Dynamics-Aware Sparse Attention for Efficient Autoregressive Video Diffusion

Yuanyu He[1,2*] **Zhuokun Chen**[3*] **Yefei He**[1,†] **Zhiwei Tang**[2]
Jiasheng Tang[2] **Yinghao Yu**[2] **Jianfei Cai**[3] **Bohan Zhuang**[1,‡]

[1]Zhejiang University [2]Alibaba Group [3]Monash University

Abstract

Autoregressive Diffusion Transformers have emerged as a powerful paradigm for long video generation. However, they are often constrained by prohibitive computational overhead and substantial memory footprints. This inefficiency arises from dense computation mechanisms that allocate uniform processing resources across all spatiotemporal regions, failing to leverage the inherent temporal redundancy of video where motion is typically confined to sparse areas. In this paper, we propose **Dynamics-Aware Sparse Attention (DASA)**, a unified framework that shifts from dense computation to sparse, motion-driven generation to eliminate both storage and computational redundancies. Inspired by video codec principles, our approach explicitly decouples video content into dynamic and static regions. To mitigate memory bottlenecks, we selectively compress the historical KV cache by retaining only dynamic features within the generated chunk. Furthermore, to reduce computational overhead, we employ a region-level computation allocation strategy based on localized dynamics. Additionally, a dynamics-guided distillation method is introduced to enhance performance via a lightweight distillation training process. Experiments on VBench demonstrate that our approach accelerates autoregressive video generation by 2.16×, while preserving high visual fidelity. Code is available on our project page: https://yuanyu0.github.io/DASA/.

1 Introduction

Video generation has witnessed unprecedented breakthroughs, propelled by the rapid progress of both proprietary [1–4] and open-source [5–8] foundational models. While these models demonstrate remarkable spatial-temporal fidelity, their outputs remain largely constrained to brief 5- to 20-second clips due to the quadratic computational complexity inherent in full-attention mechanisms. To achieve causally consistent, minute-level synthesis, the field is shifting toward Autoregressive (AR) Diffusion Models [9–13]. By factorizing the generation process into sequential chunks, AR diffusion offers a promising pathway toward infinite-length video synthesis.

Despite these compelling advantages, chunk-wise AR diffusion currently faces two severe efficiency bottlenecks in long video generation. First, prohibitive memory expansion. To maintain long-term causal consistency, AR video models typically retain the recent historical KV cache across chunks. For example, when generating a 1080P video with a sliding window manner [14], the KV cache can occupy 30GB of memory space, while the model weights only require 3GB. Moreover, the KV cache grows further as video resolution and duration increase. Second, severe latency stems from redundant computation. Current AR architectures blindly allocate identical dense compute resources to every token per chunk, despite most scenes having high temporal redundancy.

∗ Equal contribution. † Project Lead. ‡ Corresponding author: bohan.zhuang@gmail.com

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

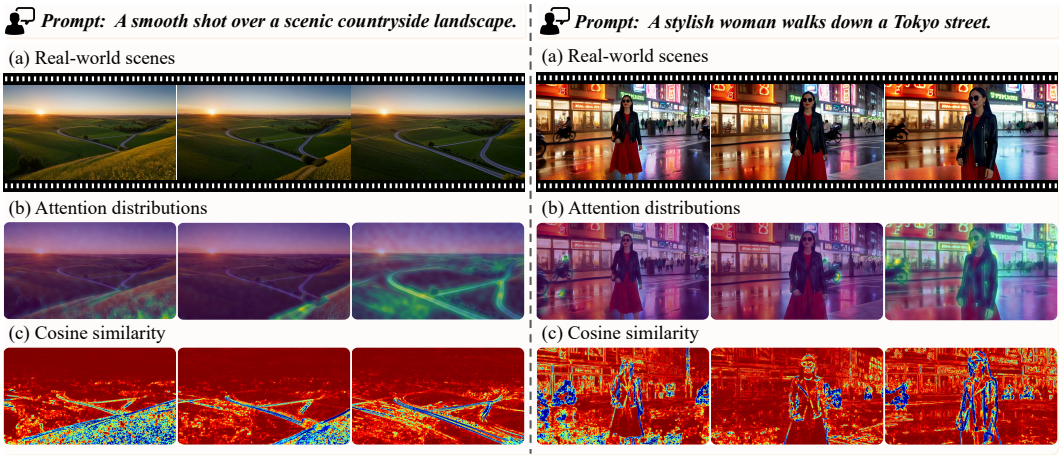

Figure 1: Spatiotemporal dynamic analysis of video generation under different motion priors (left: camera panning; right: subject motion). **(a) Real-world scenes.** The videos are synthesized via Self Forcing, conditioned on a single text prompt provided at the initialization stage. **(b) Attention distributions.** Visualizations reveal the distribution of temporal attention scores, indicating that the generation process predominantly focuses on active dynamic elements alongside adjacent temporal context. **(c) Cosine similarity** between the currently generated frame and preceding frames. The consistently high similarity observed in static background regions demonstrates the potential for extensive feature reuse during computation.

While recent acceleration methods [15–17] for AR diffusion have analyzed attention patterns to prune inactive tokens or heads, they often overlook the inherent nature of video—where most scenes exhibit high temporal redundancy and motion is confined to sparse regions. As shown in Figure 1 (a), real-world videos are dominated by static backgrounds and continuous dynamics, demonstrating massive spatiotemporal redundancy in adjacent frames. Addressing these intertwined memory and computational bottlenecks is therefore crucial for efficient AR-diffusion based video generation.

To address these challenges, we propose **Dynamics-Aware Sparse Attention (DASA)**, an efficient framework for AR video generation. Inspired by modern video coding principles [18, 19], we tackle both memory and computational bottlenecks through a unified dynamic and static separation perspective. From a storage standpoint, our analysis reveals that AR models primarily attend to historical dynamic elements and immediate temporal contexts (Figure 1 (b)). Based on this observation, we introduce a Frame-wise Memory Management module that employs layer-wise similarity pruning and a static token refresh strategy. This allows the framework to aggressively compress the KV cache while preserving essential high-frequency details. From a computational perspective, we implement a Region-level Computation Allocation strategy. By identifying static regions that exhibit minimal feature variation across chunks (Figure 1 (c)), we utilize an early-exit mechanism to bypass redundant forward passes, focusing resources exclusively on dynamic updates. By integrating these mechanisms, our framework curtails the KV cache memory footprint while reducing the computational overhead during chunk-wise video generation.

While these mechanisms are inherently training-free, pursuing higher acceleration ratios and framework scalability necessitates their integration into the training phase. However, directly exposing the network to static-feature reuse during training creates an optimization shortcut, resulting in motion sluggishness. To counteract this degradation, we introduce a Dynamics-guided Distillation strategy. By leveraging motion priors, we apply a spatially varying loss reweighting that explicitly amplifies the learning penalty in dynamic regions, successfully preserving the natural dynamics of the generated videos.

In summary, our main contributions are as follows:

- We identify that current autoregressive video diffusion suffers from inefficient resource allocation across highly redundant sequences. We observe that the generation process predominantly focuses

on active dynamic elements, while static backgrounds exhibit extensive feature redundancy across adjacent chunks.

- We propose Dynamics-Aware Sparse Attention (DASA) to eliminate spatiotemporal redundancy from both storage and computation perspectives. By integrating adaptive memory management, motion-aware computation skipping, and dynamics-guided distillation, DASA facilitates scalable video synthesis while avoiding static degradation.
- Extensive experiments demonstrate DASA's effectiveness in accelerating autoregressive video synthesis while preserving high visual fidelity, achieving a 2.16× inference speedup at 1080P with a minimal quality drop of less than 0.2%.

2 Related Work

2.1 Video Diffusion Models

The pursuit of ultra-long and interactive video generation [20–31] has been fundamentally shaped by the struggle against computational and memory bottlenecks. Early video diffusion models [5, 6, 32, 33] relied on full bidirectional attention to generate short clips, but the quadratic complexity of attention mechanisms imposed a strict ceiling on video length. To circumvent this, the community shifted toward autoregressive diffusion [34–37], which decomposed image-to-video generation into either frame-wise [38, 39] or chunk-wise [40] steps. Approaches like Causvid [41] apply DMD [42] to distill 50-step diffusion model into a 4-step generator, enabling faster streaming generation. However, scaling this autoregressive paradigm for long-term consistency requires caching extensive historical tokens, which inevitably triggers an unsustainable memory explosion. Consequently, recent models [14, 43] often employ a sliding window strategy as a trade-off to maintain a fixed-length memory. Simultaneously, to address the exposure bias in long rollouts, various training-inference alignment mechanisms were proposed, including Diffusion-Forcing [41], Self-Forcing [14], Rolling-Forcing [44], and Causal-Forcing [45]. Currently, this trajectory has culminated in interactive world models [9–12, 46]. For these world models, efficiency is no longer merely about generating faster, but rather a strict prerequisite to achieving real-time interaction , necessitating a fundamental rethinking of how foundational models allocate computational resources.

2.2 Efficient Autoregressive Video Generation

To meet the stringent efficiency demands of AR video generation, recent efforts have explored various acceleration strategies, which broadly fall into two directions. The first direction focuses on altering the generation paradigm or introducing auxiliary modules to reduce computational latency. For instance, MAGI-1 [40] breaks sequential bottlenecks by enabling the parallel generation of multiple video chunks with varying noise levels. Other works focus on context compression. For example, Framepack [38] compresses historical frames using fixed patterns based on frame-wise importance, while LoViC [47] introduces an autoencoder to compress historical contexts into compact representations.

The second direction accelerates inference by exploiting the model's intrinsic properties, predominantly through attention sparsification. Specifically, Dummy Forcing [15] prunes inactive attention heads during generation. Light Forcing [16] introduces a progressive, chunk-aware sparsity allocation and hierarchical sparse attention to capture information in a coarse-to-fine manner. Furthermore, Sparse Forcing [17] integrates sparsity into the training phase, motivated by the empirical observation that attention concentrates on salient visual blocks. Yet, it primarily optimizes the KV cache side by restricting attended heads or entries, still retaining all current-query tokens for computation. In contrast, our DASA directly skips updates for redundant current-query tokens through region-level early exit.

Fundamentally, while existing approaches primarily derive acceleration strategies from model-centric observations like attention distributions or structural priors [48–61], they often overlook the intrinsic physical nature of the video data itself. We argue that a more natural source of efficiency lies in the data-centric characteristics: video motion is inherently sparse and highly concentrated, whereas the background remains largely static.

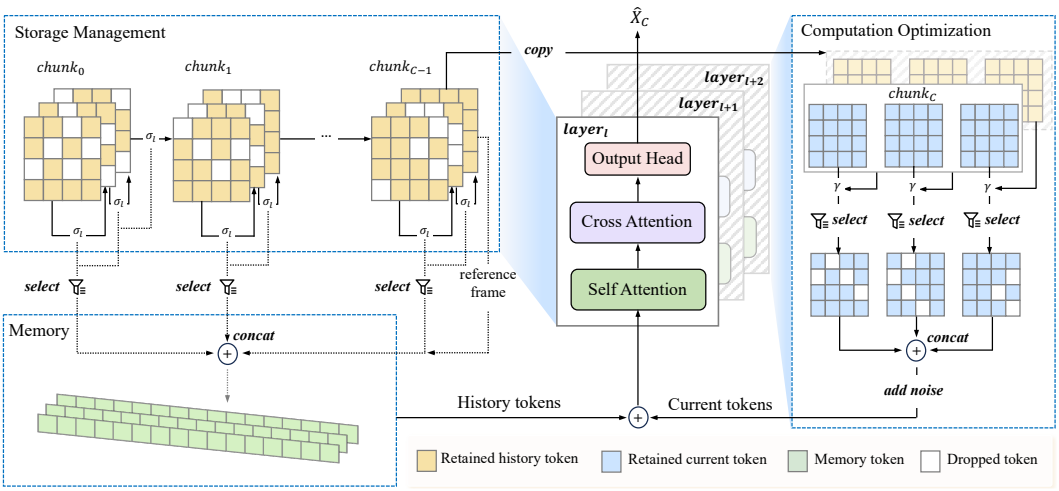

Figure 2: **Overall architecture of the proposed framework.** We introduce a unified approach to optimize AR diffusion generation for both storage and computational efficiency. **Storage Management (Left):** Tokens from historical chunks are selectively cached in memory based on a layer-wise threshold σ_l, discarding redundant information. The final frame of the immediately preceding chunk serves as a *reference frame*, with all its tokens unconditionally preserved. **Computation Optimization (Right):** This reference frame acts as an anchor to guide token selection for the current chunk$_C$. Redundant tokens are filtered out via a threshold γ to reduce computational overhead. **AR Diffusion Generation (Middle):** Finally, the retained historical tokens from memory and the active tokens from the current chunk are concatenated and processed by the AR diffusion.

3 Method

To address the redundancy inherent in long video generation, we present Dynamics-Aware Sparse Attention, a unified framework that jointly optimizes AR video generation from the perspectives of memory storage and computational allocation. The key idea is to exploit the temporal redundancy of video by explicitly distinguishing dynamic regions. We first introduce our Frame-wise Memory Management in Sec 3.1, which alleviates memory bottlenecks by selectively retaining dynamic features in the historical KV cache, together with a static refresh mechanism to maintain long-range consistency. We then describe Region-level Computation Allocation in Sec 3.2, which reduces redundant computation by assigning different processing budgets to static and dynamic regions within each generated chunk. Finally, in Sec 3.3, we incorporate these sparse generation strategies into training through Dynamics-Guided Distillation Adaptation, a lightweight distillation procedure that mitigates the potential static bias introduced by sparsification and improves robustness across generative settings. Figure 2 provides an overview of the proposed framework.

3.1 Frame-wise Memory Management

Due to the high spatial-temporal redundancy in continuous video streams, storing all tokens indiscriminately not only incurs severe memory footprint but is also fundamentally unnecessary. Current AR video models typically employ a sliding window approach [14, 41], sometimes paired with sink frames [43], to constrain memory by retaining only recent context. However, rigidly discarding older memory creates severe temporal inconsistencies in interactive world generation, especially when navigating back to past scenes. To overcome this limitation, rather than simply dropping old frames, we propose a Frame-wise Memory Management module, which dynamically regulates token caching through similarity-based pruning and a static token refresh strategy. We provide more algorithm details in Appendix A.1.

Layer-wise retention ratio. Motivated video coding standards like H.264, which utilize Intra-coded and Predictive-coded frames, we determine whether a token should be retained in memory based on its feature similarity with its adjacent frame. Specifically, let x_t^k denote the latent representation of the k-th spatial token at frame t. We compute the cosine similarity between temporally adjacent

tokens:

$$\text{sim}_t^k = \cos\left(\boldsymbol{x}_t^k, \boldsymbol{x}_{t+1}^k\right). \tag{1}$$

Unlike H.264, which uses the previous frame as a reference, our approach retains the last generated frame as the reference frame and save all its tokens. This strategy results in significantly smoother generated videos. We hypothesize that this entire frame serves as an anchor for the newly generated chunk, which aligns with our observations in Figure 1 (b). If a token's representation exhibits minimal changes compared to its latter frame, it is deemed redundant and can be safely discarded from the memory buffer. Thus, a preliminary retention mask is generated by thresholding the similarity: $M_{\text{sim}}^k = \mathbb{I}\left(\text{sim}_t^k < \sigma\right)$, where $\mathbb{I}(\cdot)$ is the indicator function and σ is the threshold.

As visualized in the Appendix D.1, we observe that the models attend to more dynamic tokens as the layer goes deeper. We assume that shallow layers capture high-frequency details that change rapidly, while deeper layers extract abstract, low-frequency semantic information that remains highly stable across frames. Leveraging this property, we apply a layer-wise retention rate by linearly decaying the threshold σ_l for layer l as the layer depth increases:

$$\sigma_l = \sigma_{max} - \frac{\sigma_{max} - \sigma_{min}}{L} \cdot l \tag{2}$$

where σ_{max} and σ_{min} are the predefined upper and lower bounds of the threshold, and L is the total number of layers. A lower threshold in deeper layers enforces a stricter retention condition, further compressing the memory footprint.

Static token refresh strategy. While similarity-based pruning significantly reduces redundancy, we observe temporal degradation in static regions as they are continuously pruned. If a background token remains similar across multiple frames and is repeatedly dropped, the model is forced to rely on a heavily outdated token. Consequently, the model fails to capture subtle temporal variations—such as global illumination shifts—leading to error accumulation that severely degrades overall visual fidelity.

To mitigate this, we introduce a Static Token Refresh strategy. We maintain an aging counter Δt^k for each spatial position k, defined as the number of frames elapsed since a token was last written to the memory at its location: $\Delta t^k = t - t_{\text{write}}^k$. If a token at a specific location has been continuously dropped for a predefined maximum age threshold τ, we force the model to retain this token. The final boolean mask M_{final}^k, which determines whether the k-th token is saved to the memory bank, is defined as the union of the similarity and aging conditions:

$$M_{\text{final}}^k = M_{\text{sim}}^k \vee \mathbb{I}\left(\Delta t^k \geq \tau\right). \tag{3}$$

When $M_{\text{final}}^k = 1$, the token is stored, and its timestamp t_{write}^k is reset to the current frame t. This mandatory refresh acts as a periodic calibration, effectively preventing the fading of high-frequency background details while keeping the overall memory growth well-controlled.

3.2 Region-level Computation Allocation

Unlike traditional video diffusion models [5, 6] that compute full spatial-temporal attention across the entire video sequence simultaneously, AR diffusion models adopt a chunk-wise generation paradigm. While this can efficiently scale to long video generation, we observe a significant computational bottleneck: the standard diffusion process iteratively denoises all tokens within a chunk, ignoring the strong inter-chunk temporal coherence. In reality, the current generating chunk shares substantial similarities with the preceding synthesized chunk, particularly in backgrounds and smoothly moving subjects. To effectively eliminate this redundant computation, we propose a *Region-level Computation Allocation* strategy that dynamically reuses previously computed features from the preceding chunk, thereby skipping unnecessary computations for highly redundant regions. More algorithm details are provided in Appendix A.2.

Motion-aware early exit. In the first denoising step of the current chunk C, the model predicts an estimated clean state $\hat{X}_C$. We leverage this intermediate $\hat{X}_C$ to evaluate temporal redundancy against the previously finalized chunk X_{C-1}. Since physical motion in videos is typically local and continuous, we perform a local alignment search rather than comparing strictly identical spatial coordinates.

Specifically, for a predicted token $\hat{\boldsymbol{x}}_C^{(i,j)}$ at spatial coordinate (i, j) in chunk C, we search for its best matching token within a local $w \times w$ spatial window $\mathcal{N}_w(i, j)$ in the previous chunk $C-1$. The aligned token $\tilde{\boldsymbol{x}}_{C-1}^{(i,j)}$ and its corresponding maximum similarity score $\text{sim}_C^{(i,j)}$ are defined as:

$$\tilde{\boldsymbol{x}}_{C-1}^{(i,j)} = \boldsymbol{x}_{C-1}^{(u^*,v^*)}, \quad \text{where} \quad (u^*, v^*) = \underset{(u,v)\in\mathcal{N}_w(i,j)}{\arg\max} \cos\left(\hat{\boldsymbol{x}}_C^{(i,j)}, \boldsymbol{x}_{C-1}^{(u,v)}\right), \tag{4}$$

and $\text{sim}_C^{(i,j)} = \cos\left(\hat{\boldsymbol{x}}_C^{(i,j)}, \tilde{\boldsymbol{x}}_{C-1}^{(i,j)}\right)$. This local search efficiently tracks moving objects, while static backgrounds naturally align at the exact same coordinate (i, j). Furthermore, by keeping the window size w as a small fixed constant, this local search avoids the quadratic $\mathcal{O}(L^2)$ complexity of a global matching process (where L is the spatial sequence length). Instead, it scales linearly with L, ensuring the additional alignment overhead remains minimal.

Based on the alignment results, a token is classified as redundant if its maximum similarity exceeds a predefined threshold γ. This yields a binary mask M:

$$M^{(i,j)} = \mathbb{I}\left(\text{sim}_C^{(i,j)} \geq \gamma\right). \tag{5}$$

For these redundant tokens, we halt their subsequent denoising iterations—triggering an early exit—and directly copy the corresponding aligned tokens from the previous chunk. Let $\tilde{X}_{C-1}$ denote the feature map composed of all aligned tokens. The output latent $\hat{X}_C^{\text{out}}$ is updated via spatial blending:

$$\hat{X}_C^{\text{out}} = M \odot \tilde{X}_{C-1} + (1 - M) \odot \mathcal{F}_\theta(\hat{X}_C) \tag{6}$$

where $\mathcal{F}_\theta(\cdot)$ denotes the rest of the denoising forward passes applied exclusively to the remaining unaligned (dynamic) tokens. This streamlined process securely bypasses heavy, unnecessary forward passes for visually redundant areas without compromising generation quality.

3.3 Dynamics-Guided Distillation Adaptation

While our inference technique efficiently accelerates generation and maintains overall quality, minor domain shifts or artifacts can still occur. To further elevate visual quality, we integrate the aforementioned mechanisms into the training phase. However, during this native adaptation, we find that while the spatial quality of the generated videos improves, they suffer from motion lag, causing actions and object movements to become noticeably restricted.

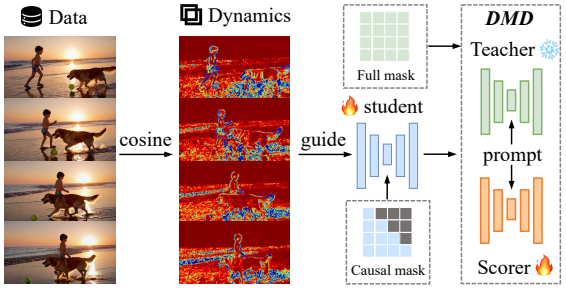

Figure 3: **Dynamics-guided Distribution Matching Distillation.** To mitigate static bias, we introduce a dynamic weight mask based on temporal cosine similarity. By amplifying loss in high-motion regions, it penalizes static predictions and forces the model to learn complex dynamics.

We attribute this degradation to an optimization shortcut. When the region-level static scene reuse mechanism is exposed during training, the network discovers a trivial solution to minimize the reconstruction error. Since predicting static or minimal changes yields a safely lower loss than taking the risk to generate complex motions, the model develops a "static bias." It over-relies on copying historical tokens and inadvertently suppresses its internal capability to generate dynamic motions.

To counteract this static bias without discarding the efficiency gains, we introduce a *Dynamics-Guided Distillation Adaptation* built upon the standard Distribution Matching Distillation [62] framework. We leverage the robust motion priors from the data sample to explicitly penalize static predictions. Specifically, let $\boldsymbol{z}_t^{(i,j)}$ denote the token representation of the sample video at frame t and spatial coordinate (i, j). We compute the temporal similarity $\text{sim}_t^{(i,j)}$ between adjacent frames using the previously defined cosine similarity. To identify areas of significant change and penalize "lazy" predictions, we map this similarity into a spatially-varying dynamic weight mask $W_t^{(i,j)}$:

$$W_t^{(i,j)} = 1 + \alpha \cdot \max\left(0, 1 - \text{sim}_t^{(i,j)}\right), \tag{7}$$

where α is a dynamic scaling factor.

Finally, this weight mask is applied token-wise to the standard DMD loss $\mathcal{L}_{\text{base}}$. The proposed dynamics-guided distillation loss $\mathcal{L}_{\text{dynamic}}$ is formulated as:

$$\mathcal{L}_{\text{dynamic}} = \frac{1}{N} \sum_{t,i,j \in \Omega} W_t^{(i,j)} \cdot \mathcal{L}_{\text{base}}^{(t,i,j)}, \tag{8}$$

where N denotes the total number of tokens. Regions exhibiting larger temporal variations in the sampled data naturally yield a lower similarity $\text{sim}_t^{(i,j)}$, thereby receiving a significantly amplified loss weight. This dynamic guidance acts as a strong penalty against the Student model's optimization shortcut, forcing it to faithfully reconstruct the real data's motion dynamics in the generated videos.

4 Experiments

4.1 Experimental Setup

Implementation details. We integrate our approach into representative baselines, including Self-Forcing [14] and LongLive [43], and further finetune them to enhance sample quality and generation speed. To comprehensively validate our method, our experimental scope spans multiple settings: standard 5s video generation, 30s long video generation, and high-resolution (720P and 1080P) video generation. All efficiency experiments are conducted on NVIDIA H100 GPUs. Due to page limit, we provide more implementation details in the Appendix B.

Evaluation. We evaluate our method on VBench [63], a comprehensive video generation benchmark. Specifically, we adopt the extended version of its 946 evaluation prompts introduced by [14]. Following standard practice, we sample 5 videos per prompt and assess the quality score and semantic score across all 16 quantitative dimensions defined in VBench (see Appendix C.3 for full per-dimension results). Additionally, we provide a detailed peak KV-cache memory profiling in Appendix C.1. All evaluation experiments are conducted on NVIDIA H20 GPUs.

Table 1: Quantitative results for **5-second short video generation** on VBench [63]. The best results are in **bold**, and the second best are underlined. The color box denotes the baseline model.

Type	Model	Params	Efficiency		Visual Quality		
			Throughput (FPS)↑	Speedup↑	Quality Score↑	Semantic Score↑	Total Score↑
Diffusion	LTX-Video [64]	1.9B	8.98	-	82.30	70.79	80.00
	CogVideoX [65]	5B	0.85	-	82.68	79.48	82.04
	Wan2.1 [66]	1.3B	0.78	-	85.30	80.09	84.26
	Open-Sora [32]	11B	0.08	-	84.28	79.89	83.40
	Hunyuan Video [6]	13B	0.22	-	85.07	76.88	83.43
Autoregressive	SkyReels-V2 [67]	1.3B	0.49	-	84.70	74.53	82.67
	MAGI-1 [40]	4.5B	0.19	-	82.04	67.74	79.18
	NOVA [68]	0.6B	0.88	-	80.39	79.05	80.12
	Pyramid Flow [69]	2B	6.77	-	84.74	69.62	81.72
	CausVid [41]	1.3B	0.69	-	83.73	78.49	82.69
	Self Forcing [14]	1.3B	18.42	1.00×	84.97	81.06	84.19
	TeaCache [70]	1.3B	21.69	1.18×	84.22	80.47	83.47
	Dummy Forcing [15]	1.3B	25.19	1.37×	84.47	81.08	83.79
	Sparse Forcing [17]	1.3B	18.80	1.02×	84.58	81.24	83.91
	Light Forcing [16]	1.3B	24.18	1.31×	84.34	80.70	83.61
	DASA	1.3B	26.21	1.42×	84.57	80.79	83.81
	DASA w/ adaptation	1.3B	**27.58**	**1.50**×	**84.64**	**81.32**	**83.98**
	LongLive [43]	1.3B	18.30	1.00×	83.77	80.73	83.16
	TeaCache [70]	1.3B	21.82	1.19×	83.10	80.73	82.63
	Dummy Forcing [15]	1.3B	25.10	1.37×	83.16	**80.99**	82.73
	Light Forcing [16]	1.3B	22.33	1.22×	83.44	80.28	82.81
	DASA	1.3B	26.50	1.45×	83.18	80.87	82.72
	DASA w/ adaptation	1.3B	**27.91**	**1.53**×	**83.78**	80.35	**83.09**

4.2 Main Results

4.2.1 Short Video Generation

This subsection evaluates our method on standard 5-second video generation using the VBench dataset, as presented in Table 1. We integrate DASA into state-of-the-art AR diffusion models

and compare it with TeaCache [70] and Dummy Forcing [15]. Since modern AR diffusion models typically use a low-step generation paradigm of 4 denoising steps, cross-step caching methods like TeaCache face inherent acceleration limits and yield a marginal speedup of 1.18×. While Dummy Forcing improves this by dropping attention heads based on the first generated chunk, its static pruning pattern may lead to information loss and degraded visual quality.

Unlike these baselines that rely on structural or step-wise approximations, our method exploits the inherent spatial-temporal redundancy of videos to dynamically allocate computational resources based on content complexity. By adaptively assigning more resources to highly dynamic regions while bypassing static areas, our adaptive pipeline achieves a speedup of 1.53×, consistently preserving the high visual and semantic quality. This further acceleration stems from fine-tuning under sparse feature reuse, which improves robustness to sparsity and permits a lower inference similarity threshold γ for higher token reuse. Furthermore, it is worth noting that our method optimizes AR diffusion models at a different level, making it compatible with TeaCache and Dummy Forcing (see Appendix C.2 for empirical verification).

Table 2: Quantitative results for **30-second long video** and **high-resolution video** generation on VBench [63].

Model	Efficiency		Visual Quality		
	Throughput (FPS)↑	Speedup↑	Quality Score↑	Semantic Score↑	Total Score↑
SkyReels-V2 [67]	0.49	-	80.77	53.37	75.29
FramePack [38]	0.92	-	83.61	75.32	81.95
30s Video Generation					
Self Forcing	18.42	1.00×	84.15	80.24	83.37
Dummy Forcing	25.19	1.37×	**84.11**	79.18	83.12
DASA	**26.21**	**1.42**×	84.05	**79.78**	**83.20**
LongLive	18.30	1.00×	83.57	80.96	83.05
Dummy Forcing	25.10	1.37×	83.23	79.59	82.50
DASA	**26.50**	**1.45**×	**83.39**	**80.59**	**82.83**
720P Video Generation					
Self Forcing	5.92	1.00×	84.30	81.21	83.68
Dummy Forcing	9.52	1.61×	84.16	80.86	83.50
DASA	**10.10**	**1.71**×	**84.40**	**80.92**	**83.71**
LongLive	5.98	1.00×	83.76	81.50	83.31
Dummy Forcing	9.74	1.63×	**83.49**	**81.63**	**83.12**
DASA	**10.37**	**1.73**×	83.38	81.54	83.01
1080P Video Generation					
Self Forcing	1.35	1.00×	84.12	79.42	83.18
Dummy Forcing	2.64	1.96×	83.10	**79.89**	82.46
DASA	**2.81**	**2.08**×	**83.55**	79.73	**82.79**
LongLive	1.36	1.00×	82.13	80.40	81.78
Dummy Forcing	2.74	2.02×	**81.92**	80.24	81.58
DASA	**2.94**	**2.16**×	81.90	**80.55**	**81.63**

4.2.2 Long Video Generation

Crucially, we highlight the scalability of our approach in 30-second long video generation, with quantitative results presented in Table 2. By maintaining contextual coherence over extended frames, DASA seamlessly scales to longer sequences. When integrated with baselines like Self Forcing and LongLive, our method achieves speedups of up to 1.45×, demonstrating its capability to accelerate extended generation without compromising overall semantic integrity.

4.2.3 High-resolution Video Generation

To further validate the robustness of our method under high computational demands, we extend our evaluation to 720P and 1080P high-resolution generation, as detailed in Table 2. Notably, the acceleration benefits of DASA scale with resolution. Our approach delivers speedups of up to 2.16× at 1080P. Despite massive reductions in memory and computational overhead, DASA successfully

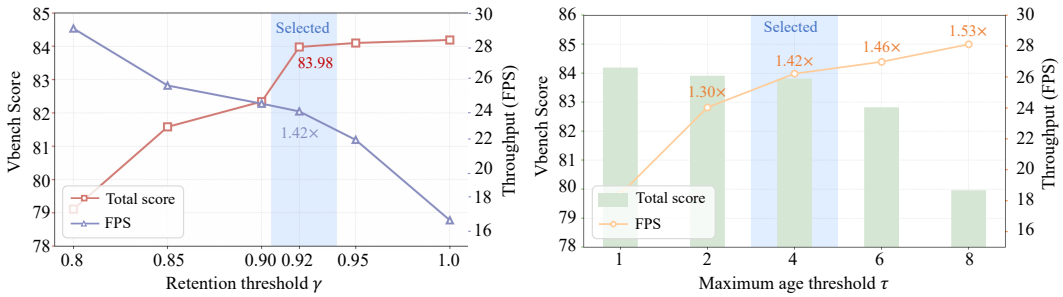

Figure 4: Ablation study on the impact of early-exit threshold γ and maximum age threshold τ.

preserves superior visual fidelity and fine spatial details, yielding good trade-off between throughput and total video quality.

4.3 Ablation Study

Effect of memory and computation retention. We evaluate two thresholds controlling spatial-temporal redundancy: memory retention σ and early-exit γ. To avoid a rigid speed-quality trade-off, we adopt a dynamic layer-wise σ (denser in shallow layers, sparser in deep ones), which successfully reduces memory footprint while preserving fidelity, as detailed in Table 3. Moreover, comparing decay schedules under identical endpoint thresholds in Table 5 reveals that exponential decay prunes deeper layers too aggressively and causes a substantial quality drop, whereas our linear decay achieves a superior speed-quality trade-off. For the Motion-Aware Early Exit, Figure 4 (a) demonstrates that γ dictates the speed-quality balance; an optimal $\gamma = 0.92$ achieves a $1.42\times$ speedup with a competitive Vbench total score of 83.98. In combination, these approaches bypass unnecessary memory operations and computational overhead.

Effect of static token refresh. As shown in the Figure 4 (b), increasing τ boosts throughput, but excessively large values ($\tau \geq 6$) severely degrade visual quality. This confirms our premise that unconstrained token dropping causes severe error accumulation. Conversely, the baseline ($\tau = 1$) preserves details but limits acceleration. We select $\tau = 4$ as the optimal balance, achieving a $1.42\times$ speedup with highly competitive visual fidelity.

Table 3: Ablation study on the impact of memory retention ratio σ.

σ		Throughput (FPS)↑	Quality Score↑	Semantic Score↑	Total Score↑
$\sigma_{\max}$	$\sigma_{\min}$				
Self Forcing		18.42	84.97	81.06	84.19
0.8	0.8	22.34 (1.21×)	84.89	81.00	84.11
0.6	0.6	24.11 (1.31×)	84.12	80.04	83.30
0.8	0.2	26.21 (1.42×)	84.57	80.79	83.81

Table 4: Ablation study on the impact of local alignment and adaptation.

	Throughput (FPS)↑	Quality Score↑	Semantic Score↑	Total Score↑
Self Forcing	18.42	84.97	81.06	84.19
w/ align.	26.21 (1.42×)	84.57	80.79	83.81
w/ adapt.	26.32 (1.43×)	84.70	81.24	84.01
ours	27.58 (1.50×)	84.64	81.32	83.98

Effect of local alignment. As shown in Table 4, by dynamically identifying and reusing redundant tokens via local window search, our motion-aware early exit bypasses unnecessary denoising tokens, yielding a significant $1.42\times$ throughput increase. When motion exceeds the local search range, unmatched tokens are not reused and safely undergo full denoising. We further ablate the local search window size w in Table 6 that enlarging w recovers more correspondences and boosts speedup, but an overly large window increases false reuse and degrades quality. We adopt $w = 3$ as the default trade-off.

Effect of dynamics-guided adaptation. Introducing the region-reuse mechanism during training can induce a "static bias," restricting motion generation. Our dynamics-guided adaptation explicitly penalizes these static predictions. As shown in Table 4, the dynamics-guided adaptation effectively mitigates the motion degradation, achieving the highest Semantic Score by preserving authentic dynamics. Furthermore, this harmonized training boosts the inference throughput to 27.58 FPS ($1.50\times$), validating that our design achieves a desirable balance between efficiency and visual fidelity.

Table 5: Ablation study on layer-wise decay schedules for memory retention σ.

Schedule	Speedup↑	Quality Score↑	Semantic Score↑
Self Forcing	1.0×	84.97	81.06
Constant ($\sigma = 0.8$)	1.21×	84.89	81.00
Linear (ours)	1.42×	84.57	80.79
Exponential	1.49×	81.16	76.34

Table 6: Ablation study on the local search window size w at 480P.

Window w	Speedup↑	Quality Score↑	Semantic Score↑
Self Forcing	1.0×	84.97	81.06
2	1.32×	84.85	80.37
3 (ours)	1.42×	84.57	80.79
5	1.50×	82.55	79.82

5 Conclusion

In this work, we explore the intrinsic dynamics and spatial-temporal redundancies in autoregressive long video generation. To exploit this nature for faster generation, we decompose this redundancy from two orthogonal perspectives: memory footprint and computational overhead. Specifically, we introduce a Frame-wise Memory Management module to dynamically prune temporally redundant tokens and strictly bound KV cache growth. Concurrently, we propose a Region-level Computation Allocation scheme that utilizes a motion-aware early exit mechanism to bypass unnecessary denoising computations for static backgrounds. Extensive experiments demonstrate that our token-level optimizations significantly accelerate video synthesis while maintaining high visual fidelity.

Limitations and future work. While our approach effectively reduces spatial-temporal redundancy at the *token* level, it currently does not explore redundancies across other architectural dimensions. Notably, our method is highly compatible with existing state-of-the-art acceleration techniques. For instance, TeaCache exploits redundancies across the *denoising step* dimension, whereas Dummy Forcing accelerates generation from the *attention head* dimension. In future work, we plan to systematically integrate our token-level strategy with step-wise TeaCache. We anticipate that unifying these complementary dimensions will unlock unprecedented synergistic acceleration ratios for autoregressive video generation.

Acknowledgments and Disclosure of Funding

This research was supported by the Zhejiang Provincial Natural Science Foundation of China under Grant No. LQN26F020010. The authors declare no competing interests.

References

[1] Tim Brooks, Bill Peebles, Connor Holmes, John DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, et al. Video generation models as world simulators, 2024. Accessed: 2026-05-02.

[2] Kuaishou Technology. Kling v2.5 turbo: Video generation model, 2025. Accessed: 2026-01-13.

[3] Team Seedance, Heyi Chen, Siyan Chen, Xin Chen, Yanfei Chen, Ying Chen, Zhuo Chen, Feng Cheng, Tianheng Cheng, Xinqi Cheng, et al. Seedance 1.5 pro: A native audio-visual joint generation foundation model. *arXiv preprint arXiv:2512.13507*, 2025.

[4] Team Seedance, De Chen, Liyang Chen, Xin Chen, Ying Chen, Zhuo Chen, Zhuowei Chen, Feng Cheng, Tianheng Cheng, Yufeng Cheng, et al. Seedance 2.0: Advancing video generation for world complexity. *arXiv preprint arXiv:2604.14148*, 2026.

[5] 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.

[6] Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.

[7] Bing Wu, Chang Zou, Changlin Li, Duojun Huang, Fang Yang, Hao Tan, Jack Peng, Jianbing Wu, Jiangfeng Xiong, Jie Jiang, et al. Hunyuanvideo 1.5 technical report. *arXiv preprint arXiv:2511.18870*, 2025.

[8] Bin Lin, Yunyang Ge, Xinhua Cheng, Zongjian Li, Bin Zhu, Shaodong Wang, Xianyi He, Yang Ye, Shenghai Yuan, Liuhan Chen, et al. Open-sora plan: Open-source large video generation model. *arXiv preprint arXiv:2412.00131*, 2024.

[9] Team HunyuanWorld. Hy-world 1.5: A systematic framework for interactive world modeling with real-time latency and geometric consistency. *arXiv preprint*, 2025.

[10] Robbyant Team, Zelin Gao, Qiuyu Wang, Yanhong Zeng, Jiapeng Zhu, Ka Leong Cheng, Yixuan Li, Hanlin Wang, Yinghao Xu, Shuailei Ma, Yihang Chen, Jie Liu, Yansong Cheng, Yao Yao, Jiayi Zhu, Yihao Meng, Kecheng Zheng, Qingyan Bai, Jingye Chen, Zehong Shen, Yue Yu, Xing Zhu, Yujun Shen, and Hao Ouyang. Advancing open-source world models. *arXiv preprint arXiv:2601.20540*, 2026.

[11] Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. In *ICML*, 2024.

[12] Zile Wang, Zexiang Liu, Jaixing Li, Kaichen Huang, Baixin Xu, Fei Kang, Mengyin An, Peiyu Wang, Biao Jiang, Yichen Wei, et al. Matrix-game 3.0: Real-time and streaming interactive world model with long-horizon memory. *arXiv preprint arXiv:2604.08995*, 2026.

[13] Desai Xie, Zhan Xu, Yicong Hong, Hao Tan, Difan Liu, Feng Liu, Arie Kaufman, and Yang Zhou. Progressive autoregressive video diffusion models. In *CVPR*, pages 6322–6332, 2025.

[14] Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *arXiv preprint arXiv:2506.08009*, 2025.

[15] 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.

[16] 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.

[17] 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.

[18] Haichao Zhang, Wenhao Chai, Shwai He, Ang Li, and Yun Fu. Dense video understanding with gated residual tokenization. *arXiv preprint arXiv:2509.14199*, 2025.

[19] 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.

[20] Longbin Ji, Xiaoxiong Liu, Junyuan Shang, Shuohuan Wang, Yu Sun, Hua Wu, and Haifeng Wang. Videoar: Autoregressive video generation via next-frame & scale prediction. *arXiv preprint arXiv:2601.05966*, 2026.

[21] Junchao Huang, Ziyang Ye, Xinting Hu, Tianyu He, Guiyu Zhang, Shaoshuai Shi, Jiang Bian, and Li Jiang. Live: Long-horizon interactive video world modeling. *arXiv preprint arXiv:2602.03747*, 2026.

[22] Roberto Henschel, Levon Khachatryan, Hayk Poghosyan, Daniil Hayrapetyan, Vahram Tadevosyan, Zhangyang Wang, Shant Navasardyan, and Humphrey Shi. Streamingt2v: Consistent, dynamic, and extendable long video generation from text. In *CVPR*, pages 2568–2577, 2025.

[23] Xinyuan Chen, Yaohui Wang, Lingjun Zhang, Shaobin Zhuang, Xin Ma, Jiashuo Yu, Yali Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. Seine: Short-to-long video diffusion model for generative transition and prediction. In *ICLR*, 2023.

[24] 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*, 2026.

[25] 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.

[26] 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, 2026.

[27] 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.

[28] 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. *arXiv preprint arXiv:2509.24695*, 2025.

[29] Yicong Hong, Yiqun Mei, Chongjian Ge, Yiran Xu, Yang Zhou, Sai Bi, Yannick Hold-Geoffroy, Mike Roberts, Matthew Fisher, Eli Shechtman, et al. Relic: Interactive video world model with long-horizon memory. *arXiv preprint arXiv:2512.04040*, 2025.

[30] Wuyang Li, Wentao Pan, Po-Chien Luan, Yang Gao, and Alexandre Alahi. Stable video infinity: Infinite-length video generation with error recycling. *arXiv preprint arXiv:2510.09212*, 2025.

[31] Inferix Team, Tianyu Feng, Yizeng Han, Jiahao He, Yuanyu He, Xi Lin, Teng Liu, Hanfeng Lu, Jiasheng Tang, Wei Wang, et al. Inferix: A block-diffusion based next-generation inference engine for world simulation. *arXiv preprint arXiv:2511.20714*, 2025.

[32] Zangwei Zheng, Xiangyu Peng, Tianji Yang, Chenhui Shen, Shenggui Li, Hongxin Liu, Yukun Zhou, Tianyi Li, and Yang You. Open-sora: Democratizing efficient video production for all. *arXiv preprint arXiv:2412.20404*, 2024.

[33] Wenyi Hong, Ming Ding, Wendi Zheng, Xinghan Liu, and Jie Tang. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. *arXiv preprint arXiv:2205.15868*, 2022.

[34] 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 *ICML*. PMLR, 2025.

[35] Yuchao Gu, weijia Mao, and Mike Zheng Shou. Long-context autoregressive video modeling with next-frame prediction. *arXiv preprint arXiv:2503.19325*, 2025.

[36] Shangjin Zhai, Zhichao Ye, Jialin Liu, Weijian Xie, Jiaqi Hu, Zhen Peng, Hua Xue, Danpeng Chen, Xiaomeng Wang, Lei Yang, et al. Stargen: A spatiotemporal autoregression framework with video diffusion model for scalable and controllable scene generation. In *CVPR*, pages 26822–26833, 2025.

[37] Mingzhen Sun, Weining Wang, Gen Li, Jiawei Liu, Jiahui Sun, Wanquan Feng, Shanshan Lao, SiYu Zhou, Qian He, and Jing Liu. Ar-diffusion: Asynchronous video generation with auto-regressive diffusion. In *CVPR*, pages 7364–7373, 2025.

[38] 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 *NeurIPS*, 2025.

[39] Wenming Weng, Ruoyu Feng, Yanhui Wang, Qi Dai, Chunyu Wang, Dacheng Yin, Zhiyuan Zhao, Kai Qiu, Jianmin Bao, Yuhui Yuan, et al. Art-v: Auto-regressive text-to-video generation with diffusion models. In *CVPR*, pages 7395–7405, 2024.

[40] Sand. ai, Hansi Teng, Hongyu Jia, Lei Sun, Lingzhi Li, Maolin Li, Mingqiu Tang, Shuai Han, Tianning Zhang, W. Q. Zhang, Weifeng Luo, Xiaoyang Kang, Yuchen Sun, Yue Cao, Yunpeng Huang, Yutong Lin, Yuxin Fang, Zewei Tao, Zheng Zhang, Zhongshu Wang, Zixun Liu, Dai Shi, Guoli Su, Hanwen Sun, Hong Pan, Jie Wang, Jiexin Sheng, Min Cui, Min Hu, Ming Yan, Shucheng Yin, Siran Zhang, Tingting Liu, Xianping Yin, Xiaoyu Yang, Xin Song, Xuan Hu, Yankai Zhang, and Yuqiao Li. Magi-1: Autoregressive video generation at scale, 2025.

[41] 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*, 2025.

[42] Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and William T Freeman. Improved distribution matching distillation for fast image synthesis. In *NeurIPS*, 2024.

[43] Shuai Yang, Wei Huang, Ruihang Chu, Yicheng Xiao, Yuyang Zhao, Xianbang Wang, Muyang Li, Enze Xie, Yingcong Chen, Yao Lu, and Song Hanand Yukang Chen. Longlive: Real-time interactive long video generation. 2025.

[44] Kunhao Liu, Wenbo Hu, Jiale Xu, Ying Shan, and Shijian Lu. Rolling forcing: Autoregressive long video diffusion in real time. *arXiv preprint arXiv:2509.25161*, 2025.

[45] 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.

[46] Kaiwen Zhang, Zhenyu Tang, Xiaotao Hu, Xingang Pan, Xiaoyang Guo, Yuan Liu, Jingwei Huang, Li Yuan, Qian Zhang, Xiao-Xiao Long, et al. Epona: Autoregressive diffusion world model for autonomous driving. In *ICCV*, pages 27220–27230, 2025.

[47] Jiaxiu Jiang, Wenbo Li, Jingjing Ren, Yuping Qiu, Yong Guo, Xiaogang Xu, Han Wu, and Wangmeng Zuo. Lovic: Efficient long video generation with context compression. *arXiv preprint arXiv:2507.12952*, 2025.

[48] Haocheng Xi, Shuo Yang, Yilong Zhao, Chenfeng Xu, Muyang Li, Xiuyu Li, Yujun Lin, Han Cai, Jintao Zhang, Dacheng Li, et al. Sparse videogen: Accelerating video diffusion transformers with spatial-temporal sparsity. *arXiv preprint arXiv:2502.01776*, 2025.

[49] 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. *arXiv preprint arXiv:2505.18875*, 2025.

[50] Ruyi Xu, Guangxuan Xiao, Haofeng Huang, Junxian Guo, and Song Han. Xattention: Block sparse attention with antidiagonal scoring. *arXiv preprint arXiv:2503.16428*, 2025.

[51] Yefei He, Yuanyu He, Shaoxuan He, Feng Chen, Hong Zhou, Kaipeng Zhang, and Bohan Zhuang. Neighboring autoregressive modeling for efficient visual generation. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 19000–19010. IEEE, 2025.

[52] Shaoxuan He, Benlei Cui, Bukun Huang, Zhizeng Ye, Yunyun Sun, Longtao Huang, Hui Xue, Yang Yang, Haiwen Hong, Jingqun Tang, et al. Tc-pade: Trajectory-consistent pade approximation for diffusion acceleration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 35768–35778, 2026.

[53] Zhiwei Tang, Yuanyu He, Yizheng Han, Wangbo Zhao, Jiasheng Tang, Fan Wang, and Bohan Zhuang. K-forcing: Joint next-k-token decoding via push-forward language modeling. *arXiv preprint arXiv:2606.10820*, 2026.

[54] Yifei Xia, Suhan Ling, Fangcheng Fu, Yujie Wang, Huixia Li, Xuefeng Xiao, and Bin Cui. Training-free and adaptive sparse attention for efficient long video generation. In *CVPR*, pages 15982–15993, 2025.

[55] Yuechen Zhang, Jinbo Xing, Bin Xia, Shaoteng Liu, Bohao Peng, Xin Tao, Pengfei Wan, Eric Lo, and Jiaya Jia. Training-free efficient video generation via dynamic token carving. *arXiv preprint arXiv:2505.16864*, 2025.

[56] Yefei He, Feng Chen, Yuanyu He, Shaoxuan He, Hong Zhou, Kaipeng Zhang, and Bohan Zhuang. Zipar: Parallel auto-regressive image generation through spatial locality. *arXiv preprint arXiv:2412.04062*, 2024.

[57] Akide Liu, Zeyu Zhang, Zhexin Li, Xuehai Bai, Yuanjie Xing, Yizeng Han, Jiasheng Tang, Jichao Wu, Mingyang Yang, Weihua Chen, et al. Fpsattention: Training-aware fp8 and sparsity co-design for fast video diffusion. *Advances in Neural Information Processing Systems*, 38:138786–138816, 2026.

[58] Zeyu Zhang, Shuning Chang, Yuanyu He, Yizeng Han, Jiasheng Tang, Fan Wang, and Bohan Zhuang. Blockvid: Block diffusion for high-quality and consistent minute-long video generation. *arXiv preprint arXiv:2511.22973*, 2025.

[59] Weijie Wang, Xiaoxuan He, Youping Gu, Yifan Yang, Zeyu Zhang, Yefei He, Yanbo Ding, Xirui Hu, Donny Y Chen, Zhiyuan He, et al. World-r1: Reinforcing 3d constraints for text-to-video generation. *arXiv preprint arXiv:2604.24764*, 2026.

[60] Jintao Zhang, Chendong Xiang, Haofeng Huang, Jia Wei, Haocheng Xi, Jun Zhu, and Jianfei Chen. Spargeattention: Accurate and training-free sparse attention accelerating any model inference. *arXiv preprint arXiv:2502.18137*, 2025.

[61] Peiyuan Zhang, Yongqi Chen, Runlong Su, Hangliang Ding, Ion Stoica, Zhengzhong Liu, and Hao Zhang. Fast video generation with sliding tile attention. *arXiv preprint arXiv:2502.04507*, 2025.

[62] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *CVPR*, pages 6613–6623, 2024.

[63] Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. Vbench: Comprehensive benchmark suite for video generative models. In *CVPR*, pages 21807–21818, 2024.

[64] Yoav HaCohen, Nisan Chiprut, Benny Brazowski, Daniel Shalem, Dudu Moshe, Eitan Richardson, Eran Levin, Guy Shiran, Nir Zabari, Ori Gordon, Poriya Panet, Sapir Weissbuch, Victor Kulikov, Yaki Bitterman, Zeev Melumian, and Ofir Bibi. Ltx-video: Realtime video latent diffusion. *arXiv preprint arXiv:2501.00103*, 2024.

[65] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024.

[66] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

[67] Guibin Chen, Dixuan Lin, Jiangping Yang, Chunze Lin, Junchen Zhu, Mingyuan Fan, Hao Zhang, Sheng Chen, Zheng Chen, Chengcheng Ma, Weiming Xiong, Wei Wang, Nuo Pang, Kang Kang, Zhiheng Xu, Yuzhe Jin, Yupeng Liang, Yubing Song, Peng Zhao, Boyuan Xu, Di Qiu, Debang Li, Zhengcong Fei, Yang Li, and Yahui Zhou. Skyreels-v2: Infinite-length film generative model, 2025.

[68] Haoge Deng, Ting Pan, Haiwen Diao, Zhengxiong Luo, Yufeng Cui, Huchuan Lu, Shiguang Shan, Yonggang Qi, and Xinlong Wang. Autoregressive video generation without vector quantization. *arXiv preprint arXiv:2412.14169*, 2024.

[69] Yang Jin, Zhicheng Sun, Ningyuan Li, Kun Xu, Kun Xu, Hao Jiang, Nan Zhuang, Quzhe Huang, Yang Song, Yadong Mu, and Zhouchen Lin. Pyramidal flow matching for efficient video generative modeling. 2024.

[70] 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 *CVPR*, pages 7353–7363, 2025.

A Algorithm Details

This part provides the precise algorithmic formulations of our proposed optimization framework, complementing the conceptual methodology described in the main text. To explicitly demonstrate how we exploit spatial-temporal redundancy for efficient long video generation, we detail the execution pipelines of our two core modules. First, Algorithm 1 formalizes the Frame-wise Memory Management strategy, illustrating the dynamic KV cache pruning and static refresh mechanisms. Second, Algorithm 2 outlines the Region-level Computation Allocation scheme, showing how the motion-aware early exit mechanism is practically implemented during the chunk-wise denoising loop.

A.1 Memory Management

Algorithm 1 presents the core computational workflow of our Frame-wise Memory Management module, which dynamically optimizes the KV cache for efficient long video generation. The algorithm executes the methodology described in the main text, primarily integrating layer-wise retention threshold decay, similarity-based temporal redundancy pruning, and a mandatory static token refresh strategy. By synergistically applying these mechanisms, our approach reduces memory footprint while effectively preventing the temporal degradation of background details, ensuring high overall visual fidelity.

Algorithm 1 Frame-wise Memory Management for Video Generation

Input: Latent tensor $X \in \mathbb{R}^{B \times N_f \times N_s \times D}$, Keys K, Values V
Input: Layer index l, Total layers L, Sim thresholds $\sigma_{\text{max}}, \sigma_{\text{min}}$, Max token age τ
Input: Reference frame X_{ref}, Token age tracker T_{update}, Current frame num t_{curr}
Output: Output tensor O, Updated KV Cache state

// 1. Layer-wise Retention Threshold Decay
$\sigma_l \leftarrow \sigma_{\text{max}} - \frac{\sigma_{\text{max}} - \sigma_{\text{min}}}{L} \cdot (l+1)$
// 2. Temporal Context Concatenation
if X_{ref} is not **null then**
    $X_{\text{comb}} \leftarrow \text{Concat}(X_{\text{ref}}, X, \text{dim} = 1)$
    $K_{\text{comb}} \leftarrow \text{Concat}(K_{\text{ref}}, K, \text{dim} = 1)$
    $V_{\text{comb}} \leftarrow \text{Concat}(V_{\text{ref}}, V, \text{dim} = 1)$
else
    $X_{\text{comb}} \leftarrow X, \quad K_{\text{comb}} \leftarrow K, \quad V_{\text{comb}} \leftarrow V$
end if
// 3. Similarity-based Token Pruning
$X_{\text{prev}} \leftarrow X_{\text{comb}}[:, :-N_{\text{ref}}, \dots]$
$X_{\text{next}} \leftarrow X_{\text{comb}}[:, 1:(-N_{\text{ref}}+1), \dots]$
$\text{sim} \leftarrow \text{CosineSimilarity}(X_{\text{prev}}, X_{\text{next}}, \text{dim} = -1)$
$M_{\text{sim}} \leftarrow \text{sim} < \sigma_l$ // Preliminary boolean mask based on similarity
// 4. Static Token Refresh Strategy
for each frame index $t \in \{1, \dots, N_f\}$ **do**
    $M_{\text{curr}} \leftarrow M_{\text{sim}}[:, t]$
    $\Delta t \leftarrow t_{\text{curr}} - T_{\text{update}}$ // Calculate token age
    $M_{\text{age}} \leftarrow \Delta t \geq \tau$
    $M_{\text{final}}[:, t] \leftarrow M_{\text{curr}} \vee M_{\text{age}}$ // Force refresh if token is too old
    $T_{\text{update}}[M_{\text{final}}[:, t]] \leftarrow t_{\text{curr}}$ // Reset aging counter for retained tokens
    $t_{\text{curr}} \leftarrow t_{\text{curr}} + 1$
end for
// 5. Reference Frame Preservation & Cache Update
$M_{\text{tail}} \leftarrow \mathbf{1} \in \{0,1\}^{B \times N_{\text{ref}} \times N_s}$ // Unconditionally retain the last frame(s)
$M_{\text{all}} \leftarrow \text{Concat}(M_{\text{final}}, M_{\text{tail}}, \text{dim} = 1)$
$K_{\text{save}} \leftarrow K_{\text{comb}}[M_{\text{all}}], \quad V_{\text{save}} \leftarrow V_{\text{comb}}[M_{\text{all}}]$ // Extract retained tokens via masking
$\text{KVCache.Append}(K_{\text{save}}, V_{\text{save}})$
$X_{\text{ref}} \leftarrow X[:, -N_{\text{ref}}:, \dots]$ // Update reference frame for the next chunk
// 6. Standard Attention Computation with Compact Cache
$O \leftarrow \text{Attention}(Q, \text{KVCache}_K, \text{KVCache}_V)$
return O

A.2 Computation Allocation

Algorithm 2 details the execution pipeline of our Region-level Computation Allocation strategy, specifically highlighting the Motion-Aware Early Exit mechanism. Operating within a chunk-wise generation framework, the algorithm leverages the first denoising step to perform a local spatial-temporal alignment between the current chunk and the preceding finalized chunk. By extracting a binary mask via thresholding and spatial dilation, it accurately segregates dynamic tokens from static or redundant backgrounds. In all subsequent denoising iterations, redundant tokens securely bypass the heavy forward passes (early exit) and are directly blended using cached historical features, thereby significantly eliminating redundant computations while preserving temporal coherence.

Algorithm 2 Region-level Computation Allocation (Motion-Aware Early Exit)

Input: Noisy latent chunk Z_C, Previous clean chunk X_{C-1}
Input: Denoising steps $\{t_0, t_1, \ldots, t_{S-1}\}$, Search window size w
Input: Similarity threshold γ, Morphological dilation kernel size k
Output: Denoised output chunk X_C

$M_{\text{dyn}} \leftarrow \mathbf{1}$ // Initialize mask to compute all tokens at the first step
$\tilde{X}_{C-1} \leftarrow$ null
// 1. Spatial Denoising Iterations for Current Chunk C
for each step index $s \in \{0, 1, \ldots, S-1\}$ **do**
  // Execute sparse Generator forward pass for dynamic tokens
  $\hat{X}_C \leftarrow \text{Generator}(Z_C, \text{timestep} = t_s, \text{active_mask} = M_{\text{dyn}})$
  // 2. Local Alignment & Mask Generation (Executed only at step 0)
  if $s = 0$ **and** X_{C-1} is not **null then**
    // Local motion search within $w \times w$ window to track objects
    $\tilde{X}_{C-1}, S_{\text{sim}} \leftarrow \text{LocalNxNMatch}(X_{C-1}, \hat{X}_C, \text{range} = w)$
    // Determine dynamic tokens via max-pooling & thresholding
    $M_{\text{dyn}} \leftarrow \text{MinPool2D}(S_{\text{sim}}) < \gamma$
    $M_{\text{dyn}} \leftarrow \text{Dilate}(M_{\text{dyn}}, \text{kernel} = k)$ // Spatial dilation for dynamic region
    $M_{\text{dyn}} \leftarrow \text{Upsample}(M_{\text{dyn}})$ // Expand to match original sequence length
    $M_{\text{static}} \leftarrow \neg M_{\text{dyn}}$ // Mask for visually redundant regions
  end if
  // 3. Spatial Blending & Computation Early Exit
  if $s > 0$ **and** M_{static} is available **then**
    // Bypass computation for static areas, copy from aligned history
    $\hat{X}_C \leftarrow M_{\text{static}} \odot \tilde{X}_{C-1} + M_{\text{dyn}} \odot \hat{X}_C$
  end if
  // 4. Prepare Latent for Next Denoising Step
  if $s < S - 1$ **then**
    $Z_C \leftarrow \text{Scheduler.AddNoise}(\hat{X}_C, t_{s+1})$
  end if
end for
// Return the fully denoised chunk
$X_C \leftarrow \hat{X}_C$
return X_C

B More Implementation Details

Setup for frame-wise memory management implementation. Here, we provide the specific implementation details of the proposed Frame-wise Memory Management module. For the layer-wise retention ratio, the similarity threshold decays linearly from an upper bound σ_{max} to a lower bound σ_{min} across the transformer layers. During the training phase, to enhance the model's robustness and prevent overfitting to strictly static backgrounds, we introduce a stochastic retention mechanism where naturally dropped tokens have a 5% probability of being forcefully retained. For the static token refresh strategy, a token is unconditionally revived if its continuous dropping age exceeds a predefined maximum token age threshold. Furthermore, to ensure seamless temporal transition across

autoregressive chunks, we unconditionally preserve the exact last frame of the previously generated chunk to serve as the structural anchor for the current chunk.

Setup for computation allocation implementation. In this section, we detail the engineering setup for the Region-level Computation Allocation and Motion-Aware Early Exit mechanism. During the local alignment search between adjacent chunks, we constrain the spatial search window to a 3×3 grid to balance matching accuracy and computational overhead. To distinguish redundant tokens, we apply a strict cosine similarity threshold of $\gamma = 0.92$ by default, while adopting a more aggressive threshold of $\gamma = 0.85$ during and after Dynamics-guided Adaptation due to improved robustness to sparsity. To conservatively encompass moving subjects and prevent visual artifacts at motion boundaries, the resulting dynamic binary mask undergoes morphological spatial dilation using a 3×3 kernel (with stride 1 and padding 1). Crucially, to maximize inference efficiency, this motion search and mask generation process is strictly executed *only once* during the first denoising timestep. The acquired mask is then kept fixed for all subsequent denoising steps within the current chunk. Finally, after the spatial denoising loop concludes, an additional forward pass infused with clean context noise is executed to reliably update the KV cache for the upcoming autoregressive chunk.

C More Quantitative Results

C.1 Peak Memory Profiling

To verify the actual memory savings of our method, we directly profile the peak KV-cache memory footprint of Self Forcing [14] and our DASA using a batch size of 1 across various video durations and resolutions. Crucially, DASA physically evicts pruned KV entries rather than merely applying attention masks over a dense cache. As shown in Table 7, DASA substantially reduces the peak KV-cache memory by **44.2%–54.1%** across different settings. Notably, under the memory-intensive 1080P setting, DASA cuts the KV-cache memory from 32.42 GB down to 15.13 GB, while consistently maintaining comparable VBench total scores.

Table 7: **Peak KV-cache memory footprint and VBench performance comparison** between Self Forcing and DASA. DASA achieves significant physical memory reduction while preserving generation quality.

Setting	Self Forcing	DASA (Ours)	Memory Reduction	VBench Total (Self Forcing / DASA)
5s, 480P	7.62 GB	3.50 GB	54.1%	84.19 / 83.81
30s, 480P	7.74 GB	4.32 GB	44.2%	83.37 / 83.20
5s, 720P	13.98 GB	7.74 GB	44.6%	83.68 / 83.71
5s, 1080P	32.42 GB	15.13 GB	53.3%	83.18 / 82.79

C.2 Compatibility Experiments

To demonstrate the orthogonality of our approach with existing acceleration techniques, we integrate DASA with Dummy Forcing [15] on the Self Forcing baseline at 480P resolution. While DASA reduces redundant spatial-token computation, Dummy Forcing operates along a different dimension by pruning redundant attention heads. As shown in Table 8, their combination achieves higher throughput than either method alone, yielding a **1.59**$\times$ speedup compared to 1.36$\times$ for Dummy Forcing and 1.42$\times$ for DASA alone, while maintaining comparable VBench generation quality. This result confirms that DASA and head-level pruning are effectively complementary at the token and head levels, respectively.

C.3 Per-Dimension VBench Evaluation

To provide a more fine-grained analysis of generation quality, we report the full per-dimension VBench scores across all 16 evaluation dimensions in Table 9. The results reveal an explicit trade-off that our adapted sparse-inference setting (*DASA + Adaptation*) noticeably improves motion- and temporal-related dimensions—particularly *Dynamic Degree* (70.12 vs. 68.89) and *Temporal Style* (26.02 vs. 24.50)—even surpassing the dense baseline, while only slightly reducing static frame metrics such as *Imaging Quality* and *Aesthetic Quality*.

Table 8: **Compatibility analysis of integrating DASA with Dummy Forcing** on Self Forcing at 480P. Combining token-level and head-level optimizations yields compounded speedup while preserving generation quality.

Method	FPS ↑	Speedup ↑	Quality Score ↑	Semantic Score ↑
Self Forcing	18.46	1.00×	84.97	81.06
Dummy Forcing	25.18	1.36×	84.47	81.08
DASA (Ours)	26.21	1.42×	84.57	80.79
DASA + Dummy Forcing	**29.36**	**1.59**×	84.51	80.36

Table 9: **Full per-dimension VBench evaluation results.** We compare the dense baseline (Self Forcing), training-free DASA, and DASA with Dynamics-guided Adaptation across all 16 dimensions.

VBench Dimension	Self Forcing	DASA	DASA + Adaptation
Quality Dimensions			
Subject Consistency	95.62	95.31	95.40
Background Consistency	96.34	96.08	96.15
Temporal Flickering	99.04	98.62	98.74
Motion Smoothness	98.35	98.34	**98.44**
Dynamic Degree	68.89	68.63	**70.12**
Imaging Quality	69.61	69.30	68.66
Aesthetic Quality	65.72	65.42	64.96
Semantic Dimensions			
Object Class	95.30	94.91	95.02
Multiple Objects	87.38	86.92	87.14
Human Action	97.20	96.71	**97.46**
Color	88.62	88.21	88.35
Spatial Relationship	81.90	81.56	**82.07**
Scene	54.93	54.70	54.71
Appearance Style	20.43	20.42	**20.65**
Temporal Style	24.50	24.49	**25.31**
Overall Consistency	26.80	26.76	26.77
Aggregated Scores			
Quality Score	84.97	84.57	84.64
Semantic Score	81.06	80.79	**81.32**
Total Score	84.19	83.81	83.98

D More Qualitative Results

D.1 Attention Distributions

To validate the motivation behind our layer-wise token retention strategy, we investigate how the model allocates its spatial attention at different stages of generation. To illustrate this progression, we extract and normalize the attention weights from layers 5, 15, and 25, utilizing them as representative examples of shallow, intermediate, and deep network layers, respectively. This visualization helps to confirm our hypothesis regarding the network's transition from capturing fine-grained details to abstract semantic concepts.

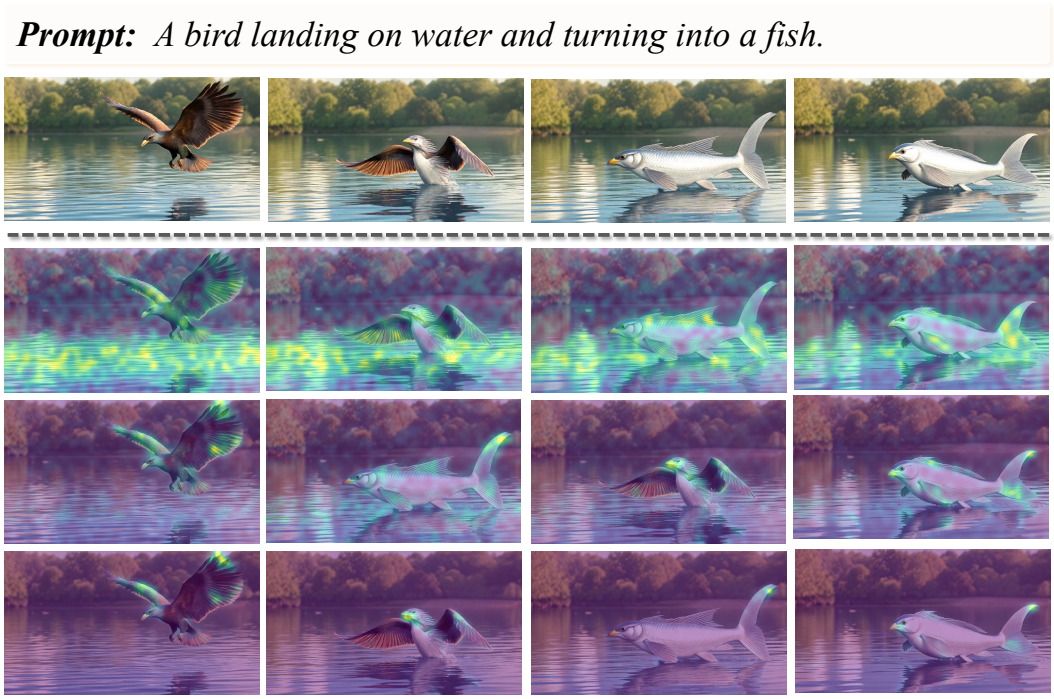

Figure 5: Attention distribution in layers 5, 15, and 25 of LongLive.



Figure 6: Attention distribution in layers 5, 15, and 25 of LongLive.

Prompt: *A a sleek bullet train speeding through a dense rainforest.*

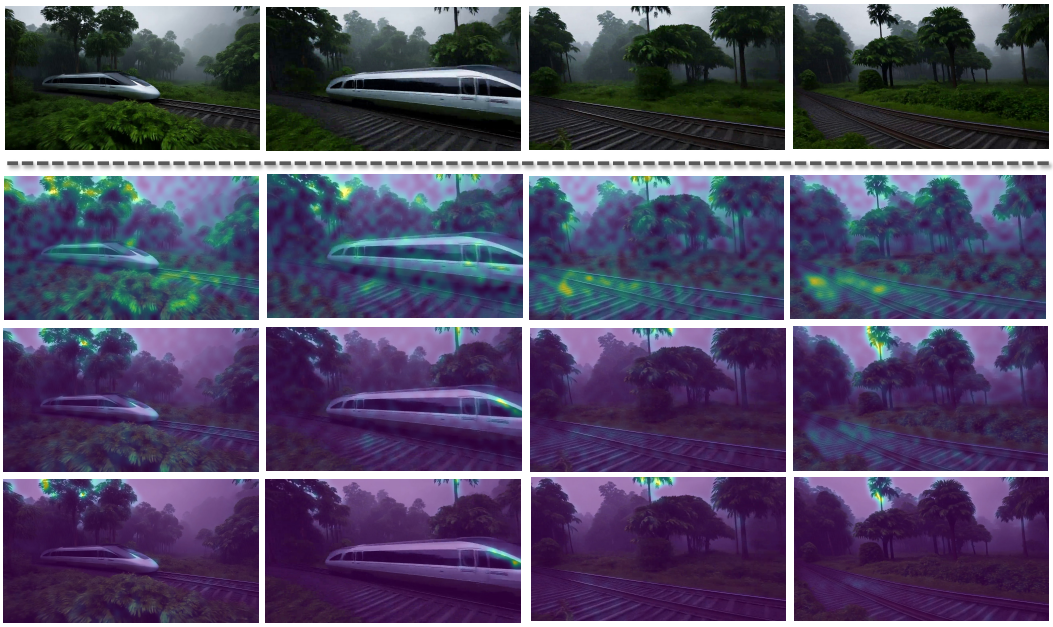

Figure 7: Attention distribution in layers 5, 15, and 25 of LongLive.

Prompt: *A sea creature with translucent, iridescent blue-green skin.*

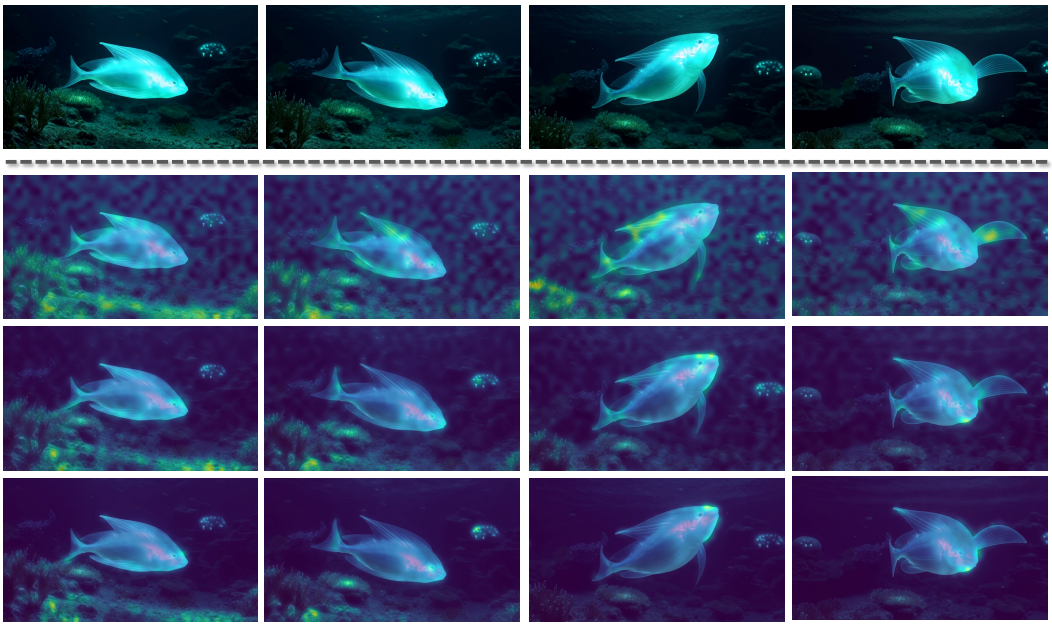

Figure 8: Attention distribution in layers 5, 15, and 25 of LongLive.

D.2 Visualization

The following qualitative comparison demonstrates that our DASA framework generates video sequences that are visually nearly identical to the Self Forcing. This visual consistency across diverse dynamic scenarios—including fluid human actions, complex camera panning, detailed static backgrounds, and fast-moving vehicles—validates that our spatiotemporal redundancy reduction strategy achieves high generation quality while providing substantial computational acceleration.

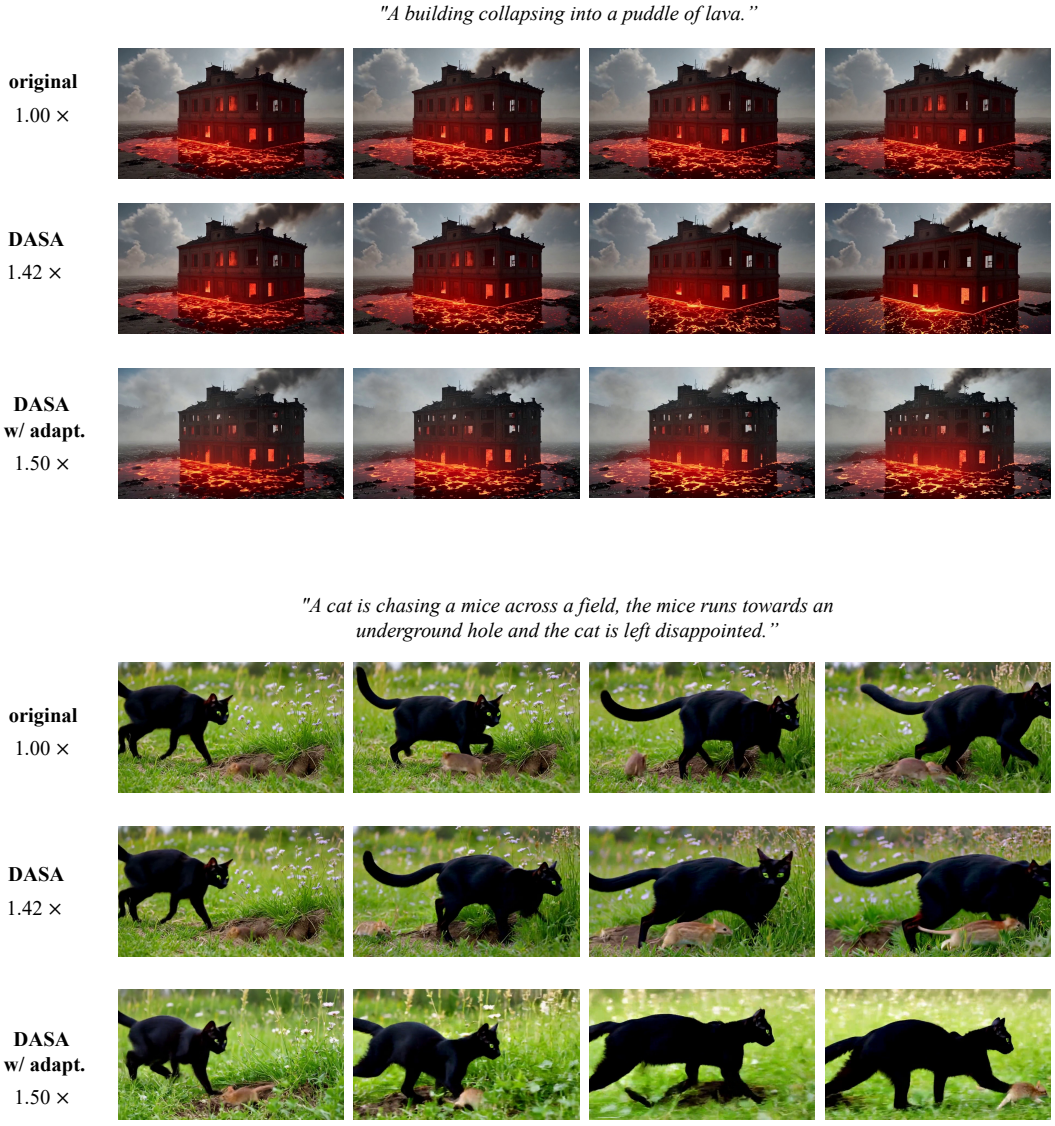

Figure 9: Visualization of 480P videos generated by Self Forcing.



Figure 10: Visualization of 720P videos generated by Self Forcing.

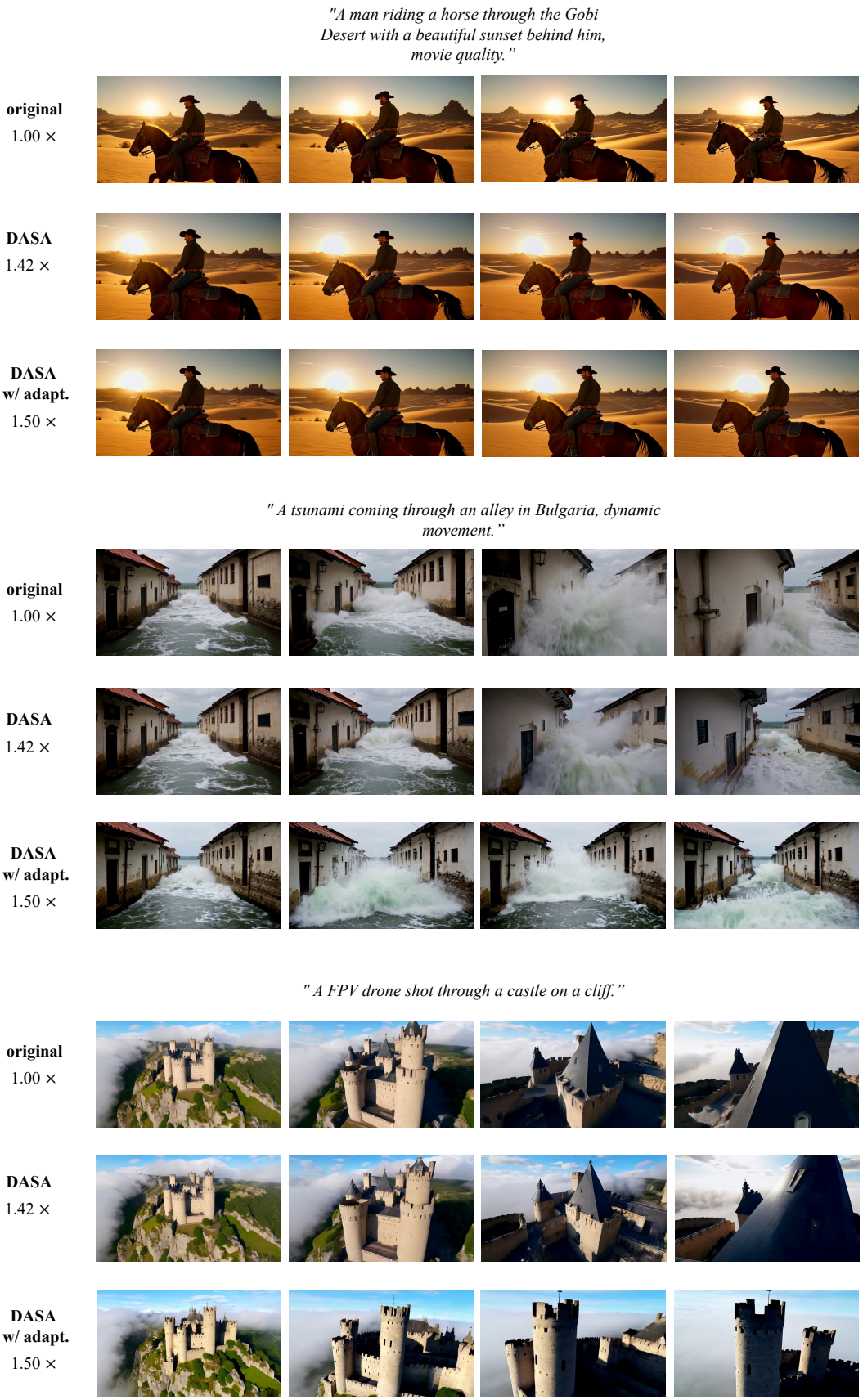

Figure 11: Visualization of 1080P videos generated by Self Forcing.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The claims in the abstract and introduction are consistent with the proposed method and supported by experiments showing improved efficiency with minimal quality drop (Sections 3–4).

   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: Limitations are discussed, including focusing only on token-level redundancy and not exploring other optimization dimensions (Limitations section).

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [N/A]

Justification: The work is empirical and does not rely on theoretical proofs.

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: Comprehensive experiments are conducted across multiple settings with standard benchmarks and metrics (Section 4).

   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: Code is publicly available now; sufficient details are provided for reproduction.

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: Datasets, evaluation protocol, and hardware are clearly described (Section 4.1).

Guidelines:

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

7. **Experiment statistical significance**

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

Answer: [No]

Justification: The paper reports single-run quantitative results without error bars or statistical significance analysis. Due to the high computational cost of large-scale video generation experiments, multiple runs were not conducted.

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: Compute resources (H100/H20 GPUs) and efficiency metrics are reported (Section 4.1).

   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: Potential misuse of video generation is acknowledged; no new risks are introduced.

   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: [Yes]

    Justification: The work improves the efficiency of autoregressive video generation, which may benefit applications such as simulation, content creation, and interactive systems. However, improved video generation capabilities may also lower the barrier for generating realistic synthetic media, which could be misused for disinformation or deepfakes. Potential risks could be mitigated through responsible deployment practices such as controlled release and usage monitoring.

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: [No]

    Justification: The paper does not release models or datasets, and therefore no specific safeguards are implemented. Potential risks are acknowledged, and safeguards can be considered in future deployment.

    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: The paper uses publicly available datasets and baseline models with proper citations, and explicitly states their licenses or terms of use.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.

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

13. **New assets**

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

    Answer: [N/A]

    Justification: The paper does not introduce or release any new datasets, models, or code as assets.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or experiments with human subjects.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects, and thus no IRB approval is required.

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 part of the core methodology or experiments. Any usage was limited to writing and editing assistance, which does not require disclosure.

    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.