the functional heterogeneity across attention heads, and its aggressive compression results in degraded temporal dynamics and discontinuity across chunks (i.e., flickering and broken transitions at chunk boundaries), as shown in Section 4 [1]. We posit that for AR video diffusion models, effective context utilization is a key problem to both quality and efficiency. This raises a pivotal question:

How do video diffusion models leverage their KV cache for autoregressive generation?

Our findings reveal a heterogeneous behavior. We observe markedly distinct attention patterns and functional roles across the attention heads of AR diffusion models. Through a series of empirical ablation studies in Section 2, we categorize them into two categories. *Static heads* densely attend to the current chunk and the most recent frame, which we denote as the *transition anchor frame*, to preserve intra-frame fidelity and visual continuity across AR chunks. *Dynamic heads* sparsely capture the inter-frame correspondences across the same spatial regions, governing subject consistency and motion dynamics. Moreover, we find that this head division remains stable across samples and denoising steps, and generalizes broadly across multiple AR diffusion models. Based on these observations, we propose FORCING-KV, a hybrid KV cache compression method for AR video diffusion models that decouples static structural attention from dynamic context utilization.

FORCING-KV first introduces a one-shot, model-level **offline head profiling** procedure (see Section 3.1) that identifies static and dynamic heads based on frame-wise attention mass. Subsequently, we apply a hybrid KV cache compression strategy. For static heads, we adopt **static structural pruning** (see Section 3.2) to consistently preserve the transition anchor frame and prune distant frames. For dynamic heads, we employ **dynamic similarity pruning** (see Section 3.3), which computes segment-wise similarity between adjacent frames in the KV cache to retain temporally evolving content while pruning redundant and unchanged content. To summarize, our main contributions are:

(1) *Novel Pattern Discovery*: We uncover a universal head specialization pattern shared by mainstream autoregressive video diffusion models: transitions across autoregressive chunks are mediated by static heads that concentrate on the transition anchor, whereas long-horizon consistency and dynamics are sustained by dynamic heads through sparse inter-frame attention.

(2) *Hybrid KV cache Compression*: Building upon this, we propose FORCING-KV, a compression strategy that preserves structurally critical content for static heads while applying dynamic similarity pruning for dynamic heads, decoupling static attention from dynamic context utilization.

(3) *Extensive Experiments*: Evaluations across models, benchmarks and video durations show that FORCING-KV is both high-fidelity and efficient: It compresses **53%-73%** of the KV cache and achieves **1.35**× and **1.50**× speedups with **30%** peak cache memory reduction on LongLive and Self Forcing at 480P resolution. The efficiency gain further scales to **2.82**× at 1080P.

[1] We provide a detailed related work discussion in Appendix A.

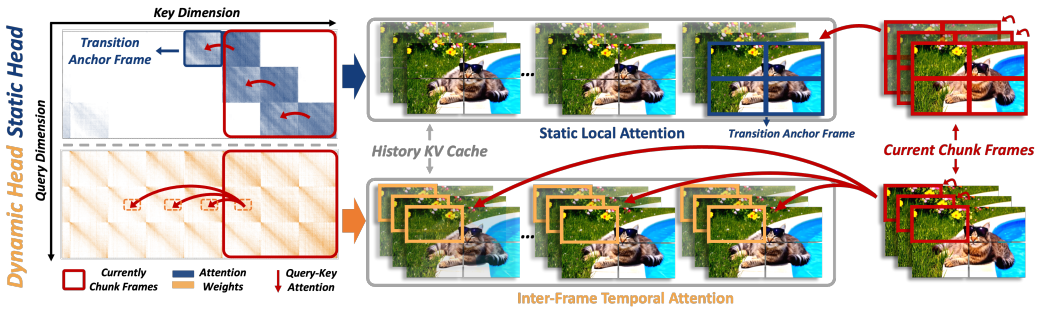

Figure 2: **Attention head patterns in AR video diffusion models**. Static heads densely focus on the intra-frame dependencies and local transition, whereas dynamic heads sparsely capture the inter-frame evolution of corresponding regions.

2 Observation

In this section, we investigate the underlying principles of KV cache utilization in AR video diffusion models to motivate the compression strategy. We begin with intuitive observations of attention head patterns in Section 2.1, followed by empirical evidence that verifies the functional roles of different heads in Section 2.2, and finally investigate their stability and generalizability in Section 2.3.

2.1 Attention Head Pattern of Autoregressive Video Diffusion Models

Video diffusion models typically exhibit a spatial-temporal functional specialization (Xi et al., 2025; Yang et al., 2025). In AR video diffusion models, the introduction of KV cache allows this property to manifest over the evolving context of autoregressive generation.. This naturally raises the question:

Question 1: *How do AR video diffusion models organize attention over spatiotemporal content during chunk-wise long generation?*

To address this, we employ models including Wan2.1 (Wan et al., 2025), SkyReels-V2 (Chen et al., 2025), Self Forcing (Huang et al., 2025a), and Longlive (Yang et al., 2026) to generate videos using VBench (Huang et al., 2024) prompts [2]. Through comparison across bidirectional, autoregressive, multi-step and few-step video diffusion models, we categorize the attention heads into **static head** and **dynamic head**, and summarize the patterns in Figure 2:

Observation 1 (Static and Dynamic Head Pattern): *Static heads consistently attend to the current chunk and the most recent frame densely, preserving intra-frame fidelity and visual continuity across local autoregressive chunks. Dynamic heads sparsely capture the inter-frame evolution of corresponding regions to exploit long-range temporal context.*

As illustrated in Figure 2, the static head primarily attends to local spatial frames. Consequently, its attention map exhibits a chunk-wise pattern, with consistent attention placed on the currently generated chunk. Concurrently, static heads also place particular attention on the most recent frame in the historical cache, which we refer to as the *transition anchor frame*. We regard this as a distinctive characteristic of autoregressive video diffusion models, where transitions across autoregressive chunks are primarily mediated through local, static attention to the transition anchor frame, rather than to the full set of historical frames. Through this attention pattern, the static head provides a structural scaffold for the video. We regard it as an invariant and static behavior in autoregressive video generation, independent of the prompts and the specific generated content.

In contrast, the dynamic head exhibits a diagonal stripe pattern with a constant interval in the KV cache. This phenomenon is highly interpretable: since both the number of frames per chunk and the number of tokens per frame are fixed, the same spatial region across different frames appears with a fixed stride along the key dimension. As a result, the dynamic head associates each generated region with information from the corresponding regions in historical frames (motion, object evolution),

[2] We provide the detailed attention map visualization (Figure 9) in Appendix C.

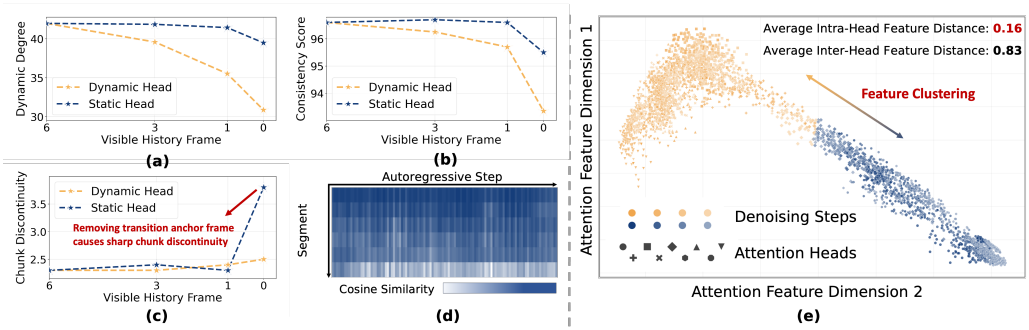

Figure 3: **Ablations on head functional properties**. **Left:** (a-c) Gradually masking contextual information for dynamic heads leads to a progressive decline in dynamic degree and consistency, while masking the transition frame for static heads causes a sharp rise in chunk discontinuity, revealing different functional emphases. (d) The cosine similarity of key states of adjacent frames across different autoregressive steps and different frame segments. **Right:** (e) Principal component analysis (PCA) of attention features from a subset of attention heads, evaluated across one hundred prompt samples and four denoising steps. The observed head functioning is highly stable.

enabling the model to exploit long-range temporal context. Because different spatial regions evolve dynamically over the course of the video, we refer to this head pattern as dynamic.

2.2 Functional Properties of Static and Dynamic Heads

Having established an intuitive interpretation of the head patterns, we proceed to further examine the functional roles of the two types of heads.

Question 2: *What are the specific functional roles of the two types of heads in aotoregressive generation, and what context in the KV cache is genuinely essential for them?*

We investigate this by conducting separate ablation studies that progressively mask the context accessible to each head until all historical frames are removed. Videos are generated using LongLive (Yang et al., 2026), a high-performing model for long-horizon video generation. For evaluation, we adopt 128 prompts from MovieGen (Polyak et al., 2024) and use VBench-Long (Huang et al., 2025c) as benchmark. Since existing metrics do not adequately capture flickering and broken transitions at chunk boundaries, we introduce a metric based on optical flow, termed *chunk discontinuity* [3]. It measures abrupt changes through the difference in optical flow between adjacent video frames. Through the above evaluation, we summarize our empirical finding as:

Observation 2 (Functional Properties): *Static heads are crucial for visual continuity across autoregressive chunks while being insensitive to distant context. Dynamic heads govern subject consistency and motion dynamics, drawing on global context that is informative yet partially redundant.*

As shown in Figure 3 (a-c), as the visible historical frames is progressively masked, both dynamic degree and consistency gradually decline for dynamic heads, while remaining nearly stable for static heads. By contrast, masking the most recent frame (i.e., transition anchor frame) causes a sharp increase in chunk discontinuity for static heads, indicating significantly more abrupt transitions at chunk boundaries. Meanwhile, the result also verify that transitions across adjacent chunks are primarily mediated through dense attention to the transition anchor frame, rather than being distributed in the historical context. In other words, local continuity and the utilization of historical context can be seen as focused head-level behaviors that were overlooked in previous studies.

Moreover, we observe that adjacent frames in autoregressive generation exhibit substantial regional similarity and redundancy, and part of the frame segments have greater similarity throughout the generation, as shown in Figure 3 (d). These insights provide empirical support for our hybrid compression scheme, that static heads are pruned statically while dynamic heads are pruned based on similarity, decoupling static local continuation from dynamic long context utilization.

[3]The detailed formulation and metric effectiveness are provided in Appendix B.

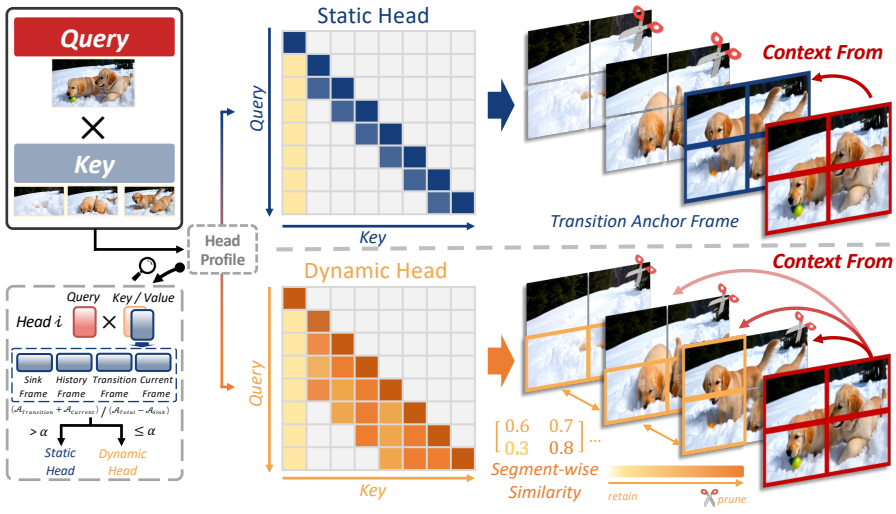

Figure 4: **Overview of FORCING-KV**. We perform offline head profiling to classify attention heads into Static and Dynamic. During inference, static heads are pruned leveraging the structural pattern, while dynamic heads are pruned adaptively based on segment-wise similarity of adjacent frames. For simplicity, we use one frame per chunk as an example.

2.3 STABILITY OF HEAD PROPERTIES

Furthermore, we conduct a statistical analysis of the above head properties to address the question:

Question 3: *Do the above head properties remain stable, or do they exhibit substantial variation?*

To provide a comprehensive study, we experiment on LongLive with 100 standard VBench prompts across all four denoising steps. For a random subset of heads, we extract the key states of each latent frame in the cache and compute frame-wise attention scores as features. Based on the features, we visualize the distribution using principal component analysis (PCA) as shown in Figure 3 (e).

Observation 3 (Stability of Head Properties): *Head functional specialization remains stable across different samples and denoising steps in its attention distribution.*

As shown in Figure 3 (e), the features of each head form tightly clustered distributions at both sample level and denoising steps level, with average intra-head divergence (0.16) substantially smaller than average inter-head divergence(0.83). This provides a basis for effective head classification.

3 FORCING-KV

Motivated by these observations, we propose FORCING-KV, a hybrid compression scheme for autoregressive diffusion models, as depicted in Figure 4. We conduct a model-level **offline head profiling** in Section 3.1 that identifies static and dynamic heads. We then apply **static structural pruning** for static heads in Section 3.2 and **dynamic similarity pruning** for dynamic heads in Section 3.3.

3.1 OFFLINE HEAD PROFILING

Given the consistent functional behaviors of static and dynamic heads in *Observation 3*, we propose an offline head profiling strategy to categorize them before actual inference. According to the head pattern, the attention mass of static heads along the key dimension is concentrated on the currently generated chunk and the transition anchor frame, whereas the attention mass of dynamic heads is distributed more evenly across the entire attention window. This provides an intuitive criterion for head classification: utilizing the proportion of total attention mass assigned to the local static frames. Since some models apply special treatment to sink frames in their training recipes (Yesiltepe et al., 2025; Yang et al., 2026; Lu et al., 2026), we exclude the sink frames from this computation. Finally, given the per-head attention mass assigned to the entire attention window $\mathcal{A}_{Total}$, the current chunk

$\mathcal{A}_{Curr}$, the transition frame $\mathcal{A}_{Transition}$, and the sink frame $\mathcal{A}_{Sink}$, the head profiling metric is defined as:

$$\text{HeadType} = \begin{cases} \text{Static}, & \text{if } \dfrac{\mathcal{A}_{\text{Curr}} + \mathcal{A}_{\text{Transition}}}{\mathcal{A}_{\text{Total}} - \mathcal{A}_{\text{Sink}}} > \alpha, \\ \text{Dynamic}, & \text{otherwise.} \end{cases} \tag{1}$$

Here, α is a model-specific hyperparameter, and the classification can be completed within a single prompt. Notably, the metric aligns naturally with our subsequent compression strategy, where frames with lower accumulated attention mass are better eviction candidates, consistent with KV eviction methods such as H_2O (Zhang et al., 2023). In Section 4.3, we show that this simple criterion is sufficient to distinguish the majority of heads and is not sensitive to α, which promotes scalability.

3.2 Static Structural Pruning for Static Heads

In *Observation 1*, we show that static heads are highly sensitive to the transition anchor frame while underutilizing distant context. Therefore, we adopt a structured compression strategy for static heads by retaining the key and value states of the transition anchor frame and the current chunk to preserve intra-frame spatial structure and local chunk transitions. Given that each chunk contains C frames and each frame consists of F tokens, for the i-th AR step, the self-attention is formulated as:

$$\mathbf{O}_i^{\text{static}} = \text{Attention}\Big(Q_{Curr}, [K_{Sink}, K_{Transition}, K_{Curr}], [V_{Sink}, V_{Transition}, V_{Curr}]\Big). \tag{2}$$

Here, K_{Sink}, $K_{\text{Transition}}$, and K_{Curr} denote the key states corresponding to the sink frame, the transition frame range $[(iC-1)F, iCF]$, and the current chunk range $[iCF, (i+1)CF]$, respectively. Static compression structurally preserves the sink frames and the transition anchor frame, and can be readily extended to frame-wise generation as well.

3.3 Dynamic Similarity Pruning for Dynamic Heads

Dynamic heads assign high attention mass to regions separated by fixed intervals, corresponding to the same spatial locations across different frames. However, these segments differ substantially in their temporal evolution as shown in Figure 3 (d): some remain highly similar across frames with only limited variation (potentially background regions or static objects), whereas others undergo continuous changes due to motion, actions, or object evolution. Accordingly, we assess the redundancy of different segments during generation for dynamic compression.

Specifically, we first partition each latent frame into n segments and compute the segment-wise cosine similarity between corresponding segments in each frame and its next adjacent frame. Given a compression ratio r, we evict the top-k segments in each frame with the highest similarity values, preserving $(1-r)\%$ of all frame segments. Similar to Equation (2), let $K_{t,j}$ and $V_{t,j}$ denote the key and value state of the j-th segment in the t-th history frame, and we formulate the compression:

$$s_t^{(j)} = \text{Cosine}\big(K_{t,j},\, K_{t+1,j}\big), \qquad \mathcal{I}_t^{\text{keep}} = \text{BottomK}\Big(\{s_t^{(j)}\}_{j=1}^{n},\, \lfloor (1-r)n \rfloor\Big). \tag{3}$$

Here, $s_t^{(j)}$ are the cosine similarity values, and $\mathcal{I}_t^{\text{keep}}$ are the indices of the selected low-similarity segment to keep. For similarity computation, we use only the key states over attention heads in the first DiT block as a proxy to avoid the substantial computation of all blocks. This design choice is justified in Appendix O. Given the compressed key and value states $\widetilde{K}_{\mathcal{H}} = \bigcup_{t\in\mathcal{H}}\{K_{t,j} \mid j \in \mathcal{I}_t^{\text{keep}}\}$ and $\widetilde{V}_{\mathcal{H}} = \bigcup_{t\in\mathcal{H}}\{V_{t,j} \mid j \in \mathcal{I}_t^{\text{keep}}\}$, the self-attention computation for dynamic head is denoted as:

$$\mathbf{O}_i^{\text{dynamic}} = \text{Attention}\Big(Q_{Curr}, \Big[K_{\text{Sink}}, \widetilde{K}_{\mathcal{H}}, K_{Curr}\Big], \Big[V_{\text{Sink}}, \widetilde{V}_{\mathcal{H}}, V_{Curr}\Big]\Big). \tag{4}$$

The underlying observation is that adjacent video frames contain similar visual cues, resulting in redundancy within short temporal windows. (Gao et al., 2025). As a result, removing highly similar segments introduces minimal information loss. In Section 4.3, we empirically show that this scheme offers an advantage over random and uniform token reduction in temporal dynamics.

Table 1: Quantitative results on efficiency and quality on **long video generation** with VBenchLong.

Setting	Method	Efficiency Metrics		Specific Metrics		General Metrics					
		FPS↑	Speedup↑	Chunk Disc.↓	Dynamic Degree↑	**Total Score**↑	Imaging Quality↑	Subject Cons.↑	Background Cons.↑	Motion Smooth.↑	Aesthetic Quality↑
LongLive 60-second	Vanilla	20.48	1.00×	2.6	42.40	80.23	68.84	97.82	96.82	98.77	61.47
	StreamingLLM	22.50	1.10×	2.5	40.89	80.28▲	69.40	97.91	96.88	98.75	61.78
	Dummy Forcing ($L = 1$)	28.06	1.37×	3.6	26.02	79.35	71.25	97.95	96.94	98.79	62.01
	Dummy Forcing ($L = 2$)	26.36	1.29×	2.9	34.10	79.44	70.71	97.38	96.52	98.45	61.40
	FORCING-KV (Ours)	26.71	1.30×	2.5▲	43.56▲	**80.43**▲	70.24	97.79	96.66	98.57	61.50
LongLive 30-second	Vanilla	21.10	1.00×	2.4	42.54	80.38	68.91	97.99	96.93	98.80	61.67
	StreamingLLM	22.34	1.06×	2.4	40.93	80.38	69.35	96.06	96.98	98.78	61.92
	Dummy Forcing ($L = 1$)	27.45	1.30×	3.0	26.56	79.37	70.73	98.07	96.99	98.82	62.06
	Dummy Forcing ($L = 2$)	26.21	1.24×	2.3	33.90	79.61	70.65	97.64	96.64	98.54	61.73
	FORCING-KV (Ours)	26.77	1.27×	2.4	43.65▲	**80.65**▲	70.29	98.00	96.81	98.66	61.98
Self Forcing 30-second	Vanilla	17.76	1.00×	3.4	46.86	79.72	67.63	97.20	96.38	98.14	61.14
	StreamingLLM	21.86	1.23×	2.8▲	54.50▲	80.06▲	67.90	96.93	96.16	98.10	59.67
	Dummy Forcing ($L = 1$)	27.75	1.56×	3.5	46.55	79.95▲	69.05	97.20	96.38	98.14	61.13
	Dummy Forcing ($L = 6$)	22.11	1.24×	3.4	50.47▲	79.78▲	68.83	96.62	96.03	98.03	59.97
	FORCING-KV (Ours)	26.65	1.50×	2.7▲	52.23▲	**80.07**▲	68.67	97.00	96.08	98.08	60.16

↑ indicates higher is better, ↓ indicates lower is better. ▲ highlights improved performance over vanilla.

4 EXPERIMENTS

Models and Baselines. We conduct the experiments using mainstream AR video generation models including Self Forcing (Huang et al., 2025a), Longlive (Yang et al., 2026), Causal Forcing (Zhu et al., 2026), and Krea-Realtime-14B (Millon, 2025) [4]. We compare our method against both the vanilla sliding window cache setting and representative training-free KV cache compression baselines. StreamingLLM (Xiao et al., 2023) serves as a naive baseline that uniformly retains sink and shorter sliding window tokens for all heads. Dummy Forcing (Guo et al., 2026) employs a speed-oriented local pruning strategy. Method implementation details are in Appendix K.

Benchmarks and Evaluation Metrics. For the **performance metric**, we evaluate both short and long video generation on VBench (Huang et al., 2024) and VBenchLong (Huang et al., 2025c), and conduct a user study [5]. Specifically, we generate 5-second videos using 946 official VBench prompts and evaluate all 16 dimensions. For 30-second and 60-second videos, we adopt 128 prompts from MovieGen (Polyak et al., 2024), weighting the total score using the standard VBench coefficients, consistent with previous work (Yin et al., 2025b; Lu et al., 2026). All prompts are sampled with 5 different seeds. We use *chunk discontinuity* metric defined in Appendix B to quantify chunk jittering. For the **efficiency metric**, we calculate the frames generated per second (FPS) within the diffusion transformer (DiT) and the corresponding speedups on a single NVIDIA H200 GPU, consistent with previous studies (Huang et al., 2025a; Yang et al., 2026) [6]. End-to-end time is in Appendix J.

4.1 MAIN RESULTS

Comparison with Vanilla Base Model. As shown in Tables 1 and 2 and Figure 5, we evaluate our method on LongLive and Self Forcing across generation length of 5, 30, and 60 seconds. Our method achieves 1.30× and 1.50× speedups on LongLive and Self Forcing for long video generation, and 1.35× and 1.44× speedups for short video generation. After our head-wise compression, over 73% and 53% of the KV cache is evicted for Self Forcing and Longlive, which provides the efficiency gains. Moreover, FORCING-KV maintains comparable performance on VBench (80.43 vs. 80.23), visual quality in the user study (45.0% vs. 50.0%), and stronger temporal dynamics (52.8% vs. 42.2%). We assume that existing models treat all heads uniformly during training, forcing certain heads (i.e., static heads) to attend to distant context, but such capability may not be sufficiently learned. As a result, compressing these heads can even lead to a positive effect since noise is removed. This also explains why StreamingLLM can also maintain performance after evicting distant tokens. The comparable performance of FORCING-KV across multiple metrics also empirically reflects the redundancy in the KV cache, answering our central question in Section 1.

[4] Results on Causal Forcing, Krea-14B and interactive generation on Longlive are in Appendices E and F.

[5] Setup, protocol, and screenshots of the user study are in Appendix D.

[6] The overhead of FORCING-KV is included throughout the section, with a detailed breakdown Appendix J.

Table 2: Quantitative results on **short video generation** with VBench.

Setting	Method	Efficiency Metrics		Specific Metrics		General Metrics		
		FPS↑	Speedup↑	Chunk Disc.↓	Dynamic Degree↑	**Total Score**↑	Quality Score↑	Semantic Score↑
LongLive 5-second	Vanilla	21.85	1.00×	2.1	40.28	83.19	83.63	81.41
	StreamingLLM	24.56	1.12×	2.1	43.33▲	82.85	83.29	81.09
	Dummy Forcing ($L = 1$)	29.69	1.36×	2.4	37.22	82.83	83.33	80.86
	Dummy Forcing ($L = 2$)	28.36	1.30×	2.1	36.67	83.15	83.73	80.85
	FORCING-KV (Ours)	29.58	1.35×	2.1	45.56▲	**83.23**▲	83.84	80.80
Self Forcing 5-second	Vanilla	19.56	1.00×	2.1	66.39	83.91	84.71	80.70
	StreamingLLM	23.36	1.19×	2.4	65.00	83.80	84.52	80.89
	Dummy Forcing ($L = 1$)	28.31	1.45×	2.6	64.44	83.87	84.64	80.81
	Dummy Forcing ($L = 6$)	25.41	1.30×	2.4	63.33	83.79	84.50	80.96
	FORCING-KV (Ours)	28.18	1.44×	2.1	69.17▲	**83.98**▲	84.82	80.61

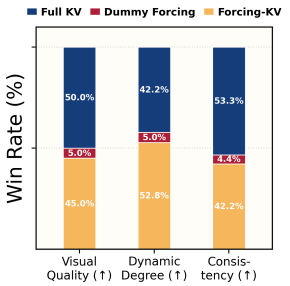

Figure 5: User study.

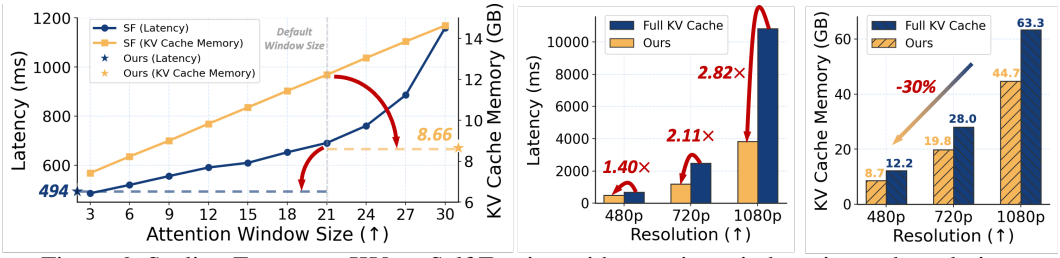

Figure 6: Scaling FORCING-KV on Self Forcing with attention window size and resolution.

Comparison with Other Compression Strategies. ① Vs. StreamingLLM: With its sliding window design closely aligned with the training regime of the base model, StreamingLLM serves as a competitive baseline. However, FORCING-KV achieves head-level decomposition, which leads to substantially higher compression ratios and resulting speedups (1.50× vs. 1.23×). ② Vs. Dummy Forcing: While achieving comparable speedups, FORCING-KV consistently achieves substantially higher quality. Specifically, our compression is grounded in the observations from Section 2 and accurately preserves the anchor for transition, while Dummy Forcing does not apply in its profiling and compression design. This results in markedly lower chunk discontinuity (2.5 vs. 3.6 and 2.7 vs. 3.5). Moreover, FORCING-KV leverages similarity to compress historical cache and achieves a much higher dynamic degree (43.56 vs. 26.02). Dummy Forcing achieves higher image quality in some cases, but this may reflect the benchmark's preference for static content. Notably, chunk continuity and motion dynamics have a substantial impact on perceptual quality, which explains why our method shows a clear advantage over Dummy Forcing in the user study (45.0% vs. 5.0%) [7].

4.2 SCALING WITH ATTENTION WINDOW SIZE AND RESOLUTION

In this section, we empirically show that the efficiency gains of FORCING-KV become increasingly pronounced as the attention window and resolution scale. As shown in Figure 6, the KV cache grows linearly, causing the attention computation to scale quadratically and the memory consumption to scale linearly. Therefore, for the same compression ratio, FORCING-KV delivers increasing gains, rising from 1.40× to **2.82**× with constant ∼30% peak cache memory reduction. Theoretically, resolution affects the token numbers in each frame, while the adaptation mechanism of FORCING-KV remains unchanged [8]. We believe that high-resolution video generation and longer context memory horizon (Cai et al., 2026b) are promising future directions, highlighting the potential of our method.

4.3 ABLATION STUDY

In Table 3, we conduct separate ablation studies on the effectiveness of the head profiling strategy, the hybrid compression design, and the dynamic similarity pruning strategy. Unless otherwise specified, all experiments are performed on 30-second video generation with LongLive [9].

[7] The difference can be observed in the qualitative examples provided in Appendix R and video demos.
[8] We provide theoretical demonstration of this property in Appendix P.
[9] Full ablation results for 5-second generation on LongLive and Self Forcing are reported in Appendix I.

Table 3: Ablation study of head profiling strategy and hybrid KV cache compression on LongLive (30-second video generation).

Method	Chunk Disc.↓	Dynamic Degree↑	Total Score↑
Head Profiling			
FORCING-KV ($\alpha = 0.8$)	2.4	43.65	80.65
FORCING-KV ($\alpha = 0.5$)	2.4	42.87	80.63
Random Profiling	2.5	40.72	80.44
Human Profiling	2.3	44.44	80.71
KV Cache Compression			
FORCING-KV	2.4	43.65	80.65
w/o static-head cache	4.1	42.58	79.57
w/o dynamic-head cache	2.6	40.78	80.25

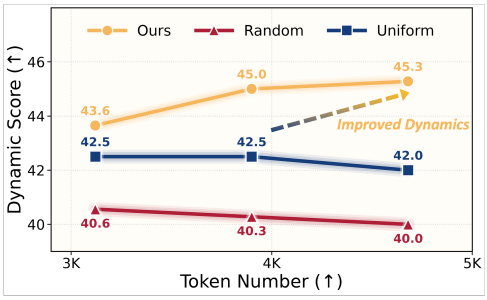

Figure 7: Comparison of random, uniform token pruning, and our proposed similarity-based pruning under different token retention budgets.

Effectiveness of Head Profiling. Owing to the clear distinction among head types and their stability, the simple profiling strategy of FORCING-KV is sufficient to identify the majority of heads and achieves performance close to that of a strong manual profiling baseline (80.65 vs. 80.71). We also include a random profiling baseline. Because the cache retained for dynamic heads also subsumes the static portion, the degradation under random profiling mainly arises when dynamic heads are misclassified as static, causing dynamic degree to drop (40.72 vs. 43.65). This indicates the necessity of meaningful head classification. In addition, we find that head profiling of FORCING-KV is insensitive to the hyperparameter α. Decreasing α from 0.8 to 0.5 classifies more heads as static heads, which leads to only a slight drop in dynamic degree.

Effectiveness of Hybrid Compression. FORCING-KV retains the transition anchor frame for static heads and preserves the cache segments for dynamic heads. We separately prune the corresponding cache of each head type to study the effect of hybrid head modeling, denoted as *w/o static-head cache* and *w/o dynamic-head cache*. As shown in Table 3, pruning the cache of static head leads to a substantial increase in chunk discontinuity (4.1 vs. 2.4), which in turn degrades the overall score. In contrast, pruning all cache of dynamic head mainly reduces the dynamic degree (40.78 vs. 43.65), consistent with our finding in Section 2.2. This indicates that the two components retained by our compression strategy are indeed effective, and that they correspond to their function roles.

Effectiveness of Dynamic Similarity Pruning. To validate the effectiveness of the dynamic similarity pruning strategy in FORCING-KV, we conduct comparisons against other pruning criteria including random token pruning and uniform token pruning, as shown in Figure 7. Our method consistently achieves higher dynamic degree, and improves temporal dynamics as the token budget increases. We attribute this advantage to two factors. First, adjacent frames in autoregressive generation often exhibit substantial similarity as visualized in Figure 3 (d), making them suited for cross-frame pruning. Second, our method operates in a segment-wise manner, which is better aligned with the continuity of video content than discrete token-wise strategies. This design is also consistent with our observation in Figure 2 that the stripe patterns of dynamic heads exhibit a certain width.

5 CONCLUSION

We presented FORCING-KV, a hybrid KV cache compression framework for autoregressive video diffusion models. We begin by uncovering a universal head specialization pattern shared by mainstream autoregressive video models, which naturally motivates our compression strategy. While maintaining output quality, FORCING-KV achieves up to 1.35× and 1.50× speedups together with 30% cache memory reduction. The efficiency gain potentially scales with attention window size and resolution, highlighting its potential. Our work seeks to answer an important question that how autoregressive video models achieve smooth video continuation, long-range consistency, and dynamics through head-wise causal attention mechanisms, which distinguish them from bidirectional models. And it paves the way toward efficient and scalable context utilization in long video generation [10] [11].

[10] Hyperparameter sensitivity and head distribution are discussed in Appendices H and M.

[11] Limitations and future work are discussed in Appendix Q.

AI USE STATEMENT

Large Language Models (LLMs) were used to aid in the code writing and manuscript polishing. Specifically, the usage includes refining the language, improving readability, and ensuring clarity in the paper. It is important to note that LLMs were not involved in the ideation, research methodology, or experimental design.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we provide the complete implementation of FORCING-KV together with runnable demos in the supplementary submission. The provided *README* contains detailed instructions for environment setup, dependency installation, and execution. Users can directly run the provided inference scripts to reproduce the demo results included in the submission. We additionally document the experimental settings, implementation details, and hyperparameter configurations in the main paper and Appendix K to support faithful reproduction of our results.

REFERENCES

Krish Agarwal, Zhuoming Chen, Cheng Luo, Yongqi Chen, Haizhong Zheng, Xun Huang, Atri Rudra, and Beidi Chen. MonarchRT: Efficient Attention for Real-Time Video Generation. *arXiv preprint arXiv:2602.12271*, 2026.

Shengqu Cai, Weili Nie, Chao Liu, Julius Berner, Lvmin Zhang, Nanye Ma, Hansheng Chen, Maneesh Agrawala, Leonidas Guibas, Gordon Wetzstein, et al. Mode Seeking meets Mean Seeking for Fast Long Video Generation. *arXiv preprint arXiv:2602.24289*, 2026a.

Shengqu Cai, Ceyuan Yang, Lvmin Zhang, Yuwei Guo, Junfei Xiao, Ziyan Yang, Yinghao Xu, Zhenheng Yang, Alan Yuille, Leonidas Guibas, Maneesh Agrawala, Lu Jiang, and Gordon Wetzstein. Mixture of contexts for long video generation. In *ICLR*, 2026b.

Guibin Chen, Dixuan Lin, Jiangping Yang, Chunze Lin, Junchen Zhu, Mingyuan Fan, Hao Zhang, Sheng Chen, Zheng Chen, Chengcheng Ma, et al. Skyreels-v2: Infinite-length film generative model. *arXiv preprint arXiv:2504.13074*, 2025.

Hanmo Chen, Chenghao Xu, Xu Yang, Xuan Chen, and Cheng Deng. Past-and Future-Informed KV Cache Policy with Salience Estimation in Autoregressive Video Diffusion. *arXiv preprint arXiv:2601.21896*, 2026a.

Junsong Chen, Yuyang Zhao, Jincheng Yu, Ruihang Chu, Junyu Chen, Shuai Yang, Xianbang Wang, Yicheng Pan, Daquan Zhou, Huan Ling, et al. Sana-video: Efficient video generation with block linear diffusion transformer. *ICLR*, 2026b.

Shuo Chen, Cong Wei, Sun Sun, Ping Nie, Kai Zhou, Ge Zhang, Ming-Hsuan Yang, and Wenhu Chen. Context forcing: Consistent autoregressive video generation with long context, 2026c. URL https://arxiv.org/abs/2602.06028.

Justin Cui, Jie Wu, Ming Li, Tao Yang, Xiaojie Li, Rui Wang, Andrew Bai, Yuanhao Ban, and Cho-Jui Hsieh. Self-Forcing++: Towards Minute-Scale High-Quality Video Generation. *ICLR*, 2026a.

Justin Cui, Jie Wu, Ming Li, Tao Yang, Xiaojie Li, Rui Wang, Andrew Bai, Yuanhao Ban, and Cho-Jui Hsieh. LoL: Longer than Longer, Scaling Video Generation to Hour. *arXiv preprint arXiv:2601.16914*, 2026b.

Xiaomeng Fu, Jia Li, Yiming Hu, Yong Wang, Hayden Kwok-Hay So, Jiao Dai, Xiangxiang Chu, and Jizhong Han. Towards memory-efficient autoregressive video generation via instance-specific parametric absorption, 2026. URL https://arxiv.org/abs/2607.00712.

Kaifeng Gao, Jiaxin Shi, Hanwang Zhang, Chunping Wang, Jun Xiao, and Long Chen. Ca2-VDM: Efficient Autoregressive Video Diffusion Model with Causal Generation and Cache Sharing. In *International Conference on Machine Learning*, pp. 18550–18565. PMLR, 2025.

Zelin Gao, Qiuyu Wang, Jiapeng Zhu, Jingye Chen, Zichen Liu, Qingyan Bai, Jiahao Wang, Yufeng Yuan, Hanlin Wang, Yichong Lu, Ka Leong Cheng, Haojie Zhang, Jian Gao, Tianrui Feng, Yuzheng Liu, Yao Yao, Yinghao Xu, Xing Zhu, Yujun Shen, and Hao Ouyang. Infinite worlds with versatile interactions. *arXiv preprint arXiv:2607.07534*, 2026.

Hang Guo, Zhaoyang Jia, Jiahao Li, Bin Li, Yuanhao Cai, Jiangshan Wang, Yawei Li, and Yan Lu. Efficient Autoregressive Video Diffusion with Dummy Head. *arXiv preprint arXiv:2601.20499*, 2026.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self Forcing: Bridging the Train-Test Gap in Autoregressive Video Diffusion. *NeurIPS*, 2025a.

Yubo Huang, Hailong Guo, Fangtai Wu, Shifeng Zhang, Shijie Huang, Qijun Gan, Lin Liu, Sirui Zhao, Enhong Chen, Jiaming Liu, et al. Live avatar: Streaming real-time audio-driven avatar generation with infinite length. *arXiv preprint arXiv:2512.04677*, 2025b.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Ziqi Huang, Fan Zhang, Xiaojie Xu, Yinan He, Jiashuo Yu, Ziyue Dong, Qianli Ma, Nattapol Chanpaisit, Chenyang Si, Yuming Jiang, Yaohui Wang, Xinyuan Chen, Ying-Cong Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench++: Comprehensive and versatile benchmark suite for video generative models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025c. doi: 10.1109/TPAMI.2025.3633890.

Sihui Ji, Xi Chen, Shuai Yang, Xin Tao, Pengfei Wan, and Hengshuang Zhao. Memflow: Flowing adaptive memory for consistent and efficient long video narratives, 2025. URL https://arxiv.org/abs/2512.14699.

Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, Kathrina Wu, Qin Lin, Junkun Yuan, Yanxin Long, Aladdin Wang, Andong Wang, Changlin Li, Duojun Huang, Fang Yang, Hao Tan, Hongmei Wang, Jacob Song, Jiawang Bai, Jianbing Wu, Jinbao Xue, Joey Wang, Kai Wang, Mengyang Liu, Pengyu Li, Shuai Li, Weiyan Wang, Wenqing Yu, Xinchi Deng, Yang Li, Yi Chen, Yutao Cui, Yuanbo Peng, Zhentao Yu, Zhiyu He, Zhiyong Xu, Zixiang Zhou, Zunnan Xu, Yangyu Tao, Qinglin Lu, Songtao Liu, Dax Zhou, Hongfa Wang, Yong Yang, Di Wang, Yuhong Liu, Jie Jiang, and Caesar Zhong. HunyuanVideo: A Systematic Framework For Large Video Generative Models, 2025.

Kunjun Li, Zigeng Chen, Cheng-Yen Yang, and Jenq-Neng Hwang. Memory-Efficient Visual Autoregressive Modeling with Scale-Aware KV Cache Compression. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Wuyang Li, Wentao Pan, Po-Chien Luan, Yang Gao, and Alexandre Alahi. Stable Video Infinity: Infinite-Length Video Generation with Error Recycling. In *International Conference on Learning Representations*, 2026.

Bin Lin, Yunyang Ge, Xinhua Cheng, Zongjian Li, Bin Zhu, Shaodong Wang, Xianyi He, Yang Ye, Shenghai Yuan, Liuhan Chen, Tanghui Jia, Junwu Zhang, Zhenyu Tang, Yatian Pang, Bin She, Cen Yan, Zhiheng Hu, Xiaoyi Dong, Lin Chen, Zhang Pan, Xing Zhou, Shaoling Dong, Yonghong Tian, and Li Yuan. Open-Sora Plan: Open-Source Large Video Generation Model, 2024.

Feng Liu, Shiwei Zhang, Xiaofeng Wang, Yujie Wei, Haonan Qiu, Yuzhong Zhao, Yingya Zhang, Qixiang Ye, and Fang Wan. Timestep Embedding Tells: It's Time to Cache for Video Diffusion Model. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7353–7363. IEEE, 2025.

Kunhao Liu, Wenbo Hu, Jiale Xu, Ying Shan, and Shijian Lu. Rolling forcing: Autoregressive long video diffusion in real time. *ICLR*, 2026.

Yunhong Lu, Yanhong Zeng, Haobo Li, Hao Ouyang, Qiuyu Wang, Ka Leong Cheng, Jiapeng Zhu, Hengyuan Cao, Zhipeng Zhang, Xing Zhu, et al. Reward forcing: Efficient streaming video generation with rewarded distribution matching distillation. *CVPR*, 2026.

Chengtao Lv, Yumeng Shi, Yushi Huang, Ruihao Gong, Shen Ren, and Wenya Wang. Light Forcing: Accelerating Autoregressive Video Diffusion via Sparse Attention. *arXiv preprint arXiv:2602.04789*, 2026.

Yuexiao Ma, Xuzhe Zheng, Jing Xu, Xiwei Xu, Feng Ling, Xiawu Zheng, Huafeng Kuang, Huixia Li, XING WANG, Xuefeng Xiao, et al. Flow Caching for Autoregressive Video Generation. In *The Fourteenth International Conference on Learning Representations*, 2026.

Erwann Millon. Krea Realtime 14B: Real-time Video Generation, 2025.

William Peebles and Saining Xie. Scalable Diffusion Models with Transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4172–4182. IEEE, 2023.

Adam Polyak, Amit Zohar, Andrew Brown, Andros Tjandra, Animesh Sinha, Ann Lee, Apoorv Vyas, Bowen Shi, Chih-Yao Ma, Ching-Yao Chuang, et al. Movie gen: A cast of media foundation models. *arXiv preprint arXiv:2410.13720*, 2024.

Ziran Qin, Youru Lv, Mingbao Lin, Hang Guo, Zeren Zhang, Danping Zou, and Weiyao Lin. Head-aware kv cache compression for efficient visual autoregressive modeling. *arXiv preprint arXiv:2504.09261*, 2025.

Suraj Ranganath, Vaishak Menon, and Anish Patnaik. KV Cache Quantization for Self-Forcing Video Generation: A 33-Method Empirical Study, 2026.

Dvir Samuel, Issar Tzachor, Matan Levy, Micahel Green, Gal Chechik, and Rami Ben-Ari. Fast Autoregressive Video Diffusion and World Models with Temporal Cache Compression and Sparse Attention. *arXiv preprint arXiv:2602.01801*, 2026.

Meituan LongCat Team, Xunliang Cai, Qilong Huang, Zhuoliang Kang, Hongyu Li, Shijun Liang, Liya Ma, Siyu Ren, Xiaoming Wei, Rixu Xie, and Tong Zhang. LongCat-Video Technical Report. *arXiv preprint arXiv:2510.22200*, 2025.

Zachary Teed and Jia Deng. RAFT: Recurrent All-Pairs Field Transforms for Optical Flow. In *European Conference on Computer Vision*, pp. 402–419, 2020.

Hansi Teng, Hongyu Jia, Lei Sun, Lingzhi Li, Maolin Li, Mingqiu Tang, Shuai Han, Tianning Zhang, WQ Zhang, Weifeng Luo, et al. Magi-1: Autoregressive video generation at scale. *arXiv preprint arXiv:2505.13211*, 2025.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Jing Wang, Fengzhuo Zhang, Xiaoli Li, Vincent Y. F. Tan, Tianyu Pang, Chao Du, Aixin Sun, and Zhuoran Yang. Error analyses of auto-regressive video diffusion models: A unified framework, 2025. URL https://arxiv.org/abs/2503.10704.

Haocheng Xi, Shuo Yang, Yilong Zhao, Chenfeng Xu, Muyang Li, Xiuyu Li, Yujun Lin, Han Cai, Jintao Zhang, Dacheng Li, et al. Sparse Video-Gen: Accelerating Video Diffusion Transformers with Spatial-Temporal Sparsity. In *Forty-second International Conference on Machine Learning*, 2025.

Xunzhi Xiang, Zixuan Duan, Guiyu Zhang, Haiyu Zhang, Zhe Gao, Junta Wu, Shaofeng Zhang, Tengfei Wang, Qi Fan, and Chunchao Guo. Pathwise Test-Time Correction for Autoregressive Long Video Generation. *arXiv preprint arXiv:2602.05871*, 2026.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient Streaming Language Models with Attention Sinks. In *The Twelfth International Conference on Learning Representations*, 2023.

Guangxuan Xiao, Jiaming Tang, Jingwei Zuo, Shang Yang, Haotian Tang, Yao Fu, Song Han, et al. DuoAttention: Efficient Long-Context LLM Inference with Retrieval and Streaming Heads. In *The Thirteenth International Conference on Learning Representations*, 2024.

Steven Xiao, XIndi Zhang, Dechao Meng, Qi Wang, Peng Zhang, and Bang Zhang. Knot Forcing: Taming Autoregressive Video Diffusion Models for Real-time Infinite Interactive Portrait Animation. *arXiv preprint arXiv:2512.21734*, 2025.

Boxun Xu, Yuming Du, Zichang Liu, Siyu Yang, Ziyang Jiang, Siqi Yan, Rajasi Saha, Albert Pumarola, Wenchen Wang, and Peng Li. Sparse Forcing: Native Trainable Sparse Attention for Real-time Autoregressive Diffusion Video Generation. *arXiv preprint arXiv:2604.21221*, 2026.

Shuai Yang, Wei Huang, Ruihang Chu, Yicheng Xiao, Yuyang Zhao, Xianbang Wang, Muyang Li, Enze Xie, Yingcong Chen, Yao Lu, et al. Longlive: Real-time Interactive Long Video Generation. *ICLR*, 2026.

Shuo Yang, Haocheng Xi, Yilong Zhao, Muyang Li, Jintao Zhang, Han Cai, Yujun Lin, Xiuyu Li, Chenfeng Xu, Kelly Peng, et al. Sparse videogen2: Accelerate video generation with sparse attention via semantic-aware permutation. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Hidir Yesiltepe, Tuna Han Salih Meral, Adil Kaan Akan, Kaan Oktay, and Pinar Yanardag. Infinity-rope: Action-controllable infinite video generation emerges from autoregressive self-rollout. *CVPR*, 2025.

Jung Yi, Wooseok Jang, Paul Hyunbin Cho, Jisu Nam, Heeji Yoon, and Seungryong Kim. Deep forcing: Training-free long video generation with deep sink and participative compression. *arXiv preprint arXiv:2512.05081*, 2025.

Tianwei Yin, Qiang Zhang, Richard Zhang, William T Freeman, Fredo Durand, Eli Shechtman, and Xun Huang. From Slow Bidirectional to Fast Autoregressive Video Diffusion Models. In *CVPR*, 2025a.

Tianwei Yin, Qiang Zhang, Richard Zhang, William T Freeman, Frédo Durand, Eli Shechtman, and Xun Huang. From Slow Bidirectional to Fast Autoregressive Video Diffusion Models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22963–22974. IEEE, 2025b.

Shenghai Yuan, Yuanyang Yin, Zongjian Li, Xinwei Huang, Xiao Yang, and Li Yuan. Helios: Real Real-Time Long Video Generation Model. *arXiv preprint arXiv:2603.04379*, 2026.

Bowen Zeng, Feiyang Ren, Jun Zhang, Xiaoling Gu, Ke Chen, Lidan Shou, and Huan Li. HybridKV: Hybrid KV Cache Compression for Efficient Multimodal Large Language Model Inference. *arXiv preprint arXiv:2604.05887*, 2026.

Jintao Zhang, Haofeng Huang, Pengle Zhang, Jia Wei, Jun Zhu, and Jianfei Chen. Sageattention2: Efficient attention with thorough outlier smoothing and per-thread int4 quantization. In *International Conference on Machine Learning (ICML)*, 2025a.

Lvmin Zhang, Shengqu Cai, Muyang Li, Gordon Wetzstein, and Maneesh Agrawala. Frame Context Packing and Drift Prevention in Next-Frame-Prediction Video Diffusion Models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Wang Zhangyang, "Atlas", and Chen Beidi. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems*, 36:34661–34710, 2023.

Zengqun Zhao, Yanzuo Lu, Ziquan Liu, Jifei Song, Jiankang Deng, and Ioannis Patras. Relax forcing: Relaxed kv-memory for consistent long video generation. *arXiv preprint arXiv:2603.21366*, 2026a.

Zengqun Zhao, Yanzuo Lu, Ziquan Liu, Jifei Song, Jiankang Deng, and Ioannis Patras. Relax forcing: Relaxed kv-memory for consistent long video generation, 2026b. URL https://arxiv.org/abs/2603.21366.

Hongzhou Zhu, Min Zhao, Guande He, Hang Su, Chongxuan Li, and Jun Zhu. Causal Forcing: Autoregressive Diffusion Distillation Done Right for High-Quality Real-Time Interactive Video Generation. *arXiv preprint arXiv:2602.02214*, 2026.

A Detailed Related Work

Autoregressive Video Diffusion Models. Video diffusion models have evolved from bidirectional, one-shot generation to autoregressive, streaming generation. Early bidirectional video diffusion models (Lin et al., 2024; Kong et al., 2025; Wan et al., 2025) are typically built upon the Diffusion Transformer (DiT) (Peebles & Xie, 2023) architecture, enabling high-quality and controllable video generation. To address the high cost of bidirectional denoising and support long-horizon generation, a growing number of works turn to autoregressive diffusion modeling (Zhang et al., 2025b; Chen et al., 2025; Teng et al., 2025). To further reduce denoising steps, CausVid (Yin et al., 2025a) reformulates bidirectional diffusion into causal generation through distribution matching distillation. Self Forcing (Huang et al., 2025a) mitigates train-test discrepancy by performing self-rollout during the training stage, and LongLive (Yang et al., 2026) further extends this framework through KV recaching and long-horizon fine-tuning. Krea-Realtime-14B (Millon, 2025) scales video generation to 14B parameters. Causal Forcing (Zhu et al., 2026) alleviates the student–teacher mismatch in the distillation process. More recently, a growing body of work (Team et al., 2025; Yesiltepe et al., 2025; Xiao et al., 2025; Huang et al., 2025b; Lu et al., 2026; Liu et al., 2026; Cui et al., 2026a; Li et al., 2026; Cui et al., 2026b; Xiang et al., 2026; Cai et al., 2026a; Yuan et al., 2026; Cai et al., 2026a) has focused on long-horizon video generation. Representative methods include Rolling Forcing (Liu et al., 2026), Reward Forcing (Lu et al., 2026), Infinite-Rope (Yesiltepe et al., 2025), and Self Forcing++ (Cui et al., 2026a), most of which build upon the *Self Forcing* training paradigm. The growing interest in memory mechanisms (Zhang et al., 2025b; Wang et al., 2025; Ji et al., 2025; Chen et al., 2026c; Zhao et al., 2026b) further highlights the importance of long-context utilization. These efforts jointly reflect a broader trend toward long-horizon video generation, long-context utilization, and the potential for a *train-long–test-long* strategy, where effective KV cache utilization is a critical factor for scalability.

Efficient Video Generation. Video diffusion models are computationally expensive due to heavy attention computation and multi-step denoising. For bidirectional models, inference is typically accelerated through sparse attention (Xi et al., 2025; Yang et al., 2025; Xu et al., 2026), linear attention (Chen et al., 2026b), quantization (Zhang et al., 2025a), and feature caching (Liu et al., 2025) techniques. Recently, several studies (Gao et al., 2025; Agarwal et al., 2026; Lv et al., 2026; Samuel et al., 2026; Ma et al., 2026; Xu et al., 2026; Ranganath et al., 2026) have attempted to tailor these acceleration techniques to the characteristics of AR video diffusion models. However, AR video diffusion models natively rely on *KV cache* for streaming autoregressive inference, and most of the above methods do not alleviate cache size or memory overhead. Although KV cache compression has been widely studied in LLMs (Xiao et al., 2023; Zhang et al., 2023; Xiao et al., 2024; Zeng et al., 2026) and has been explored in autoregressive image generation (Li et al., 2025; Qin et al., 2025), it remains largely unexplored in AR video diffusion models. To compress the KV cache of AR video diffusion models, ISPA (Fu et al., 2026) adopts layer-wise compression through parametric absorption. Relax Forcing Zhao et al. (2026a) retains sink, intermediate, and recent context for improved memory ability. Dummy Forcing (Guo et al., 2026) observes that a subset of attention heads concentrates primarily on the currently generated chunk and exploits the attention bias for compression. However, it lacks a detailed characterization of the attention patterns and functional roles of individual heads, and the aggressive compression leads to chunk jittering and degradation in motion dynamics. Unlike these methods, our approach starts from the functional roles of individual attention heads, empirically characterizes the attention patterns in AR diffusion, and performs hybrid compression tailored to their decoupled redundancy.

B CHUNK DISCONTINUITY METRIC

Definition of chunk discontinuity. In Section 2 and Section 4, we use *chunk discontinuity* to quantify transitions across chunks. To ensure fairness and validity in evaluation, we define chunk discontinuity as an intuitive metric. Specifically, for a video containing F frames and generated autoregressively in K chunks, we first compute the optical flow difference between every pair of adjacent frames using *Recurrent All-Pairs Field Transforms* (RAFT) (Teed & Deng, 2020), a popular deep network architecture for optical flow computation. Then, we divide the average of the Top-$(K-1)$ largest values by the overall mean. The metric is defined as:

$$\delta_t = \Delta\mathrm{RAFT}(I_t, I_{t+1}), \quad t = 1, \ldots, F-1 \tag{5}$$

$$\text{Chunk Disc.} = \frac{\mathrm{Sum}\left(\mathrm{Top}_{K-1}\left(\{\delta_t\}_{t=1}^{F-1}\right)\right)/(K-1)}{\mathrm{Sum}\left(\{\delta_t\}_{t=1}^{F-1}\right)/(F-1)} \tag{6}$$

Effectiveness of chunk discontinuity. Our metric is applicable to existing autoregressive diffusion models, which generate continuous videos without scene cut. Under this setting, for a video that is temporally smooth and continuous, the optical flow difference between adjacent frames should vary relatively uniformly, resulting in a uniformly lower metric value. Conversely, a low metric value also indicates a lower average peak difference, suggesting milder temporal variation in the video. In principle, dynamic degree of a video may affect the absolute values of the metric. However, this effect is relatively minor compared with the pronounced peaks caused by chunk jittering. In other words, the metric is suited for capturing abrupt changes.

In Figure 8, we provide an example of a video with a high metric value and poor chunk continuity. The local maxima can be seen to appear regularly at fixed-interval chunk boundaries, demonstrating that our metric effectively captures discontinuities across chunks.

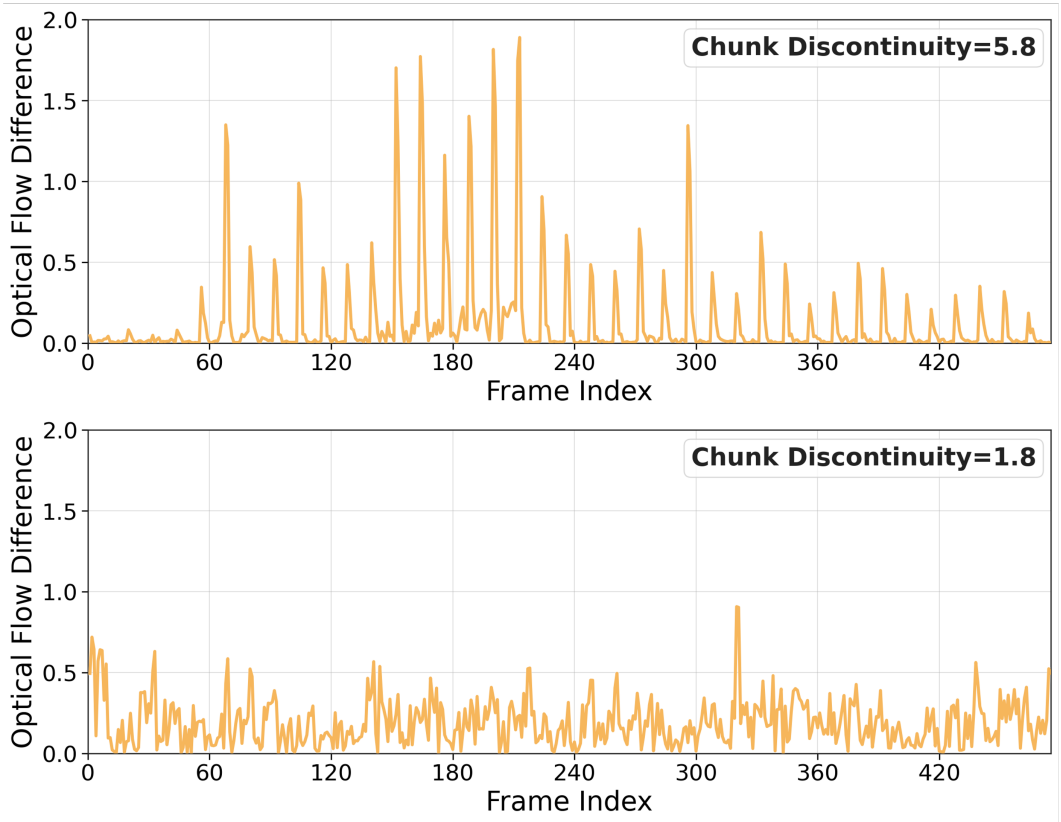

Figure 8: Case study of optical flow difference variations for a 30-second (~480 frames) video with high or low metric value.

C ATTENTION PATTERNS OF VARIOUS DIFFUSION MODELS

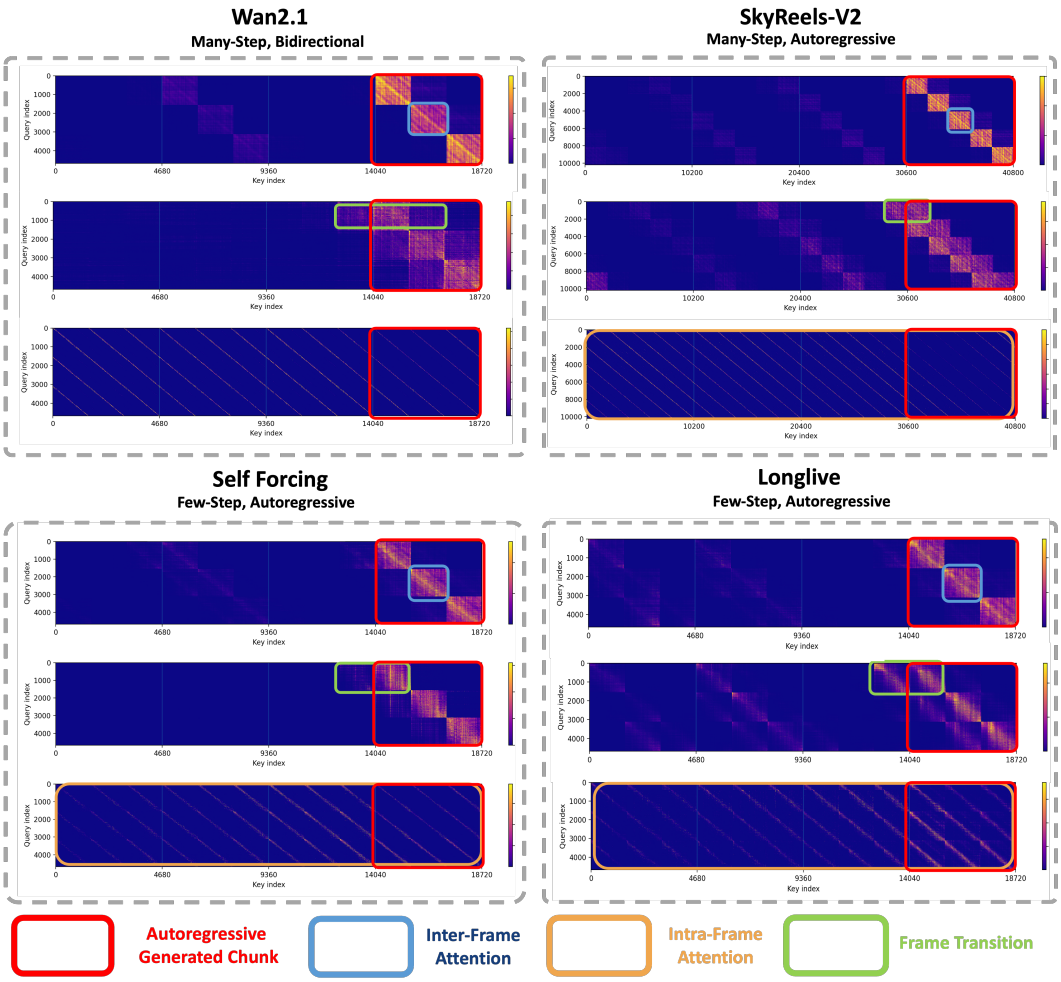

Figure 9: Attention patterns of Wan2.1 (Wan et al., 2025), SkyReels-V2 (Chen et al., 2025), Longlive (Yang et al., 2026), and Self Forcing (Huang et al., 2025a).

In this section, we conduct experiments across bidirectional and autoregressive video diffusion models and their multi-step and few-step variants, including Wan2.1, SkyReels-V2, LongLive, and Self Forcing, as shown in Figure 9. We find that the spatiotemporal functional specialization of attention heads is a common property across these models, where different heads are respectively responsible for inter-frame and intra-frame attention. An interesting observation is that **this property of autoregressive models may be inherited from the spatiotemporal attention heads of their bidirectional teachers**, given that mainstream autoregressive video models are typically initialized from bidirectional models.

Discussion (Autoregressive Properties): Prior studies on bidirectional models also identify spatial-temporal head patterns (Xi et al., 2025) for diffusion models. Our observations differ in two important aspects. First, we uncover a unique dependency on the transition anchor frame that is specific to autoregressive generation. In autoregressive diffusion, the generation follows a temporal Markovian chain, and the current chunk must access transition details from previous frames. Our visualization shows that such attention is densely concentrated on the transition anchor rather than uniformly across historical frames. Second, our observation is grounded in the KV cache, a distinctive structure of AR diffusion models. Unlike bidirectional models, autoregressive diffusion models do not generate all frames in a single pass and do not consume a static sparse attention kernel, making autoregressive KV cache compression a feasible design choice.

D User Study

Setup. To verify whether the quantitative benchmark results align with human perception, we conducted a user study with 12 participants. Each participant was presented with 15 video groups, where each group contained videos generated by different methods, including the base models (Self Forcing (Huang et al., 2025a) and LongLive (Yang et al., 2026)), Dummy Forcing (Guo et al., 2026), and FORCING-KV (ours). The videos covered 5-second and 30-second generations from Self Forcing and LongLive, and were randomly sampled from the full set of 946 videos in the VBench (Huang et al., 2024; 2025c) benchmark. To avoid positional bias, the videos were randomly arranged as *left*, *middle*, and *right*. In total, we collected 540 evaluations (12 participants × 15 video groups × 3 videos).

Evaluation protocol. Following the previous evaluation protocol (Lu et al., 2026), we ask the participants to evaluate each video according to three key criteria:

- **Visual Quality.** This criterion captures the overall quality and appeal of each video by jointly considering factors such as visual fidelity, coherence, motion quality, and the subjective viewing experience.
- **Dynamic Degree.** This criterion measures the naturalness, richness, and engagement of motions and changes in the video. Participants assess whether the generated content exhibits realistic and diverse dynamics, rather than static or repetitive patterns.
- **Consistency.** This criterion assesses whether a video maintains visual quality and coherence throughout its entire duration, without exhibiting visual drift, artifacts, or inconsistencies.

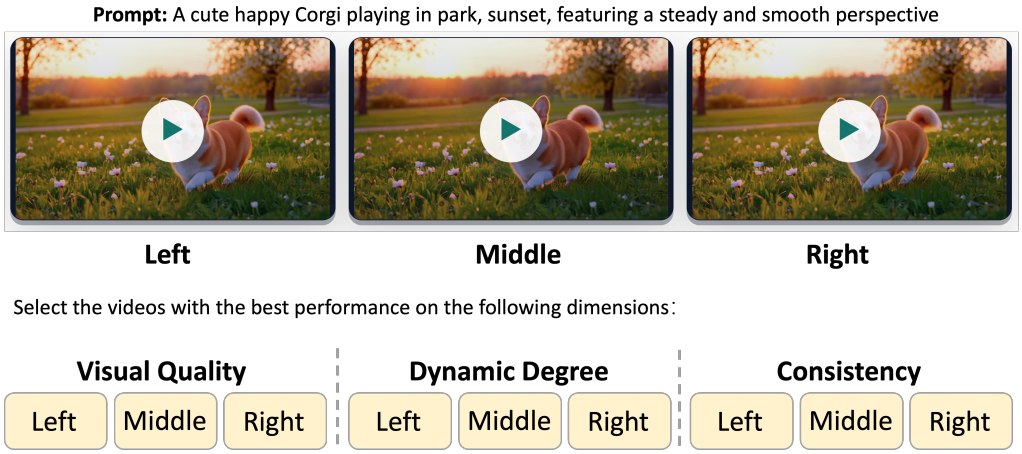

Figure 10: Protocol and screenshot of the user study.

E Results on More Base Models

We further evaluate our method on a more advanced model, Causal Forcing (Zhu et al., 2026), as well as a larger-scale model, Krea-Realtime-14B (Millon, 2025), to demonstrate the general applicability. For Causal Forcing, we use the official sliding window size of 21 latent frames. For Krea-Realtime-14B, we use the offline inference version without introducing online serving. The results in Table 4 demonstrate that our method remains effective on these two models, delivering meaningful speedups while largely maintaining the generation quality.

F Results on Interactive Video Generation

We further extend our method to interactive video generation by applying KV cache compression separately to the video segment associated with each prompt, and summarize the resulting quality and

Table 4: VBench results on 5-second video generation with Causal Forcing and Krea-Realtime-14B.

Method	Efficiency Metrics		Core Metrics		General Metrics		
	FPS↑	Speedup↑	Chunk Disc.↓	Dynamic Degree↑	Quality Score↑	Semantic Score↑	Total Score↑
Causal Forcing	22.26	1.00×	2.3	83.61	85.34	80.72	84.41
+FORCING-KV	28.17	**1.27**×	2.4	85.83	85.50	81.00	**84.60**
Krea-Realtime-14B	4.13	1.00×	1.7	77.78	85.02	81.57	84.33
+FORCING-KV	5.22	**1.26**×	1.9	73.61	85.35	81.55	**84.59**

Table 5: VBench-Long results on 60-second interactive video generation with Longlive.

Methods	FPS↑	Total↑	Imaging Quality↑	Subject Consistency↑	Background Consistency↑	Motion Smoothness↑	Dynamic Degree↑	Aesthetic Quality↑
Longlive	23.07	78.63	69.38	98.03	96.21	99.14	26.39	59.42
Dummy Forcing	25.74	78.04	69.42	96.30	95.01	98.57	30.28	59.80
FORCING-KV	**26.35**	**79.52**	70.13	97.51	95.89	98.89	36.11	60.58

efficiency in Table 5. Specifically, we use the interactive prompts provided by LongLive (Yang et al., 2026), where each prompt group contains 6 progressively evolving descriptions, each corresponding to a 10-second video segment. We evaluate the quality of the entire generated video using VBench-Long (Huang et al., 2025c). The results show that our method achieves improvements in both quality (79.52 vs. 78.63) and inference speed (26.35 fps vs. 23.07 fps).

Table 6: Quality and efficiency results with FP8 quantization.

Methods	Efficiency Metrics		Performance Metrics		
	FPS↑	Speedup↑	Chunk Disc.↓	Dynamic Degree↑	Total Score↑
LongLive - 60s					
FORCING-KV	26.71	1.30×	2.5	43.56	80.43
FORCING-KV + FP8	28.05	1.37×	2.5	43.27	80.41
Self Forcing - 30s					
FORCING-KV	26.65	1.50×	2.7	52.23	80.07
FORCING-KV + FP8	27.55	1.55×	2.8	52.15	80.06

G SEAMLESS INTEGRATION OF QUANTIZATION

We further incorporate FP8 quantization (Zhang et al., 2025a) into our KV cache compression method. This optimization reduces the computational cost of attention module by leveraging FP8 attention kernels tailored to the NVIDIA Hopper architecture, which further boosts throughput with minimal performance drop, as shown in Table 6. This shows that our method is compatible with other acceleration techniques.

H HYPERPARAMETER SENSITIVITY

We study the sensitivity of the two hyperparameter in FORCING-KV, number of segments n and compression ratio r in Tables 7 and 8. All experiments are evaluated on 128 Moviegen prompts with 1 random seed, applying FORCING-KV on Longlive.

Table 9: Ablation study on KV cache compression and head profiling strategies on LongLive - 5s.

Method	Chunk Disc.↓	Dynamic Degree↑	Total Score↑
Head Profiling			
FORCING-KV	2.1	45.56	83.22
Random Profiling	2.5	41.94	82.82
Human Profiling	2.2	47.22	83.31
KV Cache Compression			
FORCING-KV	2.1	45.56	83.22
w/o static-head cache	3.9	42.22	82.60
w/o dynamic-head cache	2.3	43.33	83.08

Table 10: Ablation study on KV cache compression and head profiling strategies on Self Forcing - 5s.

Method	Chunk Disc.↓	Dynamic Degree↑	Total Score↑
Head Profiling			
FORCING-KV	2.1	69.17	83.98
Random Profiling	3.0	59.72	82.76
Human Profiling	2.2	71.39	84.10
KV Cache Compression			
FORCING-KV	2.1	69.17	83.98
w/o static-head cache	3.4	65.28	83.73
w/o dynamic-head cache	2.8	62.50	83.56

For the number of segments n, $n = 6$ achieves the best VBench score (80.77), outperforming both $n = 3$ and larger segment numbers. This result justifies our choice of $n = 6$.

For the compression ratio r, reducing the compression ratio of dynamic heads slightly decreases the overall speedup, while gradually improving the motion dynamics. When $r = 1.0$, both the dynamic degree and VBench score drop significantly, demonstrating the effectiveness of dynamic token preservation.

Table 7: Ablation study on the number of segments n, evaluated on MovieGen prompts with 30-second videos using LongLive ($r = 0.6$).

Number of Segments	**VBench Score↑**
$n = 3$	80.71
$n = 6$	**80.77**
$n = 12$	80.63
$n = 30$	80.60

Table 8: Ablation study on the compression ratio r. Results are evaluated on MovieGen prompts with 30-second videos using LongLive ($n = 6$).

Compression Ratio	**VBench Score↑**	**Dynamic Degree↑**	**Speedup↑**
$r = 0.3$	80.78	45.72	1.23×
$r = 0.6$	80.77	44.98	1.27×
$r = 1.0$	80.09	39.42	1.34×

I MORE ABLATION STUDY RESULTS

In Section 4.3, we present the ablation results of LongLive on 30-second videos. For completeness, Tables 9 and 10 further report the ablation results on Self Forcing and LongLive for 5-second videos. The overall trends **remain consistent**. Regarding head profiling, our simple profiling strategy is already sufficient to closely approach manual profiling, while significantly outperforming the unprofiled random baseline. For KV cache compression, removing the transition anchor frame (*w/o static cache*) leads to severe chunk discontinuity, which further degrades the overall score. In contrast, removing the KV cache for dynamic heads (*w/o dynamic cache*) mainly causes a loss in dynamics, while having little effect on chunk continuity and the overall score. Since 5-second videos do not accumulate sufficient context, these effects are less pronounced than those observed on 30-second videos.

J COMPUTATION BREAKDOWN

As shown in Figure 11, FORCING-KV primarily accelerates the attention computation module, thereby reducing the overall DiT latency. The main overhead introduced by FORCING-KV comes from the gather operation and cache movement after each cache selection step, which takes 880.62 ms (4.5%). The similarity computation is very small because for each chunk, it is performed only at a single timestep and a proxy layer. Given the acceleration benefits, **the additional overhead is minor**.

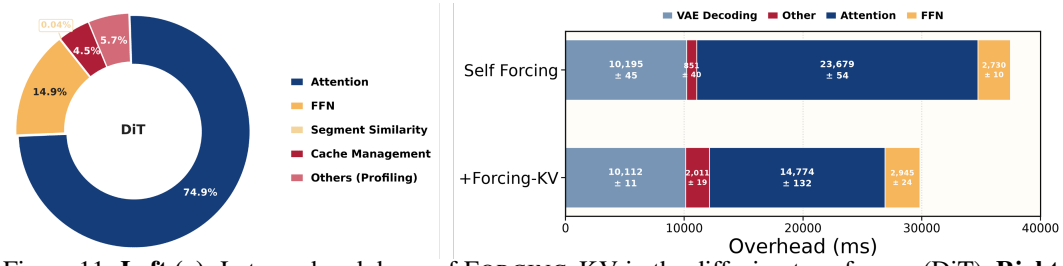

Figure 11: **Left (a)**: Latency breakdown of FORCING-KV in the diffusion transformer (DiT). **Right (b)**: End-to-end latency comparison. Results are evaluated on a single Nvidia H200 GPU.

We acknowledge that the end-to-end gain is smaller than the DiT-only speedup because VAE decoding is not accelerated. It is worth noting that, both our work and previous studies (Huang et al., 2025a; Yang et al., 2026; Lu et al., 2026) report the DiT inference time and do not include VAE decoding, mainly due to the following reasons: (i) VAE decoding is method-agnostic, both our method and previous studies (Huang et al., 2025a; Yang et al., 2026; Lu et al., 2026) specifically focus on efficiency gains in the DiT computation. (ii) Compared with DiT time, VAE decoding is not the primary bottleneck. In practical deployment, the VAE decoding overhead can often be hidden through **asynchronous execution**.

K IMPLEMENTATION DETAILS

Baselines. We mainly compare our method with KV cache compression approaches for large language models, StreamingLLM (Xiao et al., 2023), and with Dummy Forcing (Guo et al., 2026), a representative KV cache compression method for AR diffusion models. For StreamingLLM, we implement a naive frame-level strategy that retains 3 sink frames and 4 recent frames in the historical KV cache. For Dummy Forcing, following the original paper, we preserve the KV cache of the local region for local heads and adjust the history cache length L for neighbor heads. For an aggressive variant, we follow the official repository and set $L = 1$. For a conservative variant, we retain $L = 6$ for Self Forcing (Huang et al., 2025a) and $L = 2$ for LongLive (Yang et al., 2026), corresponding to their original window sizes. A larger value of L provides a broader visible context, which can improve performance at the cost of reduced speed.

Our Method. Our method consists of two components, head profiling and hybrid KV cache compression, corresponding to the hyperparameter α and the compression ratio r for dynamic heads, respectively. For head profiling, we perform offline head classification using a single prompt with $\alpha = 0.8$, and the entire procedure can be completed within a few minutes. For hybrid KV cache compression, we retain transition frames for static heads, while setting the default compression ratio for dynamic heads to $r = 0.6$. Specifically, each historical frame in the KV cache is uniformly divided into $n = 6$ contiguous segments, segment-wise similarity is computed between adjacent frames, and the retained segments are then determined according to the compression ratio r. To avoid the lack of adjacent frames at the beginning of generation, we apply KV cache compression starting from the second autoregressive step. We preserve the sink frame similar to Dummy Forcing (Guo et al., 2026). Notably, our method is designed in a simple yet effective way tailored to our empirical observations, and is fully compatible with modern attention kernels.

L VBENCH SCORES ACROSS ALL DIMENSIONS

We further provide the detailed 16-dimension VBench (Huang et al., 2024) scores of short videos in Table 2. Specifically, on individual metrics, FORCING-KV achieves improved performance over its base model in dynamic degree, overall consistency, appearance style, and total score with reduced KV cache budget. We attribute this result primarily to the fact that FORCING-KV preserves the context required by each type of head during inference, without sacrificing either local transition information or historical contextual information. Another possible factor is that existing autoregressive models may still be imperfect in their ability to utilize distant context effectively.

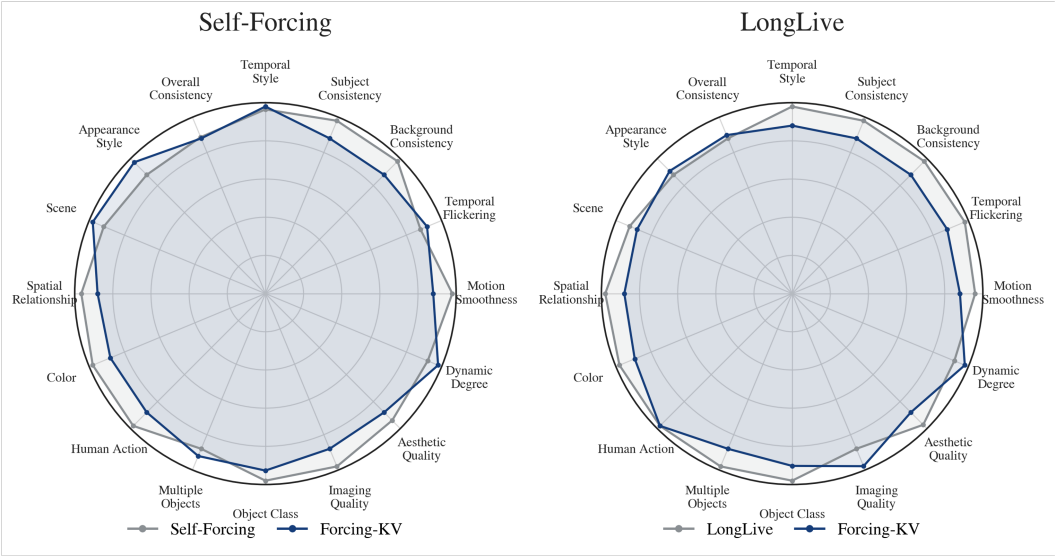

Figure 12: Visualization of VBench (Huang et al., 2024) scores. We compare FORCING-KV with its base model. FORCING-KV achieves near-lossless performance in terms of total score, and further outperforms the baseline on metrics such as dynamic degree, overall consistency, and appearance style, demonstrating its advantage.

M WHERE ARE STATIC AND DYNAMIC HEADS LOCATED?

We consider the head distribution in AR diffusion models to be an interesting research problem, and visualize the distribution of heads across layers in Figure 13. Overall, dynamic heads constitute approximately 60% of all heads, with a noticeably higher proportion in the middle layers (layer indices 13, 15, and 17). We hypothesize that this pattern arises because layers near the input and output focus more on extracting structured information to preserve local video quality, whereas the intermediate layers make heavier use of contextual information for rich feature refinement, improving detail consistency and temporal dynamics.

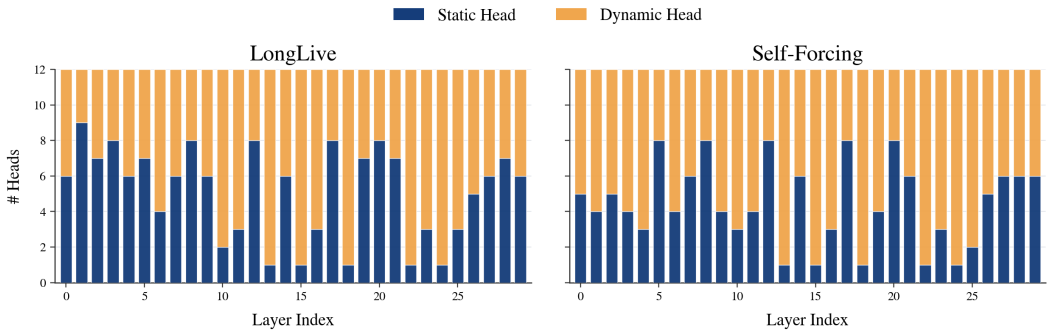

Figure 13: Head distribution across layers.

N TREND OF THE PROPORTION OF SELF-ATTENTION

In Figure 6, we show how the total runtime increases as the attention window size and resolution grow. As a complementary analysis, we further break down the proportion of time consumed by self-attention. As the sequence length grows, self-attention gradually occupies a larger share of the overall Transformer block (24% to 61% and 61% to 89%), making KV cache compression increasingly beneficial.

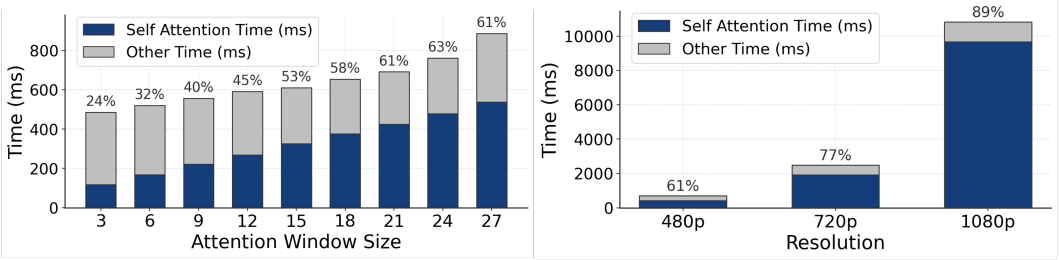

Figure 14: Variation in the proportion of total runtime occupied by self-attention.

O Proxy Choice for Similarity-Pruning

In FORCING-KV, we use a shallow proxy layer to determine the compressed content and then propagate it to the deeper layers. We justify this design from two perspectives: (i) why using proxy layer instead of full layer, and (ii) the correlation between the proxy layer and the deeper layers.

Adoption of a Proxy Layer. A primary reason for using a proxy layer is implementation efficiency. As shown in Appendix J, cache management operations (gathering and movement) constitute the major additional overhead of FORCING-KV. Repeating these operations at every layer would introduce substantial inefficiency to the attention kernel. In essence, FORCING-KV aims to estimate segment-level similarity, which primarily reflects the spatiotemporal characteristics of the video content itself, and exhibits less layer-wise variation.

Correlation of the Proxy Layer. To study the correlation of proxy layer and the deep layers, we compute the segment-wise similarity scores for all layers and heads by averaging the per-head cosine similarity scores at each segment position. We then calculate the *Pearson correlation coefficient* between the proxy layer and each deep layer and the per-head variance. The results are summarized in Table 11. Overall, the segment-wise similarity score of the proxy correlates well with the score distribution of deeper layers (0.783 and 0.946 Pearson correlation), and the attention heads generally exhibit low variation. The proxy layer captures the overall redundancy pattern across heads, which also enhances reliability.

Table 11: Correlation between proxy and deep layers, and head-wise feature similarity.

Indicator	Proxy to Each Deep Layer	Proxy to Avg. Deep Layer	Head Similarity	Head Std.
Value	0.783	0.946	0.822	0.085

P Theoretical Analysis of Applicability at High Resolutions

We first consider the query-wise full-attention computation in a Diffusion Transformer (DiT). Given a query token $\mathbf{q}_i \in \mathbb{R}^d$ and the key and value representations $\mathbf{K}, \mathbf{V} \in \mathbb{R}^{N\times d}$, where N denotes the number of visual tokens and d is the feature dimension, the attention output is computed as

$$\mathrm{Attn}(\mathbf{q}_i, \mathbf{K}, \mathbf{V}) = \mathrm{Softmax}\left(\frac{\mathbf{q}_i \mathbf{K}^\top}{\sqrt{d}}\right)\mathbf{V}. \tag{7}$$

FORCING-KV compresses the key-value pairs using head-wise strategies, dynamic similarity pruning π_{Dynamic} and static pruning π_{Static}. The corresponding attention computations are defined as

$$\mathrm{Attn}_{\mathrm{Dynamic}}(\mathbf{q}_i, \mathbf{K}, \mathbf{V}) = \mathrm{Softmax}\left(\frac{\mathbf{q}_i \left(\pi_{\mathrm{Dynamic}}(\mathbf{K})\right)^\top}{\sqrt{d}}\right)\pi_{\mathrm{Dynamic}}(\mathbf{V}), \tag{8}$$

$$\mathrm{Attn}_{\mathrm{Static}}(\mathbf{q}_i, \mathbf{K}, \mathbf{V}) = \mathrm{Softmax}\left(\frac{\mathbf{q}_i \left(\pi_{\mathrm{Static}}(\mathbf{K})\right)^\top}{\sqrt{d}}\right) \pi_{\mathrm{Static}}(\mathbf{V}). \tag{9}$$

In essence, both attention formulations can be viewed as approximations to full attention that determine the effectiveness of compression. For video generation models, increasing the resolution essentially increases the spatial size $H \times W$ of the latent representation within each frame. We posit:

Proposition (Applicability at High Resolution). *Given a well-trained base model and a compression strategy π, the effectiveness of π at the base model's native high resolution can be decomposed into whether π can be consistently applied to the enlarged spatial grid $H \times W$.*

In FORCING-KV, this applicability is naturally satisfied. For static pruning, π_{Static} retains the most recent transition anchor frame, which is independent of the spatial frame size. For dynamic similarity pruning, π_{Dynamic} performs pruning based on segment-wise similarity. When the latent resolution increases, the number of tokens contained in each segment can be proportionally increased such that each segment preserves the same relative spatial extent. Therefore, **FORCING-KV is not specific to a fixed spatial resolution and can be potentially extend to base model at higher resolutions**. In Figure 6, we validate the efficiency gains at high resolution, and quality performance at natively higher resolutions is left for future study due to the current absence of such models.

Q LIMITATION AND FUTURE WORKS

While our hybrid compression framework yields substantial gains in efficiency and performance for AR video diffusion models, it can be further improved in the following aspects. First, since mainstream open-source autoregressive video diffusion models are currently trained under the *Self Forcing* paradigm, our observations are primarily based on existing model families including Self Forcing (Huang et al., 2025a), Longlive (Yang et al., 2026), Causal Forcing (Zhu et al., 2026), SkyReels-V2 (Chen et al., 2025), Krea-Realtime-14B (Millon, 2025) and the broader family of "forcing" methods (Lu et al., 2026; Cui et al., 2026a; Yesiltepe et al., 2025; Cui et al., 2026b; Liu et al., 2026; Xiao et al., 2025). Nevertheless, we posit that our core observations are closely tied to the fundamental properties of autoregressive generation, and further evaluation on future native autoregressive models would be an interesting direction. Second, as a training-free method, investigating KV cache design during the training phase of autoregressive video models is also left for future work, which could potentially support a compact context window and enlarged memory horizon through fine-tuning.

R QUALITY EXAMPLES

We provide the quality example of FORCING-KV in Figures 15 to 17. More visualizations are provided in the demo videos.

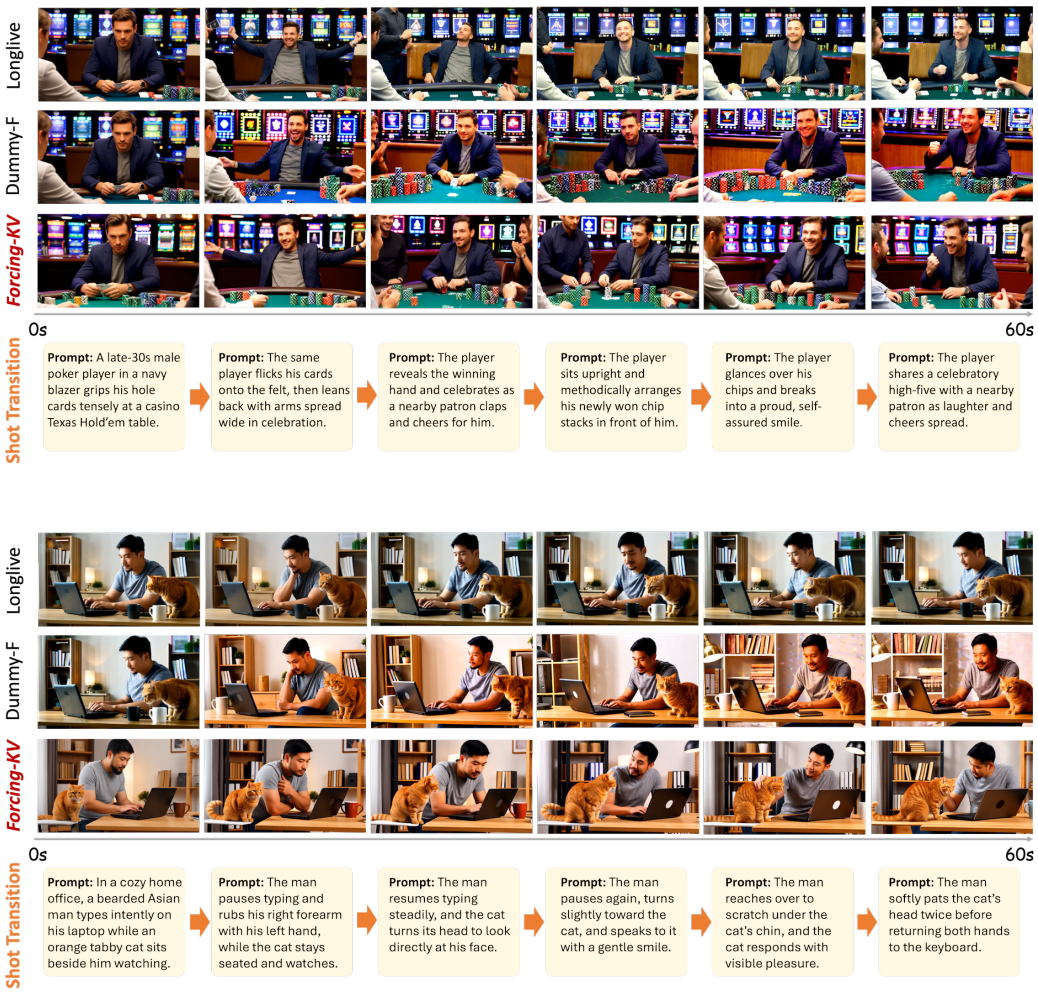

Figure 15: Quality example of 60-second interactive video on Longlive. More video demos are provided in the supplementary material.



Figure 16: Quality examples on Longlive 5s and 30s.



Figure 17: Quality examples on Self Forcing 5s and 30s.