

Parameterized Stripe Attention for Efficient Video Generation

Xingyu Jia, Baole Ai, Ang Wang, Kang Zhao, Yong Li
Alibaba Group

Abstract

Diffusion Transformers (DiTs) enable high-quality video generation but suffer from substantial inference latency, primarily attributable to the computationally expensive full spatio-temporal attention. While sparse attention methods offer potential solutions, existing approaches face an inherent flexibility–efficiency dilemma: predefined masks lack the flexibility to capture diverse attention patterns, while runtime-determined masks introduce overheads and sacrifice hardware efficiency. We identify the lack of a unified structural characterization of DiT attention as a key limitation of existing methods, and establish that video DiT attention exhibits **periodic diagonal stripe structures** along both temporal and spatial dimensions. To formally encode these structured patterns within a single efficient kernel, we present **PSA**, a parameterized stripe attention that formalizes the observed stripe regularity, unifying diverse attention patterns for efficient mask generation. This unified representation enables a single hardware-efficient CUDA kernel to process all sparse patterns, achieving FlashAttention-3-level Model FLOPs Utilization. To determine optimal sparsity configurations, we propose a training-free offline search algorithm that automatically maximizes sparsity under a specified error tolerance for each attention head. Experiments on HunyuanVideo and Wan 2.1 demonstrate that PSA achieves $1.57\times$ and $1.37\times$ end-to-end speedups over FlashAttention-3 baselines, with acceptable visual quality degradation.

1 Introduction

Diffusion Transformers (DiTs) [18] have rapidly emerged as a foundational architecture for image and video generation. DiTs employ Transformers [24] as the backbone and leverage algorithms such as Diffusion [20, 14, 15, 18] or Flow Matching [11, 5] to iteratively generate images and videos from noise. Prominent open-source models such as Wan 2.1 [25] and HunyuanVideo [8] leverage the DiT architecture for tasks like text-to-video and image-to-video synthesis, producing high-quality, highly dynamic video content. However, DiT inference requires multi-step denoising with enormous computational costs and prolonged latency. For instance, generating a 5-second, 720p video with Wan 2.1 on an NVIDIA H100 GPU takes nearly 30 minutes [25], severely limiting practical applications. Since these models utilize full spatio-temporal 3D attention [32], the attention computation constitutes the primary bottleneck, accounting for over 60% [8] of the total inference time. Consequently, designing more efficient attention mechanisms is a critical optimization target. Sparse attention—selectively computing only the most informative token interactions—offers a promising avenue, provided the attention distribution exhibits exploitable structure.

Existing sparse attention methods for DiTs lack a principled structural characterization of attention distributions, leading to an inherent trade-off between flexibility and efficiency. The first category relies on **predefined** masks designed from empirical observations rather than principled analysis. SVG [28] identifies only two fixed patterns—temporal and spatial heads—without analyzing why only these two arise or where the observed periodicity originates; consequently, it cannot capture the diverse stride, offset, and hybrid patterns revealed by our analysis. STA [35] achieves efficient local

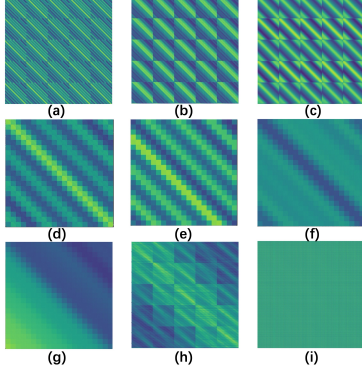


Figure 1: Heat maps in Wan 2.1 and HunyuanVideo.

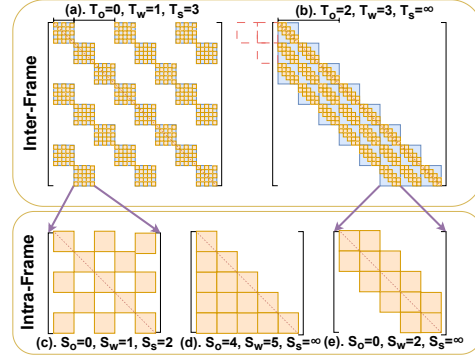


Figure 2: Visualization of the PSA parameterization that captures various attention sparse patterns.

computation via tile-wise sliding windows, but assumes only local attention and cannot represent long-range, strided, or hybrid structures as shown in fig. 1. Because these methods are not grounded in a comprehensive characterization of attention patterns, their fixed masks inevitably fail to cover the full diversity of sparse structures. The second category pursues more flexible pattern representations by determining the sparse mask dynamically **at runtime**, effectively treating attention distributions as arbitrary. For example, SVG2 [31] clusters Q and K tokens separately to enable flexible sparse patterns, overlooking the structural regularity revealed by our analysis. This abandonment of structure comes at a cost: the irregular cluster-based kernel’s reliance on FlashInfer [33] precludes Tensor Memory Accelerator (TMA) utilization and introduces non-trivial padding overhead ($\sim 20\%$ at 70% sparsity) despite token permutation; and the clustering procedure itself incurs additional computational cost ($\sim 10\%$ at 70% sparsity). Consequently, the flexibility–efficiency trade-off in existing methods stems from the lack of a principled, unified framework to characterize sparse attention: methods with fixed masks fail to cover the full diversity of DiT-specific sparse structures, while methods pursuing flexibility sacrifice efficiency by treating attention distributions as arbitrary rather than structured. This motivates the need for a more mechanistic understanding of DiT attention patterns that can support both broad pattern coverage and high-performance implementation.

Concretely, sparse attention for video DiTs involves three main challenges: (1) A principled characterization of the DiT attention structure space is needed, providing a unified and sufficiently expressive description of the diverse sparse patterns that emerge across heads, layers, and denoising timesteps. (2) A hardware-aware implementation must be developed that supports various mask configurations while maximizing GPU utilization. (3) An efficient search method is required to determine per-head sparse configurations without incurring additional inference-time overhead.

To address these challenges, we propose PSA, a parameterized stripe sparse attention framework. First, we reveal that video DiT attention exhibits **periodic diagonal stripe structures** along both temporal and spatial dimensions, through theoretical derivation (via RoPE-3D logit decomposition and video spatio-temporal redundancy) and large-scale empirical validation across all attention heads in HunyuanVideo and Wan 2.1 (fig. 1), identifying four primary patterns: intra-frame, inter-frame, hybrid, and uniform (see section 3.1 for details). Second, based on this structural understanding, we design a parameterized mask $\mathcal{M}(T_w, T_o, T_s, S_w, S_o, S_s)$ that provides a unified representation of periodic stripe sparse patterns across attention heads, with quantitative evaluation demonstrating high-fidelity approximation of full attention outputs (average cosine similarity of 0.9831). Leveraging this unified representation, we support different patterns with a single kernel, eliminating multi-kernel launch overhead. Our implementation adopts block-wise partitioning as the fundamental granularity to fully exploit modern GPU hardware resources, achieving 61% Model FLOPs Utilization (MFU) on HunyuanVideo—close to FlashAttention-3 levels—thereby resolving the flexibility–efficiency tension. Finally, observing strong similarity in attention masks across different prompts, we design an offline parallel search algorithm that requires no training and introduces no runtime search overhead, assigning the mask with the highest sparsity to each head while maintaining error bounds. Our parametric pattern representation and offline search achieve both objectives: head-adapted patterns are determined prior to inference, thereby **eliminating runtime overhead while preserving adaptability**. We validate our method on HunyuanVideo and Wan 2.1, achieving $1.57\times$ and $1.37\times$

end-to-end speedups over FlashAttention-3, respectively, with acceptable quality degradation (PSNR > 23). In summary, our main contributions are as follows.

1. We reveal that video DiT attention exhibits periodic diagonal stripe structures along both temporal and spatial dimensions, and provide both theoretical derivation and large-scale empirical validation to establish this structural property.
2. We propose a parameterized mask $\mathcal{M}(T_w, T_o, T_s, S_w, S_o, S_s)$ that unifies diverse sparse patterns, enabling a single hardware-efficient CUDA kernel to process all supported patterns with FlashAttention-3-level MFU, resolving the flexibility–efficiency tension. We further introduce 3D-padding to support arbitrary video resolutions with minimal overhead ($\approx 5\%$ for 720p).
3. We develop a training-free offline search algorithm that maximizes sparsity under a constrained error threshold for each attention head, with support for distributed parallel search to accelerate the process.
4. The PSA achieves substantial acceleration without compromising effectiveness, outperforming state-of-the-art methods such as SVG2 and STA in both accuracy and efficiency.

2 Related Work

Attention mechanisms are pivotal to the success of Transformers [24], yet their quadratic complexity poses a fundamental efficiency bottleneck. Existing efficient attention methods for video generation primarily construct sparse masks by exploiting the distributional characteristics of attention heatmaps, falling into two paradigms. The first paradigm relies on **predefined sparse mask formats**. SVG [28] classifies attention heads into spatial and temporal types and employs online profiling to generate sparse masks. VORTA [23] trains a lightweight router to dynamically select among full attention, sliding window, and coreset attention patterns. Sparse-vDiT [3] selects from a set of predefined structural patterns, including diagonal, multi-diagonal, and vertical-stripe sparsity. Additionally, other window-based attention methods exploit the locality of attention. NATTEN [6] implements sliding window attention for vision tasks, localizing self-attention to nearest neighborhoods. STA [35] employs tile-wise window attention for improved memory access and parallelism. The second paradigm **determines sparse masks dynamically** during inference based on attention scores. VMoBA [27] extends Moba [16] to adapt spatio-temporal attention within DiTs. MOD-DiT [12] avoids the sampling overhead of runtime mask determination by linearly extrapolating pattern intensities from early denoising steps. RoPeSLR [13] combines block-sparse attention with a trained low-rank compensator to restore dropped background contributions. SVG2 [31] improves efficiency through k -means clustering of Q and K vectors, followed by token permutation and top- k cluster selection to identify the most relevant attention regions.

However, existing methods suffer from notable limitations: predefined pattern approaches are constrained by a limited pattern set, while dynamic mask generation leads to non-trivial overhead and poor efficiency, yielding limited speedup over optimized dense implementations such as FlashAttention [4, 19]. In contrast, PSA uniformly parameterizes diverse sparse patterns with hardware-aware kernels, achieving superior flexibility and efficiency.

3 Method

3.1 Structural Analysis of Attention Patterns

We begin by formally characterizing the structural properties of attention maps in video DiTs, aiming to move beyond empirical observation toward a mechanistic understanding of the patterns governing these distributions. Our analysis reveals that attention in video DiTs is neither arbitrary nor purely local; rather, it consistently manifests as periodic diagonal stripe structures, parameterized by their widths, offsets, and strides. We verify this structural analysis through a comprehensive examination of attention maps in HunyuanVideo and Wan 2.1, identifying four primary attention patterns as shown in fig. 1:

1. **Intra-frame** [Fig. (a)–(c)]: diagonal stripes with varying widths, offsets, and strides within each frame, repeating identically across frames.

2. **Inter-frame** [Fig. (d)–(g)]: relatively uniform intra-frame weights, with diagonal band structures of varying widths, offsets, and strides emerging between frames.
3. **Hybrid** [Fig. (h)]: a combination of intra-frame and inter-frame patterns.
4. **Uniform** [Fig. (i)]: attention weights uniformly distributed across all positions.

We show that this structure arises from two complementary properties: **(1) video spatio-temporal redundancy**, which concentrates high attention scores within local neighborhoods, and **(2) RoPE-3D cosine periodicity**, which causes attention scores to repeat at axis-dependent intervals.

Property 1: Video spatio-temporal redundancy. Videos exhibit rich spatio-temporal redundancy at multiple scales [17, 26]. Since attention is the only component in a DiT block that computes pairwise token interactions, this redundancy is directly reflected in the attention score matrix $\mathbf{A}$. Concretely, fixing any two axes and varying the third, high attention scores concentrate within locality radii ε_t , ε_h , ε_w along the temporal, height, and width axes, respectively:

$$|\Delta t| < \varepsilon_t \ (h, w \text{ fixed}), \quad |\Delta h| < \varepsilon_h \ (t, w \text{ fixed}), \quad |\Delta w| < \varepsilon_w \ (t, h \text{ fixed}), \quad (1)$$

where $\Delta t = t - t'$, $\Delta h = h - h'$, $\Delta w = w - w'$ denote relative offsets along each axis. In the 2D heatmap, the locality conditions in eq. (1) concentrate high attention along diagonal bands—forming the stripe **width** (ε_t inter-frame, ε_w intra-frame)—while column-wise locality (ε_h) produces multiple parallel diagonals with layout-determined stride (see Appendix D).

Property 2: RoPE-3D cosine periodicity. We show that RoPE-3D [22, 8, 25] introduces structured periodicity by partitioning embedding channels into three disjoint subsets M_t , M_h , M_w encoding temporal, height, and width positions. For query q_i at position (t, h, w) and key k_j at (t', h', w') , the attention logit decomposes as

$$\begin{aligned} a_{(t,h,w),(t',h',w')} &= \sum_{m \in M_t} \|q_i^{(m)}\| \|k_j^{(m)}\| \cos(\phi^{(m)} + (t - t') \theta_m) \\ &+ \sum_{m \in M_h} \|q_i^{(m)}\| \|k_j^{(m)}\| \cos(\phi^{(m)} + (h - h') \theta_m) \\ &+ \sum_{m \in M_w} \|q_i^{(m)}\| \|k_j^{(m)}\| \cos(\phi^{(m)} + (w - w') \theta_m), \end{aligned} \quad (2)$$

where $q_i^{(m)}, k_j^{(m)} \in \mathbb{R}^2$ are the content query and key slices for channel m ; $\phi^{(m)}$ is the angle between $q_i^{(m)}$ and $k_j^{(m)}$; and $\theta_m = c^{-2m/d}$ is the angular frequency of channel m , with d the total embedding dimension and c the RoPE base. The periodicity becomes most pronounced when a single channel dominates its group [30]. Suppose a single channel $m_\lambda^* \in M_\lambda$ dominates its group, i.e.,

$$\|q_i^{(m_\lambda^*)}\| \|k_j^{(m_\lambda^*)}\| \gg \|q_i^{(m)}\| \|k_j^{(m)}\| \quad \forall m \in M_\lambda, m \neq m_\lambda^*. \quad (3)$$

We empirically verify this dominance assumption on Wan 2.1, where 87.55% of attention heads exhibit a dominant channel (see Appendix E for details). Then the contribution of group $\lambda \in \{t, h, w\}$ to eq. (2) reduces to a single cosine term. Taking $\lambda = t$ as a representative example,

$$a_{(t,h,w),(t',h',w')} \approx \|q_i^{(m_t^*)}\| \|k_j^{(m_t^*)}\| \cos(\phi^{(m_t^*)} + (t - t') \theta_{m_t^*}), \quad (4)$$

and analogously for $\lambda \in \{h, w\}$ with $(t - t')$ replaced by $(h - h')$ or $(w - w')$, respectively. The logit is thus a cosine of the relative offset along axis λ , yielding an axis-wise attention period of

$$\mathcal{T}_\lambda = \frac{2\pi}{\theta_{m_\lambda^*}} = 2\pi c^{2m_\lambda^*/d}, \quad \lambda \in \{t, h, w\}. \quad (5)$$

In the heatmap visualization, this cosine periodicity means that high attention weights recur at fixed intervals $\mathcal{T}_\lambda$ —producing the **stride** of the observed stripes. Moreover, the content-dependent phase $\phi^{(m_\lambda^*)}$ can shift the cosine peak away from the main diagonal, producing the **offset**.

Joint effect. When $\mathbf{A}$ is visualized as a 2D heatmap under the flattened 1D token ordering, the two properties identified above—video redundancy and axis-wise cosine periodicity—jointly produce the observed periodic diagonal stripes: **(1) Inter-frame stripes** are driven by temporal locality in eq. (1): the stripe *width* is governed by ε_t , and the stripe *period* is governed by $\mathcal{T}_t$. **(2) Intra-frame stripes** are driven by spatial locality within each frame block: the stripe *width* is governed by ε_w (row-wise locality), and the stripe *period* is governed by $\mathcal{T}_h$ and $\mathcal{T}_w$ (eq. (5)), with column-wise locality (ε_h) inducing repeated off-diagonals at intervals of W under row-major flattening.

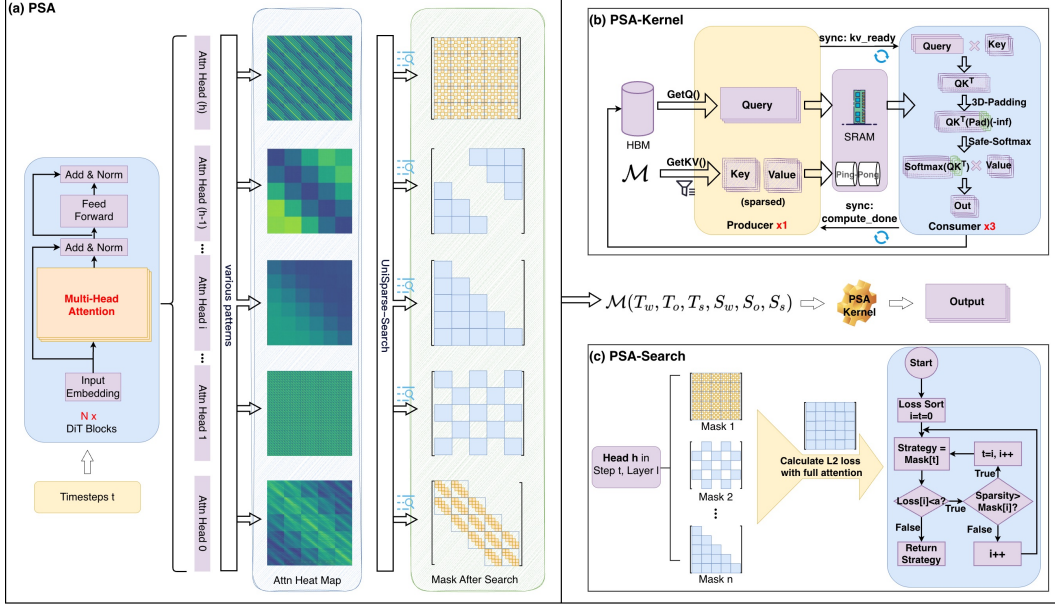


Figure 3: An overview of the PSA framework. **(a) Unified Representation:** The parametric model $\mathcal{M}$ captures diverse per-head sparse attention patterns, covering both intra-frame and inter-frame structures with varying widths, offsets, and strides. **(b) Efficient Kernel:** A dedicated CUDA kernel enables high-performance, hardware-aware computation across all supported sparse patterns. **(c) Automatic Search:** PSA-Search automatically identifies the optimal sparse configuration for each head by maximizing sparsity subject to a predefined output degradation threshold.

3.2 Efficient and Flexible Kernel

Unified Representation. To efficiently encode the sparse diagonal-stripe patterns identified above and enable a single, hardware-efficient kernel, we propose a parametric formulation $\mathcal{M}(T_w, T_o, T_s, S_w, S_o, S_s)$, where six integer parameters govern inter-frame and intra-frame diagonal bands via their widths, offsets, and strides, respectively, as illustrated in fig. 2. SVG and STA are special cases: SVG corresponds to $\mathcal{M}(T_w, 0, \infty, S_w, 0, \infty)$, and STA’s sliding window to $\mathcal{M}(T_w, 0, \infty, S_w, 0, S_s)$, where $T_s = \infty$ or $S_s = \infty$ reduces the corresponding axis to a single diagonal. These restricted configurations fail to capture multi-diagonal, hybrid, and uniform patterns, which account for over 60% of attention heads in HunyuanVideo and Wan 2.1. Our formulation expresses all four observed patterns (see fig. 3 (a)), achieving an average cosine similarity of 0.9831 with full attention across all heads. To generate attention masks, we devise the generation algorithm (algorithm 1). Given an input sequence of length $L = T \times H \times W$, the mask generation algorithm permits an attention connection between query q_i and key k_j only when both the inter-frame and intra-frame conditions of $\mathcal{M}$ are satisfied.

Hardware-Aware Design. Our kernel design targets maximum computational efficiency on modern GPUs by ensuring full utilization of every scheduled thread block, eliminating idle compute units [35]. To this end, we introduce a fundamental computational abstraction called PSABlock, which represents the minimum data granularity necessary to fully saturate the resources of a single streaming multi-processor (SM). Notably, PSABlock also aligns with the block-wise characteristics observed in the attention heatmaps (fig. 1), enabling the mask to operate naturally at this granularity. Since PSABlock is the minimum computable unit, PSA applies masking at PSABlock granularity rather than individual tokens. For video tokens organized in 3D structure, a PSABlock comprises constituent blocks along temporal, height, and width dimensions: $\text{PSABlock} = \text{PSABlock}_t \times \text{PSABlock}_h \times \text{PSABlock}_w$.

High-Performance Kernel. We develop a high-performance CUDA kernel, PSA-Kernel, specifically optimized for the PSA framework on the NVIDIA Hopper GPU architecture. Built upon the ThunderKittens [21] library, PSA-Kernel draws inspiration from its highly efficient FlashAttention-3 pipeline design, as illustrated in fig. 3 (b).

Algorithm 1 PSA Mask Definition

Input: Q index i , K index j , Parameters $\mathcal{M}(T_w, T_o, T_s, S_w, S_o, S_s)$
Output: Boolean indicating if q_i attends to k_j

```
1: function ISATTENDING( $i, j, \mathcal{M}$ )
2:    $(t_q, s_q) \leftarrow (\lfloor i/(H \cdot W) \rfloor, i \% (H \cdot W))$ 
3:    $(t_k, s_k) \leftarrow (\lfloor j/(H \cdot W) \rfloor, j \% (H \cdot W))$ 
4:    $\Delta_t \leftarrow t_k - t_q$  ▷ inter-frame distance
5:    $\text{inter\_frame\_cond} \leftarrow ((|\Delta_t - T_o|) \bmod T_s) \leq T_w - 1$ 
6:   if not  $\text{inter\_frame\_cond}$  then
7:     return false
8:   end if
9:    $\Delta_s \leftarrow s_k - s_q$  ▷ intra-frame distance
10:   $\text{intra\_frame\_cond} \leftarrow ((|\Delta_s - S_o|) \bmod S_s) \leq S_w - 1$ 
11:  return  $\text{inter\_frame\_cond} \wedge \text{intra\_frame\_cond}$ 
12: end function
```

The key challenge lies in maintaining high computational efficiency across diverse sparse patterns. We address this through two synergistic mechanisms. First, we reorganize the Q, K, V data layout from the standard (T, H, W) format to a PSABlock-centric arrangement: $(T', H', W', PSABlock_t, PSABlock_h, PSABlock_w)$, where (T', H', W') denotes the grid of PSABlocks. This transformation ensures contiguous memory placement within each PSABlock, enabling coalesced memory accesses—a prerequisite for the efficient kernel design that follows. Second, building on this layout, our kernel achieves FA3-level efficiency while supporting flexible masks as shown in fig. 3(b). A dedicated *producer* warpgroup guided by the mask logic (algorithm 1) identifies required K, V PSABlocks and asynchronously prefetches them into on-chip SRAM via TMA-driven ping-pong buffering. Concurrently, *consumer* warpgroups perform dense attention computations at PSABlock granularity using resident Q and prefetched K, V data. This decoupling of data movement and computation enables hardware-optimized dense operations, maximizing GPU utilization even under sparse attention patterns.

3D-Padding for Resolution Flexibility. The block-aligned computation described above requires that the token dimensions (T, H, W) be divisible by the corresponding PSABlock tile sizes $(PSABlock_t, PSABlock_h, PSABlock_w)$. To support arbitrary video resolutions, we employ **3D-padding**: each dimension is minimally padded to the smallest multiple of its corresponding PSABlock size, i.e., $T' = \lceil T/PSABlock_t \rceil \cdot PSABlock_t$, ensuring alignment without restricting input geometry. This design enables support for arbitrary video resolutions with minimal computational overhead. For instance, at 720p resolution, padding the height dimension (H) from 45 to 48 tokens during the last 80% of generation steps incurs only 5% overhead (Overhead = $p_{\text{steps}} \times r_{\text{padding}}$, where p_{steps} denotes the proportion of steps requiring padding and r_{padding} represents the relative padding ratio).

3.3 Automatic and Efficient Sparsity Discovery

A central challenge in applying the PSA representation is assigning an optimal mask configuration $\mathcal{M}$ to each attention head. We address this with PSA-Search, an automatic search algorithm grounded in a key empirical observation: attention patterns in DiT models vary significantly across timesteps, layers, and heads, yet remain remarkably stable with respect to input prompts [3] (see fig. 4 and Appendix G). This prompt-invariance enables an offline search strategy—optimal sparse configurations can be identified once using a representative dataset and subsequently applied to all future inferences, introducing no runtime overhead.

The PSA-Search algorithm operates on a per-head basis, as illustrated in fig. 3 (c). For each attention head, the algorithm iterates through a predefined search space of candidate sparse configurations. Given a user-defined L_2 loss threshold α , the search first identifies all candidate masks satisfying

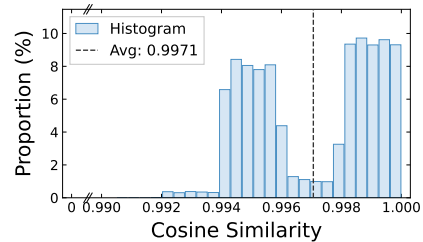


Figure 4: Evaluation of the similarity of attention heatmaps across all 11 evaluation dimensions of VBench, where prompts 1–99 are compared against prompt 0 at the same head positions within a search space of 100 (timesteps) $\times$ 40 (layers) $\times$ 40 (heads).

Table 1: Comparative evaluation of PSA against SVG, SVG2 and STA baselines across different models. Spar denotes the average sparsity across all attention heads. “-” indicates metrics that are self-referential for the baseline.

Models	Methods	Spar (%)	E2E Speed (s)↓	OC↑	TS↑	MSE↓	PSNR↑	SSIM↑	LPIPS↓
HunyuanVideo (720×1280)	FA2	0	1793.06	26.95	24.52	-	-	-	-
	FA3	0	1172.62 (1x)	26.95	24.52	-	-	-	-
	SVG	56.92	906.12 (1.29x)	22.32	21.11	0.0039	24.2716	0.8686	0.3400
	SVG2	61.92	866.10 (1.35x)	25.96	23.28	0.0016	29.6303	0.8910	0.1625
	Ours	59.74	777.28 (1.51x)	26.00	23.63	0.0012	30.3463	0.9062	0.1523
HunyuanVideo (768×1280)	FA2	0	1684.60	26.97	24.61	-	-	-	-
	FA3	0	1106.36 (1x)	26.97	24.61	-	-	-	-
	STA	58.37	746.73 (1.48x)	26.19	23.39	0.0596	12.5239	0.6036	0.3117
	SVG	56.98	797.57 (1.39x)	23.21	22.43	0.0053	22.9984	0.8654	0.1907
	SVG2	63.23	773.47 (1.43x)	25.31	22.74	0.0027	25.7979	0.7855	0.2184
	Ours	60.33	703.91 (1.57x)	26.33	23.64	0.0019	27.1983	0.8287	0.1568
Wan 2.1 14B (720×1280)	FA2	0	1873.32	25.91	23.19	-	-	-	-
	FA3	0	1214.69 (1x)	25.91	23.19	-	-	-	-
	SVG	58.51	944.81 (1.29x)	25.29	20.32	0.0157	18.5862	0.7056	0.2881
	SVG2	63.40	926.78 (1.31x)	24.69	23.33	0.0051	23.4689	0.8391	0.1703
	Ours	57.57	886.64 (1.37x)	25.29	23.59	0.0048	23.4813	0.8372	0.1601

this quality constraint (i.e., $L_2 \leq \alpha$), where the L_2 loss is computed between the outputs of the sparse attention and the full attention. Among these qualified candidates, PSA-Search selects the mask achieving maximum sparsity. Furthermore, PSA-Search supports a dual-interface: users can either specify an error tolerance α to maximize sparsity, or specify a target sparsity level and obtain the corresponding mask with minimal error. To enhance search efficiency, we leverage distributed parallelization across multiple GPUs via Ulysses [7]. Searching over 100 candidate masks completes in 2.5 hours on 8 H100 GPUs, assigning each head its maximum-sparsity mask under the error constraint without the prohibitive costs of training-based methods.

4 Experiments and Results

4.1 Setup

Models. We evaluate two state-of-the-art text-to-video models: **Wan 2.1** (14B parameters) with 81-frame sequences at 720×1280, and **HunyuanVideo** in two configurations—129 frames, 720×1280 and 117 frames, 768×1280.

Metrics. For quantitative evaluation of frame-level fidelity and perceptual quality, we report Mean Squared Error (MSE), Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS) [36]. In addition to these per-frame metrics, we conduct a more holistic assessment using the VBench benchmark [34]. Specifically, we report scores for overall consistency (OC) and temporal style adherence (TS).

Baselines. We compare against three sparse attention methods: **SVG** and **SVG2**, evaluated across all models to assess generalizability; and **STA**, evaluated only on HunyuanVideo at 768×1280 resolution due to its tile-based architectural constraints [35]. These methods represent the two paradigms discussed in section 1: **SVG** and **STA** employ predefined masks with limited flexibility, while **SVG2** determines masks dynamically at runtime, sacrificing hardware efficiency for flexibility.

Parameters. All methods target approximately 60% sparsity. For PSA, we construct a discrete candidate space for $\mathcal{M}$ based on attention heatmap analysis, uniformly sampling about 100 configurations per head. The threshold α is set to 0.03, 0.018, and 0.007 for HunyuanVideo (720×1280), HunyuanVideo (768×1280), and Wan 2.1, respectively. Search uses 1–3 prompts (inter-prompt similarity is high) and evaluation uses all VBench prompts. The initial 20% of diffusion timesteps are exempted from sparsification, as early steps are critical for coarse layout and quality [28, 31, 29, 1, 2, 37].

4.2 PSA Performance and Video Quality

We evaluate PSA-Kernel at both kernel and end-to-end (E2E) levels on a single NVIDIA H100 GPU, benchmarking against FlashAttention-2 (FA2) [4] and FlashAttention-3 (FA3) [19].

Table 2: Kernel performance of PSA across different models with the same mask pattern. Spar denotes the kernel sparsity of each individual attention head.

Models	Methods	Spar (%)	Kernel Speed (ms)↓	TFLOPs	MFU (%)↑
HunyuanVideo (720×1280)	FA3	0	251.30 (1.00x)	173.43	69.78
	SVG	65.41	173.40 (1.45x)	59.99	34.98
	SVG2	65.67	156.76 (1.60x)	59.54	38.40
	Ours	66.00	121.58 (2.07x)	58.96	49.04
HunyuanVideo (768×1280)	FA3	0	236.36 (1.00x)	163.07	69.76
	STA	58.33	111.73 (2.12x)	67.95	61.50
	SVG	58.43	178.59 (1.32x)	67.79	38.38
	SVG2	57.68	182.79 (1.29x)	69.01	38.18
	Ours	58.33	112.25 (2.11x)	67.95	61.21
Wan 2.1 14B (720×1280)	FA3	0	185.55 (1.00x)	117.05	63.78
	SVG	75.41	77.61 (2.39x)	28.78	37.50
	SVG2	74.84	69.86 (2.66x)	29.45	42.62
	Ours	75.00	55.12 (3.37x)	29.26	53.68

E2E inference performance. As presented in table 1, PSA consistently outperforms existing sparse attention baselines across different model architectures. Compared to SVG, SVG2 and STA, it achieves relative speedups of $1.13\times$, $1.10\times$, and $1.06\times$ on HunyuanVideo (768×1280), respectively.

Kernel performance. To isolate the computational efficiency of our core attention mechanism, we conduct detailed micro-benchmarks of PSA-Kernel against the kernels from SVG, SVG2 and STA under the same mask pattern, as shown in table 2. Implementation differences across methods may cause slight sparsity variations even under the same mask pattern. Notably, compared to the SVG and SVG2 kernels, our kernel achieves significant speedups of approximately $1.59\times$ and $1.63\times$ on HunyuanVideo (768×1280), respectively. Moreover, it achieves kernel performance competitive with STA while delivering superior E2E speed—the E2E advantage stems from our FA3-based implementation for unsparisified timesteps (the initial 20% of diffusion steps), whereas STA relies on a slower custom full-attention kernel during these steps. This kernel-level efficiency serves as the foundational driver of the end-to-end latency reductions, confirming that our method’s performance gains stem from a highly optimized computational core.

Video Quality. Beyond computational performance, a critical question is whether PSA preserves video generation quality. As shown in table 1, PSA achieves the best VBench scores (OC and TS) among all sparse methods across all configurations, while also delivering the best frame-level fidelity metrics (MSE, PSNR, SSIM, LPIPS). Notably, on HunyuanVideo (720×1280), PSA achieves a PSNR of 30.35 at 59.74% sparsity, confirming that the parameterized mask $\mathcal{M}$ faithfully captures each head’s attention structure.

4.3 Sensitivity Tests

Quality sensitivity tests. As shown in table 3a, we evaluate the generation quality metrics under varying levels of sparsity. The results clearly indicate that as sparsity increases from 30% to 60%, all evaluation metrics exhibit a smooth and controlled degradation. Notably, even at a high sparsity of 60%, the model maintains an acceptable generation quality (e.g., PSNR remains above 30). This demonstrates that the negative impact on generation quality is within a manageable range, even while unlocking significant computational savings.

Kernel sensitivity tests. Table 3b presents kernel performance across varying sparsity levels. Our kernel consistently outperforms SVG across all sparsity settings. While STA exhibits a marginal latency advantage at the kernel level, our kernel supports arbitrary video resolutions via 3D-padding with negligible overhead, whereas STA is constrained to a single fixed resolution. This generality, combined with competitive kernel throughput, gives our method practical deployment value.

4.4 Bracketing Comparison

For all baselines, sparsity is an *outcome* rather than an *input*: STA’s is fixed by its predefined window masks, SVG’s is quantized to the nearest feasible window combination, and SVG2’s emerges from data-dependent runtime clustering, so an exactly matched sparsity budget cannot be enforced on the baseline side. We therefore evaluate two operating points on HunyuanVideo (768×1280) whose

Table 3: Sensitivity tests on quality and kernel performance.

(a) Quality (HunyuanVideo 720 $\times$ 1280)					(b) Performance (ms, HunyuanVideo 768 $\times$ 1280)				
Sparsity	MSE $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	Sparsity	FA3	STA	SVG	Ours
30%	0.0005	34.9341	0.9519	0.0834	60%	236.366	111.508	165.028	113.494
40%	0.0007	33.7610	0.9431	0.0875	70%	236.366	79.702	125.161	80.438
50%	0.0010	31.6886	0.9171	0.1250	80%	236.366	53.624	85.027	54.093
60%	0.0012	30.3463	0.9062	0.1523	90%	236.366	25.882	45.643	26.292

Table 4: Bracketing comparison on HunyuanVideo (768 $\times$ 1280). Ours-min and Ours-max are two operating points whose sparsity strictly brackets that of all baselines.

Method	Spar (%)	Speedup $\uparrow$	PSNR $\uparrow$
SVG	56.98	1.39 $\times$	23.00
STA	58.37	1.48 $\times$	12.52
SVG2	63.23	1.43 $\times$	25.80
Ours-min	56.47	1.51$\times$	28.73
Ours-max	63.84	1.65$\times$	26.76

Table 5: Fraction of dominant-channel heads across nine model settings.

Model	Type	Dominant-channel heads (%)
Wan 2.1 14B (50-step)	video DiT, 3D full attention	87.55
Wan 2.1 14B (40-step)	video DiT, 3D full attention	89.30
Wan 2.1 1.3B	video DiT, 3D full attention, 832 $\times$ 480	90.81
HunyuanVideo (720 $\times$ 1280)	video MM-DiT	96.61
HunyuanVideo (768 $\times$ 1280)	video MM-DiT	96.33
FLUX.1-schnell [9]	image MM-DiT, 2D RoPE	89.16
CogVideoX-5b [32]	video DiT, concatenated attention	90.39
Open-Sora v1.2	video DiT, factorized attention	91.95
Hunyuan-DiT v1.2 [10]	image DiT, 2D patchify	78.93

sparsity *strictly brackets* all baselines: **Ours-min** below every baseline and **Ours-max** above every baseline. As shown in table 4, PSA dominates every baseline in both speed and quality at both endpoints: Ours-min is the fastest despite computing the most (1.51 $\times$ vs. at most 1.48 $\times$), and Ours-max achieves the best PSNR despite computing the least (26.76 vs. 25.80), confirming that the conclusions of table 1 hold regardless of any residual sparsity mismatch.

4.5 Generality of the Stripe Structure

To examine how broadly the structural premise underlying PSA holds, we measure the fraction of dominant-channel heads (eq. (3)) on nine model settings, following the protocol of Appendix E. As shown in table 5, the dominant-channel fraction exceeds 78.9% on all nine settings (median $\approx$ 90%), spanning resolutions (480p/720p/768p), model sizes (14B/1.3B), architectures (MM-DiT, text-visual concatenated, and factorized spatio-temporal attention), modalities (video/image), tokenizations (2D/3D patchify), and denoising schedules (50 vs. 40 steps). The attention heatmaps of all these models consistently exhibit the same periodic diagonal stripes as fig. 1. Notably, the RoPE-free spatial attention of Open-Sora v1.2 [38] still exhibits locality-induced stripes with stride W , indicating that spatio-temporal locality (eq. (1)) holds independently of RoPE and tokenization.

4.6 Fidelity of Searched Masks

Our masks achieve an average cosine similarity of 0.9831 with full attention. At 60% sparsity, over 93.6% of heads maintain L_2 loss below 10^{-2} , confirming that PSA-Search produces high-fidelity sparse approximations.

Table 6: Ablation study on mask at a fixed 60% sparsity (HunyuanVideo, 720×1280).

Config	MSE↓	PSNR↑	SSIM↑	LPIPS↓
SVG-like	0.0027	26.0230	0.8364	0.2073
STA-like	0.0026	26.1466	0.8133	0.2343
Inter-frame	0.0019	27.8065	0.8505	0.1757
Intra-frame	0.0042	24.0145	0.7838	0.2730
Ours	0.0012	30.3463	0.9062	0.1523

4.7 Ablation Studies

To further validate that $\mathcal{M}(T_w, T_o, T_s, S_w, S_o, S_s)$ effectively and efficiently represents the sparse patterns of attention heads, we conduct a comprehensive ablation study. For a fair comparison, all experiments are conducted at a fixed sparsity level, and we evaluate the quality of the generated videos. The following five scenarios were tested:

1. **SVG-like** ($\mathcal{M}(T_w, S_w)$): This configuration simulates the approach of SVG, where only the width of the temporal and spatial attention bands can be controlled.
2. **STA-like** ($\mathcal{M}(T_w, S_w, S_s)$): This mask simulates the behavior of STA. STA’s method of applying sparsity along the tile’s t, w, and h axes corresponds in the attention heatmap to defining the inter-frame width (T_w), intra-frame width (S_w), and the stride of intra-frame diagonals (S_s).
3. **Inter-frame-Only** ($\mathcal{M}(T_w, T_o, T_s)$).
4. **Intra-frame-Only** ($\mathcal{M}(S_w, S_o, S_s)$).
5. **Ours** ($\mathcal{M}(T_w, T_o, T_s, S_w, S_o, S_s)$).

As shown in table 6, the full six-parameter model outperforms all ablated versions, achieving a PSNR of 30.35—2.5 dB higher than the next best (Inter-frame-only). This confirms that offset and stride parameters are essential for capturing multi-diagonal and hybrid patterns. Inter-frame-only (PSNR 27.81) significantly outperforms Intra-frame-only (24.01), showing that inter-frame relationships are more critical, yet only the full model captures both. Notably, our SVG-like configuration even outperforms the actual SVG in table 1, as PSA-Search assigns more accurate per-head patterns than SVG’s fixed two-pattern strategy.

5 Conclusion

We present PSA, a parameterized stripe attention framework for efficient video generation. Our central insight is that video DiT attention is not arbitrary but exhibits periodic diagonal stripe structures along both temporal and spatial dimensions. This structural understanding enables a unified parametric formulation $\mathcal{M}(T_w, T_o, T_s, S_w, S_o, S_s)$ that resolves the flexibility–efficiency tension inherent in prior methods: it covers all observed attention patterns while permitting a single hardware-efficient CUDA kernel achieving up to 61% MFU. A training-free offline search algorithm then assigns each head its maximum-sparsity mask under an error constraint, introducing no runtime overhead. Experiments on HunyuanVideo and Wan 2.1 demonstrate end-to-end speedups of $1.57\times$ and $1.37\times$ over FlashAttention-3 with minimal quality degradation.

Acknowledgments and Disclosure of Funding

This work was supported by Alibaba Group through Alibaba Research Intern Program. Like most work on large-scale video generation, developing and evaluating our method required substantial computational resources, which carries environmental costs and is currently feasible only for research groups with access to large GPU clusters. While our method is training-free—built on top of existing pretrained video generation models without requiring additional fine-tuning—and reduces per-inference compute and energy, the overall computational demands of video generation remain significant.

References

- [1] Adnan, M., Kurella, N., Arunkumar, A., Nair, P.J.: Foresight: Adaptive layer reuse for accelerated and high-quality text-to-video generation (2025), <https://arxiv.org/abs/2506.00329>
- [2] Chen, A., Dong, B., Li, J., Lin, J., Tian, K., Yao, Y., Wang, G.: Rainfusion: Adaptive video generation acceleration via multi-dimensional visual redundancy (2025), <https://arxiv.org/abs/2505.21036>
- [3] Chen, P., Zeng, X., Zhao, M., Ye, P., Shen, M., Cheng, W., Yu, G., Chen, T.: Sparse-vdit: Unleashing the power of sparse attention to accelerate video diffusion transformers. arXiv preprint arXiv:2506.03065 (2025)
- [4] Dao, T.: FlashAttention-2: Faster attention with better parallelism and work partitioning. In: International Conference on Learning Representations (ICLR) (2024)
- [5] Esser, P., Kulal, S., Blattmann, A., Entezari, R., Müller, J., Saini, H., Levi, Y., Lorenz, D., Sauer, A., Boesel, F., et al.: Scaling rectified flow transformers for high-resolution image synthesis. In: Forty-first international conference on machine learning (2024)
- [6] Hassani, A., Walton, S., Li, J., Li, S., Shi, H.: Neighborhood attention transformer. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 6185–6194 (2023)
- [7] Jacobs, S.A., Tanaka, M., Zhang, C., Zhang, M., Song, S.L., Rajbhandari, S., He, Y.: Deepspeed ullysses: System optimizations for enabling training of extreme long sequence transformer models (2023), <https://arxiv.org/abs/2309.14509>
- [8] Kong, W., Tian, Q., Zhang, Z., Min, R., Dai, Z., Zhou, J., Xiong, J., Li, X., Wu, B., Zhang, J., et al.: Hunyuanvideo: A systematic framework for large video generative models. arXiv preprint arXiv:2412.03603 (2024)
- [9] Labs, B.F.: Flux. <https://github.com/black-forest-labs/flux> (2024)
- [10] Li, Z., Zhang, J., Lin, Q., Xiong, J., Long, Y., Deng, X., Zhang, Y., Liu, X., Huang, M., Xiao, Z., Chen, D., He, J., Li, J., Li, W., Zhang, C., Quan, R., Lu, J., Huang, J., Yuan, X., Zheng, X., Li, Y., Zhang, J., Zhang, C., Chen, M., Liu, J., Fang, Z., Wang, W., Xue, J., Tao, Y., Zhu, J., Liu, K., Lin, S., Sun, Y., Li, Y., Wang, D., Chen, M., Hu, Z., Xiao, X., Chen, Y., Liu, Y., Liu, W., Wang, D., Yang, Y., Jiang, J., Lu, Q., et al.: Hunyuan-dit: A powerful multi-resolution diffusion transformer with fine-grained chinese understanding (2024)
- [11] Lipman, Y., Chen, R.T., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling. arXiv preprint arXiv:2210.02747 (2022)
- [12] Liu, Y., Hu, Y., Zhang, Z., Jiang, K., Yuan, K.: Mixture of distributions matters: Dynamic sparse attention for efficient video diffusion transformers (2026), <https://arxiv.org/abs/2601.11641>
- [13] Liu, Y., Zhang, Z., Cai, Y., Deng, R., He, Y., Yuan, K.: Ropeslr: 3d rope-driven sparse-lowrank attention for efficient diffusion transformers (2026), <https://arxiv.org/abs/2605.20659>
- [14] Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., Zhu, J.: Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. Advances in neural information processing systems **35**, 5775–5787 (2022)
- [15] Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., Zhu, J.: Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. Machine Intelligence Research pp. 1–22 (2025)
- [16] Lu, E., Jiang, Z., Liu, J., Du, Y., Jiang, T., Hong, C., Liu, S., He, W., Yuan, E., Wang, Y., et al.: Moba: Mixture of block attention for long-context llms. arXiv preprint arXiv:2502.13189 (2025)

- [17] Pan, B., Panda, R., Fosco, C., Lin, C.C., Andonian, A., Meng, Y., Saenko, K., Oliva, A., Feris, R.: Va-red²: Video adaptive redundancy reduction (2021), <https://arxiv.org/abs/2102.07887>
- [18] Peebles, W., Xie, S.: Scalable diffusion models with transformers. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 4195–4205 (2023)
- [19] Shah, J., Bikshandi, G., Zhang, Y., Thakkar, V., Ramani, P., Dao, T.: Flashattention-3: Fast and accurate attention with asynchrony and low-precision. *Advances in Neural Information Processing Systems* **37**, 68658–68685 (2024)
- [20] Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502* (2020)
- [21] Spector, B.F., Arora, S., Singhal, A., Fu, D.Y., Ré, C.: Thunderkittens: Simple, fast, and adorable ai kernels (2024), <https://arxiv.org/abs/2410.20399>
- [22] Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., Liu, Y.: Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing* **568**, 127063 (2024)
- [23] Sun, W., Tu, R.C., Ding, Y., Jin, Z., Liao, J., Liu, S., Tao, D.: Vorta: Efficient video diffusion via routing sparse attention. *arXiv preprint arXiv:2505.18809* (2025)
- [24] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. *Advances in neural information processing systems* **30** (2017)
- [25] Wan, T., Wang, A., Ai, B., Wen, B., Mao, C., Xie, C.W., Chen, D., Yu, F., Zhao, H., Yang, J., et al.: Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314* (2025)
- [26] Wang, X., Si, Q., Wu, J., Zhu, S., Cao, L., Nie, L.: Retake: Reducing temporal and knowledge redundancy for long video understanding (2025), <https://arxiv.org/abs/2412.20504>
- [27] Wu, J., Hou, L., Yang, H., Tao, X., Tian, Y., Wan, P., Zhang, D., Tong, Y.: Vmoba: Mixture-of-block attention for video diffusion models. *arXiv preprint arXiv:2506.23858* (2025)
- [28] Xi, H., Yang, S., Zhao, Y., Xu, C., Li, M., Li, X., Lin, Y., Cai, H., Zhang, J., Li, D., et al.: Sparse videogen: Accelerating video diffusion transformers with spatial-temporal sparsity. *arXiv preprint arXiv:2502.01776* (2025)
- [29] Xia, Y., Ling, S., Fu, F., Wang, Y., Li, H., Xiao, X., Cui, B.: Training-free and adaptive sparse attention for efficient long video generation (2025), <https://arxiv.org/abs/2502.21079>
- [30] Yang, Q., Wang, J., Li, X., Bai, Y., Tong, X., Zhen, H., Hao, J., Yuan, M., Li, B.: Why attention patterns exist: A unifying temporal perspective analysis (2026), <https://arxiv.org/abs/2601.21709>
- [31] Yang, S., Xi, H., Zhao, Y., Li, M., Zhang, J., Cai, H., Lin, Y., Li, X., Xu, C., Peng, K., et al.: Sparse videogen2: Accelerate video generation with sparse attention via semantic-aware permutation. *arXiv preprint arXiv:2505.18875* (2025)
- [32] Yang, Z., Teng, J., Zheng, W., Ding, M., Huang, S., Xu, J., Yang, Y., Hong, W., Zhang, X., Feng, G., et al.: Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072* (2024)
- [33] Ye, Z., Chen, L., Lai, R., Lin, W., Zhang, Y., Wang, S., Chen, T., Kasikci, B., Grover, V., Krishnamurthy, A., Ceze, L.: Flashinfer: Efficient and customizable attention engine for llm inference serving (2025), <https://arxiv.org/abs/2501.01005>
- [34] Zhang, F., Tian, S., Huang, Z., Qiao, Y., Liu, Z.: Evaluation agent: Efficient and promptable evaluation framework for visual generative models. *arXiv preprint arXiv:2412.09645* (2024)
- [35] Zhang, P., Chen, Y., Su, R., Ding, H., Stoica, I., Liu, Z., Zhang, H.: Fast video generation with sliding tile attention. *arXiv preprint arXiv:2502.04507* (2025)

- [36] Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR (2018)
- [37] Zhang, Y., Xing, J., Xia, B., Liu, S., Peng, B., Tao, X., Wan, P., Lo, E., Jia, J.: Training-free efficient video generation via dynamic token carving (2025), <https://arxiv.org/abs/2505.16864>
- [38] Zheng, Z., Peng, X., Yang, T., Shen, C., Li, S., Liu, H., Zhou, Y., Li, T., You, Y.: Open-sora: Democratizing efficient video production for all. arXiv preprint arXiv:2412.20404 (2024)

This appendix provides supplementary materials to support the main paper. We organize the content as follows:

Section A discusses the limitations of the current work and outlines potential directions for future research.

Section B presents the pseudocode implementation of PSA-`Kernel`, offering detailed algorithmic specifications for reproducibility.

Section C introduces the PSA-`Search` process, including how to initialize mask candidates and the search results.

Section D provides the formal proposition and derivation showing how video spatio-temporal locality alone produces periodic diagonal stripes in the attention heatmap.

Section E provides extensive empirical evidence, through large-scale visualization experiments on Wan 2.1 and HunyuanVideo, to validate the dominance assumption regarding m_λ^* proposed in section 3.1.

Section F demonstrates that CogVideoX-v1.5 exhibits attention heatmap characteristics consistent with our observations, confirming that our method can be effectively applied to accelerate computation in this model as well.

Section G visualizes attention heatmaps across various prompts, revealing that identical heads exhibit highly similar heatmap patterns regardless of input prompt variations.

Section H presents the loss threshold sensitivity analysis.

Section I presents a comprehensive evaluation of end-to-end performance and generation quality across varying sparsity levels.

Section J provides qualitative comparisons with the FlashAttention-3 baseline, showcasing visual quality assessments and demonstrating that our method maintains comparable generation quality while achieving significant speedups.

A Limitations

While PSA demonstrates substantial acceleration for DiTs, two limitations warrant discussion. First, the current work is limited to video generation models, and it remains uncertain whether the same approach transfers effectively to other model families or domains, such as language models. Second, the current PSA-`Search` requires 2.5 hours on 8 H100 GPUs to evaluate all candidate masks for every attention head, which can be costly. In future work, we plan to accelerate this process via a two-stage search strategy: a coarse-grained stage that rapidly filters out unpromising candidates, followed by a fine-grained stage that performs detailed evaluation only on the surviving mask candidates.

B Pseudocode for PSA-`Kernel`

Algorithm 2 outlines the execution flow of PSA-`Kernel`, which employs a producer-consumer model to maximize hardware utilization. The workload is partitioned between two concurrently operating warpgroups within each GPU thread block. The **producer warpgroup** interprets the sparse mask configuration $\mathcal{M}$ to identify required KV blocks and asynchronously loads them from HBM to shared memory. The **consumer warpgroups** perform block-wise attention computation on available KV blocks, including safe softmax and result accumulation. Double-buffering with `kv_ready` and `compute_done` flags coordinates the pipeline, effectively hiding memory latency behind computation.

C Process of PSA-`Search`

C.1 Pseudocode for PSA-`Search`

We present the detailed pseudocode of PSA-`Search` in algorithm 3. Given a list of candidate masks $\mathcal{C}_\mathcal{M}$ and a user-specified loss threshold α , the algorithm independently searches for the optimal mask

Algorithm 2 Pseudocode for PSA-Kernel

Input: Sparsity parameters $\mathcal{M}$, Tensors Q, K, V
Output: Result tensor Out
// Executed in parallel by dedicated warp groups within a thread block

```
1: procedure PRODUCER-WARPGROUP
2:   // Determine which KV blocks are needed for this Q block
3:    $KV_{list} \leftarrow \text{GetKVBlocks}(Q_{block}, \mathcal{M})$  ▷ Uses logic from algorithm 1
4:   for each  $KV_{block}$  in  $KV_{list}$  do
5:     wait until compute_done flag is set
6:     Reset compute_done flag
7:     Load  $KV_{block}$  from HBM into SRAM
8:     Set kv_ready flag
9:   end for
10: end procedure

11: procedure CONSUMER-WARPGROUP
12:   for  $k \leftarrow 1$  to  $|KV_{list}|$  do ▷ Loop terminates after processing all blocks
13:     wait until kv_ready flag is set
14:     Reset kv_ready flag
15:     // Perform attention on data in SRAM
16:      $S \leftarrow Q_{block} \times K_{SRAM}^T$  ▷ Compute attention scores
17:      $S_{padded} \leftarrow \text{3D-PADDING}(S, -\infty)$  ▷ Pad invalid positions with  $-\infty$ 
18:      $P \leftarrow \text{SAFESOFTMAX}(S_{padded})$  ▷ Apply safe softmax
19:      $O_{partial} \leftarrow P \times V_{SRAM}$  ▷ Weighted sum of values
20:     Accumulate  $O_{partial}$  into local output  $O_{local}$ 
21:     Set compute_done flag
22:   end for
23:   Write final accumulated  $O_{local}$  to global memory Out
24: end procedure
```

configuration for each attention head. For a given head h , PSA-Search first evaluates all candidate masks by computing the L_2 loss between the sparse attention output and the full attention output, and sorts the results in ascending order of loss. Starting from the lowest-loss mask as the initial best candidate, the algorithm then scans through all qualified masks whose loss does not exceed α and greedily replaces the current best with any mask that achieves higher sparsity. This design ensures that the selected mask maximizes sparsity while remaining within the acceptable quality budget. After processing all heads, the algorithm returns a complete mask strategy map $\mathcal{S}_{\mathcal{M}}$ that assigns the sparsified yet faithful mask to each head. Notably, since PSA-Search records the per-head loss for every candidate mask, adjusting α only requires re-executing lines 10–21 without repeating the costly L_2 loss evaluation, making the threshold tuning process highly efficient.

C.2 Initialization of Mask Candidates.

Before conducting PSA-Search, it is necessary to select appropriate masks for initialization. We choose the corresponding masks based on visualized heatmap results (fig. 1). In general, for a DiT with shape (T, H, W) and PSABlock with shape $(\text{PSABlock}_t, \text{PSABlock}_h, \text{PSABlock}_w)$, we configure $\mathcal{M}(T_w, T_o, T_s, S_w, S_o, S_s)$ such that $0 < T_w < T/\text{PSABlock}_t$, $T_o \in \{0, \pm 1, \pm 2\}$, $T_s \in \{1, 2, 3, \infty\}$, $0 < S_w < W/\text{PSABlock}_w$, $S_o \in \{0, \pm 1\}$, and $S_s \in \{1, 3, 5, 6, \infty\}$.

For example, in Wan 2.1, the DiT shape is $21 \times 45 \times 80$. After applying 3D-Padding to obtain $21 \times 48 \times 80$, we set the PSABlock size to $3 \times 8 \times 16$. Then, the ranges for $\mathcal{M}(T_w, T_o, T_s, S_w, S_o, S_s)$ are configured as follows: $T_w < 7$, $T_o \in \{0, \pm 1, \pm 2\}$, $T_s \in \{1, 2, 3, \infty\}$, $S_w < 5$, $S_o \in \{0, \pm 1\}$, $S_s \in \{1, 3, 5, 6, \infty\}$.

Similarly, for HunyuanVideo at 720×1280 resolution, the DiT shape is $33 \times 45 \times 80$. After 3D-Padding to $33 \times 48 \times 80$, we design the PSABlock as $3 \times 8 \times 16$. The ranges for $\mathcal{M}(T_w, T_o, T_s, S_w, S_o, S_s)$ are set as: $T_w < 11$, $T_o \in \{0, \pm 1, \pm 2\}$, $T_s \in \{1, 2, 3, \infty\}$, $S_w < 5$, $S_o \in \{0, \pm 1\}$, $S_s \in \{1, 3, 5, 6, \infty\}$.

Algorithm 3 PSA-Search

Input: List of candidate masks $\mathcal{C}_{\mathcal{M}}$, DiT model $\mathcal{D}$, Loss threshold α
Output: An optimized mask strategy $\mathcal{S}_{\mathcal{M}}$

```
1: function PSA-SEARCH( $\mathcal{D}, \mathcal{C}_{\mathcal{M}}, \alpha$ )
2:    $\mathcal{S}_{\mathcal{M}} \leftarrow \emptyset$  ▷ Initialize an empty mask strategy map
3:   for each head  $h$  in model  $\mathcal{D}$  do ▷ Iterate over all timesteps, layers, and heads
4:      $HeadResults \leftarrow []$ 
5:     for each mask  $\mathcal{M} \in \mathcal{C}_{\mathcal{M}}$  do
6:        $Loss \leftarrow \text{L2Loss}(\text{PSA}(h, \mathcal{M}), \text{FullAttn}(h))$ 
7:       Add  $(Loss, \mathcal{M})$  to  $HeadResults$ 
8:     end for
9:     Sort  $HeadResults$  by loss in ascending order.
10:     $Best\mathcal{M} \leftarrow HeadResults[0].mask$ 
11:    for each  $(L_i, \mathcal{M}_i)$  in  $HeadResults$  do
12:      if  $L_i \leq \alpha$  then
13:        if Sparsity( $\mathcal{M}_i$ ) > Sparsity( $Best\mathcal{M}$ ) then
14:           $Best\mathcal{M} \leftarrow \mathcal{M}_i$ 
15:        end if
16:      end if
17:    end for
18:     $\mathcal{S}_{\mathcal{M}}[h] \leftarrow Best\mathcal{M}$ 
19:  end for
20:  return  $\mathcal{S}_{\mathcal{M}}$ 
21: end function
```

For HunyuanVideo at 768×1280 resolution, the DiT shape is $30 \times 48 \times 80$ without 3D-Padding. We design the PSABlock as $6 \times 8 \times 8$, and the ranges for $\mathcal{M}(T_w, T_o, T_s, S_w, S_o, S_s)$ are configured as: $T_w < 5$, $T_o \in \{0, \pm 1, \pm 2\}$, $T_s \in \{1, 2, 3, \infty\}$, $S_w < 10$, $S_o \in \{0, \pm 1\}$, $S_s \in \{1, 3, 5, 6, \infty\}$.

It is worth noting that setting $T_s = \infty$ or $S_s = \infty$ indicates that there is only one diagonal in the inter-frame or intra-frame attention, respectively. In practice, it suffices to set T_s no less than the number of frames T and S_s no less than $H \times W$.

C.3 Wan 2.1 14B

To demonstrate the practical effectiveness of our framework, we analyzed the mask distribution on the Wan 2.1 14B model. Using a loss tolerance of $\alpha = 0.007$, our search achieved a significant 57.57% overall sparsity, as shown in table 7.

C.4 HunyuanVideo 768×1280

On the HunyuanVideo model at a 768×1280 resolution, we set the L_2 loss threshold to $\alpha = 0.018$ for PSA-Search. This resulted in a heterogeneous mask combination that achieved an overall sparsity of 60.33%. The distribution of the ten most prevalent mask configurations is presented in table 8.

C.5 HunyuanVideo 720×1280

For the HunyuanVideo model at a 720×1280 resolution, we applied PSA-Search using a loss threshold of $\alpha = 0.03$. This process yielded a mask combination achieving 59.74% overall model sparsity. table 9 summarizes the ten most frequently chosen mask configurations discovered during this search.

D Locality-Induced Periodic Diagonal Stripes

In section 3.1, we stated that video spatio-temporal locality alone can produce periodic diagonal stripes in the 2D attention heatmap. We now formalize this claim.

Table 7: Top 10 most frequent PSA configurations discovered by the search algorithm in Wan 2.1 14B using a search threshold of $\alpha = 0.007$.

Mask Configuration		Head Count	
(T_w, T_o, T_s)	(S_w, S_o, S_s)	# Heads	% Total
full window		32000	20.00
(7, 0, ∞)	(3, 0, 3)	25257	15.79
(1, 0, ∞)	(5, 0, 6)	11778	7.36
(7, 0, ∞)	(1, 0, 3)	11069	6.92
(7, 0, ∞)	(3, 0, 1)	9014	5.63
(7, 0, ∞)	(3, 0, 5)	6810	4.26
(7, 0, ∞)	(1, 0, 5)	5655	3.53
(7, 0, ∞)	(5, 0, 3)	5470	3.42
(1, 0, 3)	(5, 0, 6)	5088	3.18
(3, 0, ∞)	(5, 0, 6)	4622	2.89
All Other Configurations		43237	27.02
Total		160000	100.00

Table 8: Top 10 most frequent PSA configurations discovered for the HunyuanVideo model (768×1280 resolution) using a search threshold of $\alpha = 0.018$.

Mask Configuration		Head Count	
(T_w, T_o, T_s)	(S_w, S_o, S_s)	# Heads	% Total
full window		14404	20.00
(1, 0, ∞)	(10, 0, 6)	9213	12.80
(5, 0, ∞)	(1, 0, 1)	5494	7.63
(5, 0, ∞)	(1, 0, 3)	4889	6.79
(5, 0, ∞)	(3, 0, 3)	4651	6.46
(5, 0, ∞)	(3, 0, 1)	3828	5.32
(5, 0, ∞)	(1, 0, 5)	1789	2.48
(5, 0, ∞)	(1, 0, 6)	1701	2.36
(5, 0, ∞)	(5, 0, 1)	1567	2.18
(5, 0, ∞)	(7, 0, 6)	1522	2.11
All Other Configurations		22942	31.86
Total		72000	100.00

Proposition 1 (Locality-induced diagonal stripes). Under the row-major token ordering $i = t \cdot HW + h \cdot W + w$, the locality conditions in eq. (1) produce periodic diagonal bands in the 2D attention heatmap:

1. **Temporal locality** ($|\Delta t| < \varepsilon_t$): For tokens at the same spatial position (h, w) , the 1D index difference is $\Delta i = \Delta t \cdot HW$, forming diagonal bands with stride HW and half-width $\varepsilon_t \cdot HW$.
2. **Column-wise locality** ($|\Delta h| < \varepsilon_h$): For tokens in the same column within a frame, $\Delta i = \Delta h \cdot W$, forming diagonal bands with stride W and half-width $\varepsilon_h \cdot W$.
3. **Row-wise locality** ($|\Delta w| < \varepsilon_w$): For tokens in the same row within a frame, $\Delta i = \Delta w$, forming a continuous band of width $2\varepsilon_w$ along the main diagonal.

The composition of these conditions yields periodic diagonal stripes with stride determined by the token layout (HW, W) and width determined by the locality radii $(\varepsilon_t, \varepsilon_h, \varepsilon_w)$.

Discussion. This proposition shows that locality alone suffices to produce periodic diagonal stripes without invoking RoPE periodicity: the stride is determined by the token layout $(HW$ for inter-frame, W for intra-frame), and the width by the locality radii. For the 9.70% of heads without a dominant RoPE channel (see section E), stripes arise purely from this mechanism. For heads with a dominant

Table 9: Top 10 most frequent PSA configurations for the HunyuanVideo model (720×1280 resolution) using a search threshold of $\alpha = 0.03$.

Mask Configuration		Head Count	
(T_w, T_o, T_s)	(S_w, S_o, S_s)	# Heads	% Total
full window		14400	20.00
(11, 0, ∞)	(1, 0, 1)	6223	8.64
(1, 0, ∞)	(5, 0, 6)	4937	6.86
(3, 2, ∞)	(5, 0, 6)	3995	5.55
(11, 0, ∞)	(3, 0, 1)	3964	5.51
(11, 0, ∞)	(1, 0, 3)	3511	4.88
(11, 0, ∞)	(3, 0, 3)	2898	4.03
(1, 2, 2)	(5, 0, 6)	2385	3.31
(3, 1, ∞)	(5, 0, 6)	2348	3.26
(3, 0, ∞)	(5, 0, 6)	2090	2.90
All Other Configurations		25249	35.06
Total		72000	100.00

channel, RoPE periodicity further modulates the stride to $\mathcal{T}_\lambda$ (eq. (5)) and introduces nonzero offset via $\phi^{(m_\lambda^*)}$, as described in the Joint effect paragraph of section 3.1.

E Empirical Support for m_λ^* in RoPE-3D

To empirically validate the existence of a predominant channel $m_\lambda^* \in M_\lambda$ that satisfies eq. (3) for structured attention patterns, we visualize the RoPE-channel contribution distributions of attention across a large number of attention heads in Wan 2.1 and HunyuanVideo, as shown in fig. 5. For each attention head, the RoPE-channel weights are computed by performing a dot product between the query and key vectors along each row, normalizing the resulting weights, and subsequently averaging over all rows at each corresponding position. The observations are summarized as follows:

1. Figures (a)–(b) illustrate the RoPE-channel weight distributions associated with the intra-frame pattern. In this case, the dominant channel m^* is consistently localized within M_w . Moreover, as the channel index within the M_w dimension increases, the spacing between adjacent intra-frame diagonal stripes increases accordingly, which corroborates the periodicity formulation in eq. (5).
2. Figures (c)–(d) illustrate the RoPE-channel weight distributions associated with the inter-frame pattern. Here, the dominant channel m^* is consistently localized within M_t . Analogously, as the channel index within the M_t dimension increases, the spacing between adjacent inter-frame diagonal stripes widens accordingly, consistent with the behavior described in eq. (5).
3. Figure (e) corresponds to the hybrid pattern. Under this regime, pronounced dominant channels m^* simultaneously emerge across multiple sub-groups M_t , M_h , and M_w , reflecting a superposition of multiple distinct attention patterns.
4. Figure (f) corresponds to the uniform pattern. Under this regime, the weight distributions across all channels exhibit a disordered and non-structured configuration, with no discernible dominant channel.

Quantitatively, we measure the dominance ratio as $\max_m w_m / \text{median}_m w_m$ for each head, where w_m denotes the normalized RoPE-channel weight. On Wan 2.1, **87.55%** of all attention heads exhibit a dominance ratio exceeding $3\times$, confirming that the single-channel dominance assumption underlying eq. (3) holds for the vast majority of heads. Through experimental observation, among the remaining 12.45% of heads, **2.74%** correspond to the uniform pattern, while the remaining **9.70%** still exhibit diagonal stripe structures. These stripes arise from the spatio-temporal locality ($\varepsilon_t, \varepsilon_h, \varepsilon_w$) described in eq. (1), and can likewise be efficiently sparsified by PSA. As formalized in section D, locality alone produces periodic diagonal stripes with stride determined by the token layout (HW, W), confirming that these heads are not exceptions but a natural sub-case of the unified stripe structure. To verify that PSA effectively handles these locality-driven heads, we separately

evaluate the 9.70% of heads without a dominant RoPE channel. At 60% sparsity, these heads achieve an average cosine similarity of **0.9872** with full attention, confirming that the locality-driven stripes are equally well captured by $\mathcal{M}$.

Although the uniform pattern lacks a dominant m^* , PSA-Search effectively handles these heads: the discovered masks fall into two categories—wide-stripe configurations (large T_w or S_w) and checkerboard patterns (periodic small-stride masks)—both of which achieve faithful approximations of the uniform distribution (cosine similarity > 0.98) while maintaining meaningful sparsity. This confirms that the PSA mask space is expressive enough to cover all four observed pattern types.

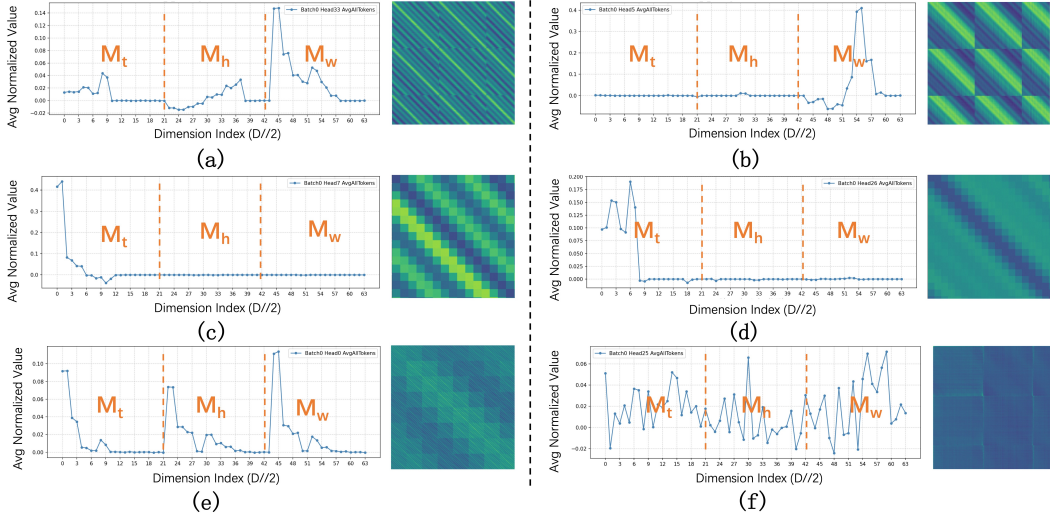


Figure 5: Empirical support for m_λ^* in RoPE-3D

F Heatmaps of Different Heads in CogVideoX

As shown in fig. 6, we analyze various patterns in the attention heatmaps of CogVideoX, which exhibit behaviors consistent with HunyuanVideo and Wan 2.1. Four distinct pattern types are identified:

- Figures (b), (e)-(h) demonstrate **intra-frame attention patterns**. These patterns are characterized by diagonal structures with varying widths, strides, and offsets within each frame, with the pattern repeating identically across all frames.
- Figures (a), (d) showcase **inter-frame attention patterns**. In these patterns, attention weights within each frame are relatively uniform, while diagonal band structures with varying widths, strides, and offsets emerge between frames.
- Figure (i) presents a **hybrid pattern**, which can be interpreted as a combination of intra-frame and inter-frame attention patterns.
- Figure (c) displays **uniform pattern**, where attention is uniformly distributed across all spatio-temporal positions.

Through further examination of the CogVideoX heatmap, we observe that it demonstrates sparse patterns analogous to those exhibited by Wan 2.1 and HunyuanVideo. The underlying reason is that these models all utilize DiT architectures for video generation, maintain structural similarities, and optimize toward identical objectives. Consequently, their weight distributions should also converge to comparable configurations, suggesting that the periodic locality characteristic of attention maps is universally applicable across DiT-based video generation scenarios. This demonstrates the general adaptability of PSA.

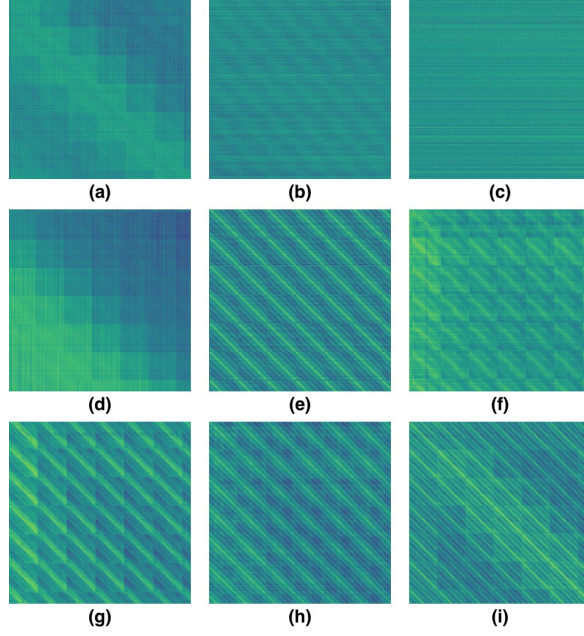


Figure 6: CogVideoX Heatmap Patterns.

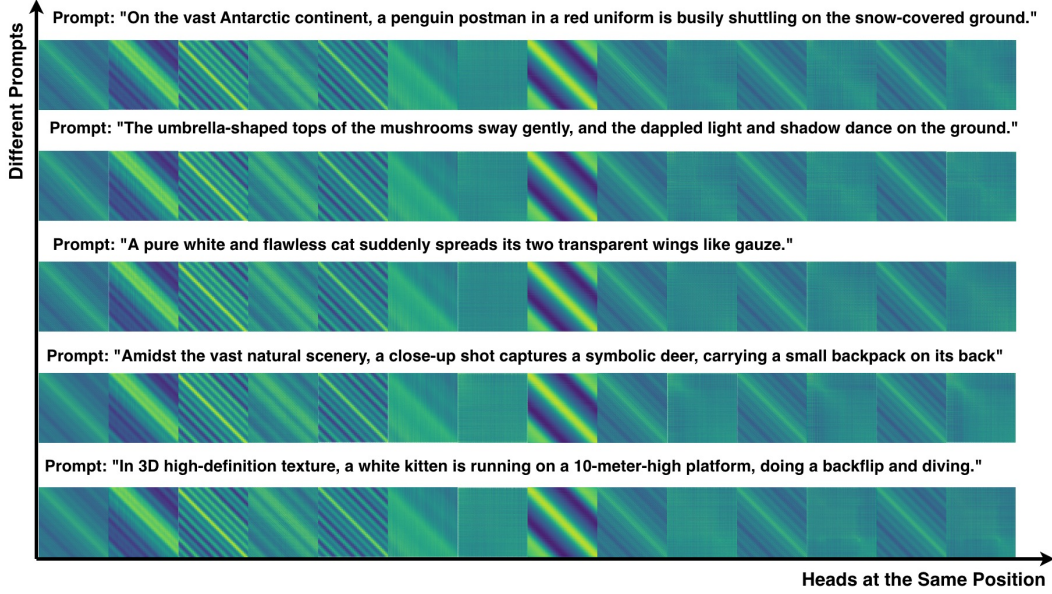


Figure 7: Heatmap Comparison Between Prompts.

G Heatmap Visualization Across Different Prompts

As shown in fig. 7, it can be intuitively observed that the heatmaps of the same head across different prompts exhibit extremely high similarity (cosine similarity > 0.99).

H Loss Threshold Sensitivity Analysis

As illustrated in fig. 8, we perform loss threshold sensitivity analysis on various α values for HunyuanVideo and Wan 2.1. The experimental results indicate that larger α values correspond to higher tolerable loss thresholds, which given the design principle of PSA-Search, result in increased

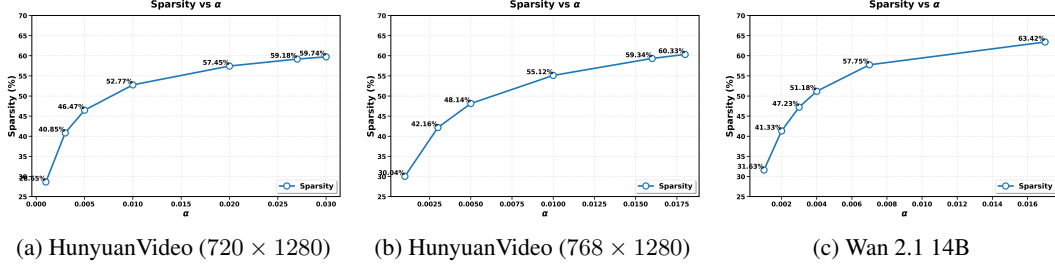


Figure 8: Sparsity variation with different α values across different models.

sparsity. To maintain sparsity at approximately 60%, we configured α as 0.03, 0.018, and 0.007 for HunyuanVideo (720×1280), HunyuanVideo (768×1280), and Wan 2.1, respectively. This observation reveals that Wan 2.1 exhibits higher sensitivity to loss, thereby necessitating a smaller α value to attain equivalent sparsity levels.

Furthermore, we obtain α values at various sparsity levels through our adaptive search algorithm for reference. As shown in table 10, setting the corresponding alpha value for a given sparsity level ensures optimal acceleration while minimizing the loss. Note that changing α does not require a time-consuming re-search, as PSA-Search saves the `HeadResults` for each head; consequently, after each adjustment of α , only the re-execution of algorithm 3 (lines 10–21) is needed, taking approximately 2 minutes.

Table 10: Alpha-sparsity correspondence for different models.

HunyuanVideo (720×1280)		HunyuanVideo (768×1280)		Wan 2.1 14B	
Sparsity (%)	α	Sparsity (%)	α	Sparsity (%)	α
18.72	0.0003	20.59	0.0003	19.65	0.0003
28.65	0.0010	30.04	0.0010	31.63	0.0010
40.85	0.0030	42.16	0.0030	42.05	0.0021
49.73	0.0070	51.66	0.0070	51.51	0.0041
62.80	0.0600	60.33	0.0180	61.22	0.0110
67.22	0.6000	69.09	0.0600	65.46	0.3000

I End-to-End Performance and Quality Across Varying Sparsity Levels

Beyond our initial analysis at 60% sparsity to match STA’s fixed sparsity (58.37%) for a direct comparison, we evaluate 10–70% sparsity (table 11). Our method consistently outperforms SVG2 in speed/quality, and achieves a better speed–quality trade-off than SVG. SVG only achieves marginal speedup at 70% sparsity while suffering severe quality degradation.

J Qualitative Comparison with Baselines

As shown in figs. 9 to 11, we execute multiple prompts across several models. By comparing the extracted video frames with the baseline using FlashAttention-3, we can observe that the quality is nearly lossless.

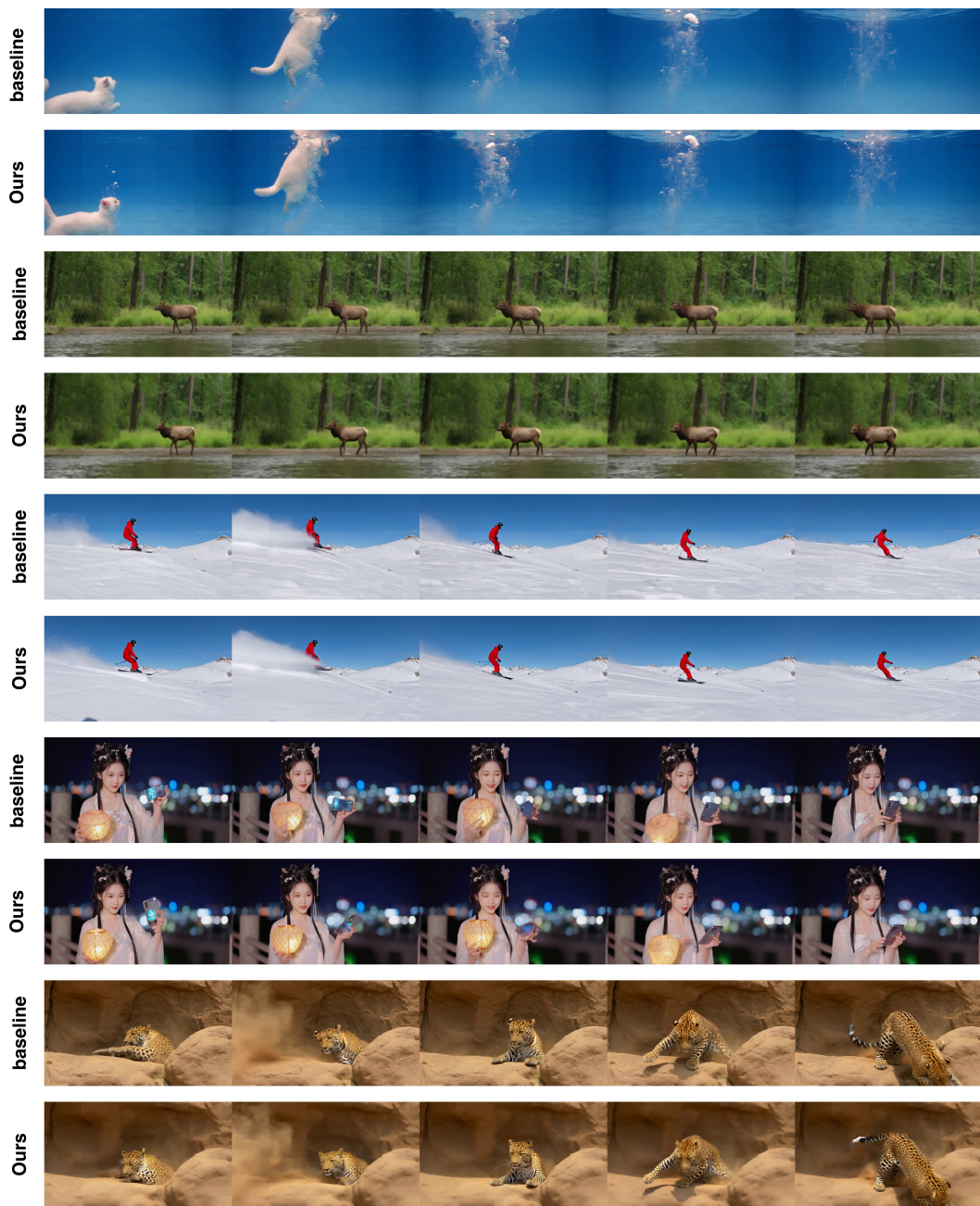


Figure 9: HunyuanVideo(720×1280) FA3 Baseline vs. Ours.

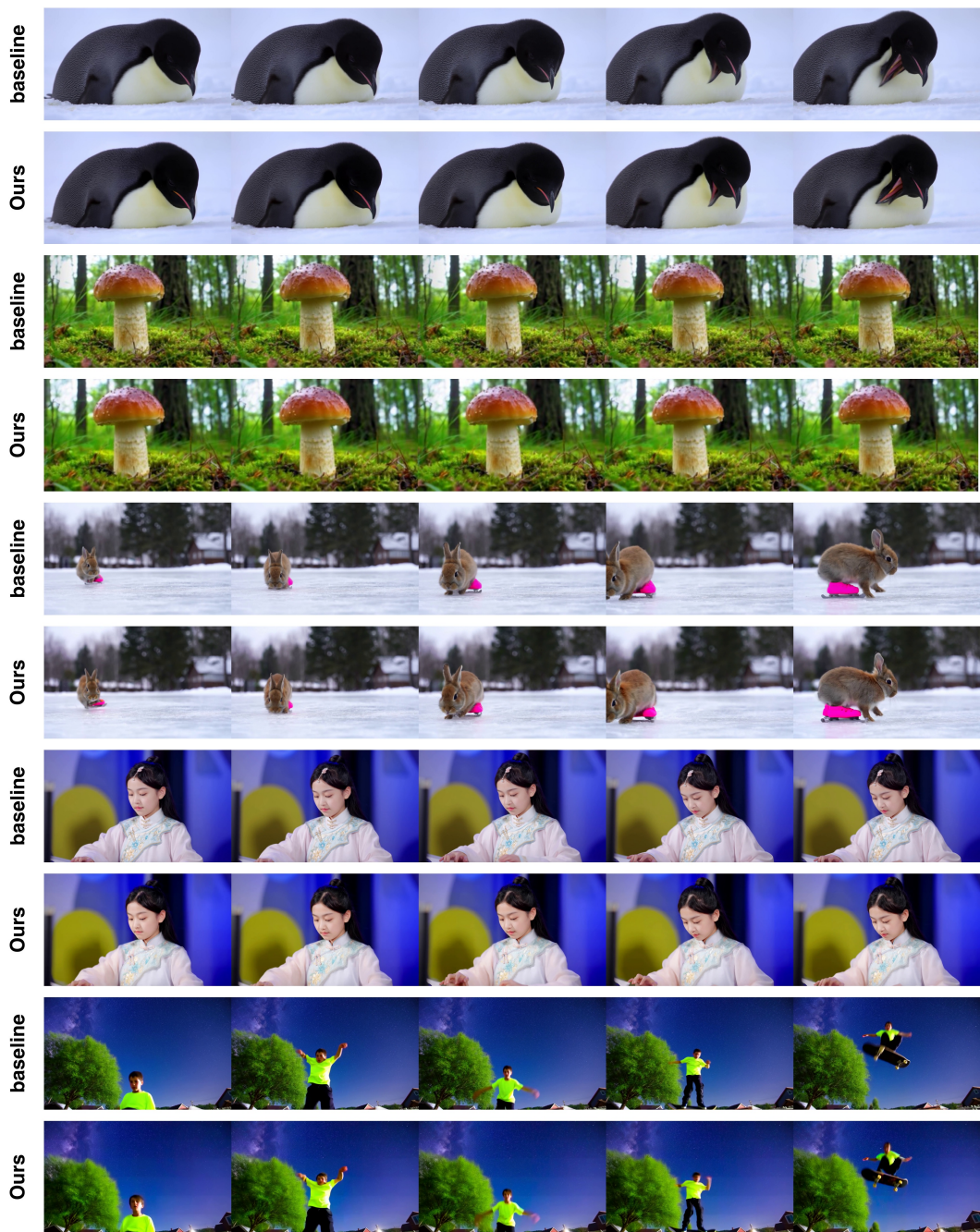


Figure 10: HunyuanVideo(768×1280) FA3 Baseline vs. Ours.

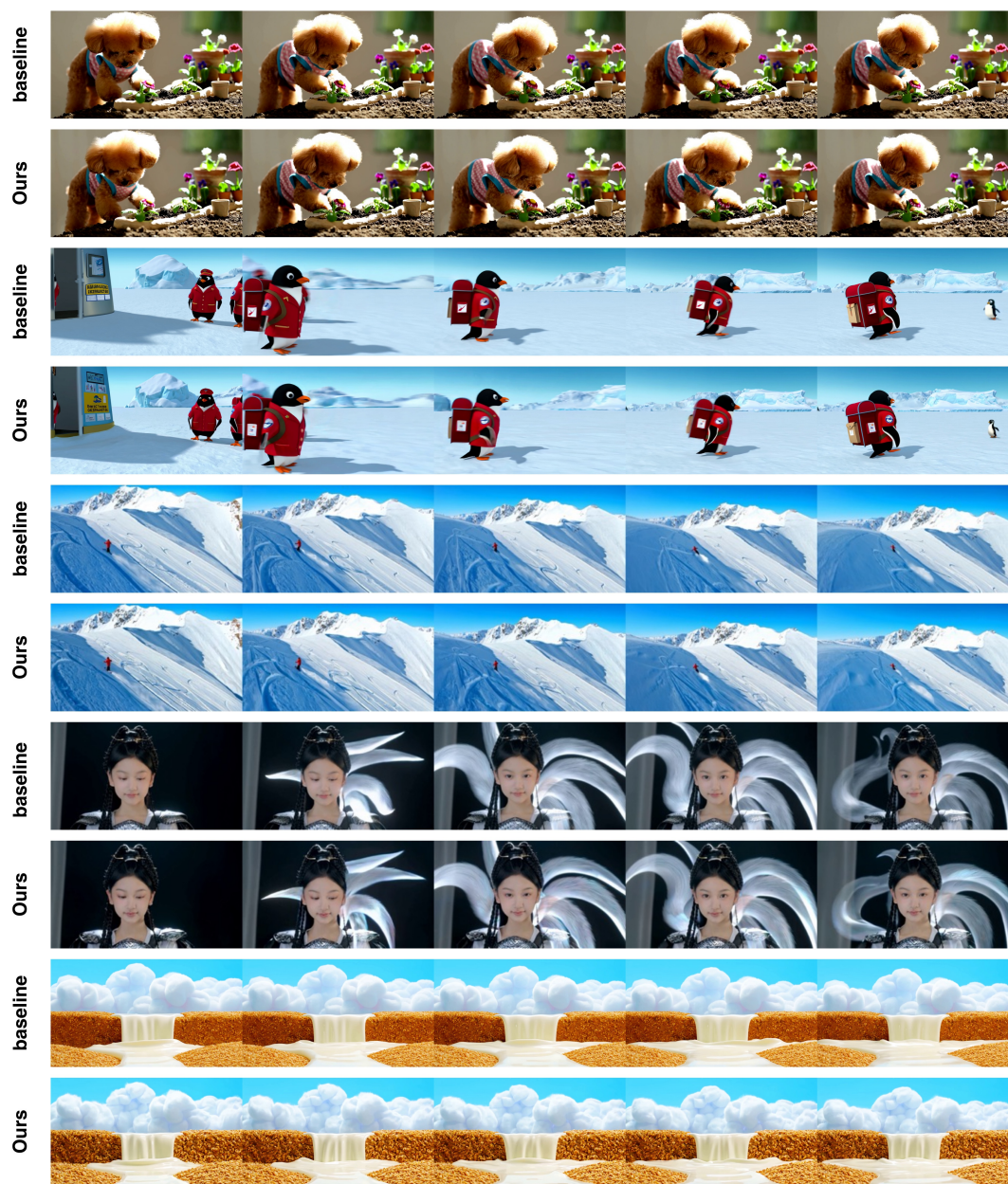


Figure 11: Wan 2.1(720×1280) FA3 Baseline vs. Ours.

Table 11: End-to-end performance and quality comparison across varying sparsity on HunyuanVideo (768×1280).

Method	Sparsity	E2E(s)	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$
SVG	10.48%	1393.25	30.34	0.9144	0.1247
	30.39%	1116.87	27.68	0.8862	0.1552
	50.32%	865.78	25.67	0.8494	0.1914
	70.16%	622.37	21.21	0.7771	0.2651
SVG2	10.49%	1333.93	31.87	0.9076	0.1206
	30.72%	1129.91	26.75	0.7753	0.2040
	50.67%	945.26	26.18	0.7314	0.2352
	70.93%	694.07	24.54	0.6610	0.2985
Ours	11.28%	1057.19	38.83	0.9647	0.0526
	30.04%	941.44	33.99	0.9318	0.0812
	51.66%	776.87	30.21	0.8952	0.1244
	69.09%	632.86	24.58	0.8220	0.2340

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 abstract and introduction clearly state three contributions of PSA (parameterized representation, efficient kernel, automatic search), which are consistent with the theoretical and experimental results in Sections 3–4.

2. Limitations

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

Answer: [Yes]

Justification: The paper discusses limitations in Appendix A.

3. Theory assumptions and proofs

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

Answer: [Yes]

Justification: The RoPE-3D attention logit decomposition (Eq. 1) and the dominant channel assumption (Eqs. 2–4) are presented with clear assumptions and derivations in Section 3.1.

4. Experimental result reproducibility

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

Answer: [Yes]

Justification: The paper provides complete algorithm pseudocode (Algorithms 1–3), model configurations (Section 4.1), search parameter ranges (Appendix B), and kernel implementation details (Appendix A), sufficient to reproduce the main experiments.

5. Open access to data and code

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

Answer: [Yes]

Justification: The code is publicly available at <https://github.com/jxyjason/PSA>, with sufficient instructions to faithfully reproduce the main experimental results.

6. Experimental setting/details

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

Answer: [Yes]

Justification: Section 4.1 provides detailed descriptions of model configurations, evaluation metrics, baseline methods, search parameters, and experimental settings.

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 randomness in video generation evaluation primarily stems from initial noise, while our comparison is between deterministic sparse attention and full attention outputs, so error bars are not required.

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: The paper states that all experiments are conducted on a single NVIDIA H100 GPU, and mentions in Section 3.3 that the search completes in 2.5 hours on 8 H100 GPUs.

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: This research is an efficiency optimization method for video generation and does not involve human subjects, sensitive data, or ethical risks.

10. Broader impacts

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

Answer: [N/A]

Justification: This research is a foundational efficiency optimization method that accelerates inference of existing models and is not directly tied to specific application deployments, with no direct path to negative societal impacts.

11. Safeguards

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

Answer: [N/A]

Justification: This paper does not release new pre-trained models or datasets; it only provides an attention efficiency optimization method.

12. Licenses for existing assets

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

Answer: [Yes]

Justification: All existing assets used in this paper are properly credited, and their licenses and terms of use are explicitly stated and fully respected.

13. New assets

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

Answer: [Yes]

Justification: The code is released at <https://github.com/jxyjason/PSA> with documentation provided in the repository; no other new assets are involved.

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: This paper does not involve crowdsourcing or research with human subjects.

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: This paper does not involve research with human subjects.

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 are not a component of the core method in this research; the core method is sparse attention optimization based on parameterized stripe attention.