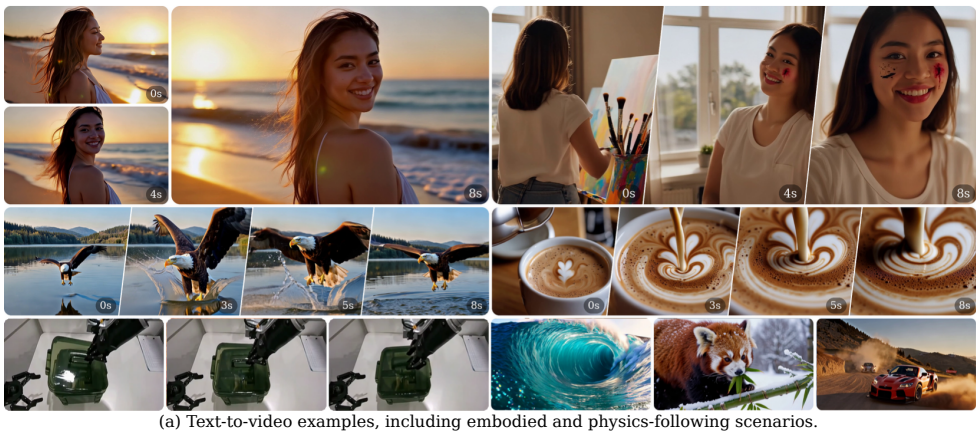
SANA-VIDEO 2.0: HYBRID LINEAR ATTENTION WITH ATTENTION RESIDUALS FOR EFFICIENT VIDEO GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce SANA-Video 2.0, a 5B hybrid video diffusion transformer for efficient video generation on a single GPU, reaching quality competitive with far larger full-softmax video DiTs at substantially lower measured latency. **Hybrid Linear–Softmax Attention** applies gated linear attention in three out of every four layers and reserves the fourth for gated global softmax, reducing quadratic attention cost while providing less rank-constrained token mixing at periodic depths. **Block Attention Residuals (AttnRes)** route completed block summaries into later linear layers, raising effective rank in deep linear-attention states by approximately 20% at modest overhead. Through **from-scratch training**, SANA-Video 2.0 learns the hybrid backbone directly, with reduced-resolution proxy studies identifying 25% softmax as a practical quality–efficiency trade-off. With 40-step sampling, the 5B model achieves VBench 84.30 in 13.2s at 480p on one H100. Its compiled DiT forward is **3.2× faster** than a matched full-softmax baseline at a 720p/60s tensor shape, with increasing speedups over the evaluated durations. In separate deployment measurements, Sol-Engine optimization (kernel fusion, caching, and sparse attention) delivers a **3.58×** end-to-end speedup at 720p/8s on B200 and reaches 13.06s at 720p/5s on one H100, approximately **120× faster** than Wan 2.2-A14B at the matched shape.



(a) Text-to-video examples, including embodied and physics-following scenarios.

(b) Generation-pipeline latency (s), one H100 (720p, 5s).

(c) 5B DiT-forward (s) vs. duration.

Figure 1: **SANA-Video 2.0 at a glance.** (a) Text-to-video examples, including embodied and physics-following scenarios. (b) One-H100 720p/5s latency with Sol-Engine 5B. VBench markers come from separate base evaluations: our 84.30 uses 480p/81f without Sol-Engine; quality and latency are not jointly evaluated. (c) 5B DiT-forward speedup over duration.

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

1 INTRODUCTION

Video generation is a rapidly advancing field, powering applications from creative content production to virtual product displays and live streaming. Recent large-scale generators, including Wan 2.1 (Wan Team et al., 2025) (14B), HunyuanVideo (Kong et al., 2024) (13B), Seedance 2.0 (Seedance et al., 2026), and closed systems such as Veo 3 and Kling, produce remarkably high-fidelity video. Among open Video DiT (Peebles & Xie, 2023) baselines such as Wan 2.1 and HunyuanVideo, full 3D softmax attention incurs an $O(N^2)$ cost that is punishing for video: after VAE compression a single 1080p clip already spans tens of thousands of latent tokens, and in our matched production-scale profiling sweep a full-softmax backbone is $2.01\times$ slower than our 25% hybrid at 1080p/121f, with both sides compiled on their best kernels (Figure 5(a)). The problem worsens for long video ($>$10s), where the quadratic attention matrix and activation workspace dominate the cost; even dedicated long-video efforts such as MAGI-1 (Sand.ai et al., 2025) remain constrained by vanilla attention.

Linear attention offers a principled escape. Its $O(N)$ complexity scales gracefully with sequence length, its state-space view exposes a clean interface for recurrent kernels, and SANA-Video (Chen et al., 2025) showed that a *pure*-linear Video DiT already delivers large speedups at competitive quality. But pure linear attention pays for its efficiency with expressiveness: the fixed-size state matrix $S \in \mathbb{R}^{d\times d}$ cannot represent every token–token interaction, which can weaken precise spatiotemporal correspondence and fine detail (Fan et al., 2025; Arora et al., 2024). Recent large language models point to a fix: keep a mostly linear-attention stack but insert a few softmax *anchors*—periodic global-attention layers that provide less rank-constrained token mixing at fixed depths—at a regular ratio (Qwen3-Next (Qwen Team, 2025), Kimi-Linear (Kimi Team, 2025)), and route information across depth with Attention Residuals (Kimi Team, 2026b), a combination the recent Kimi K3 (Kimi Team, 2026a) adopts at trillion-parameter scale. This raises a central question: *can a mostly-linear backbone retain competitive video generation quality while substantially reducing the cost of global attention for long, high-resolution video?*

This paper answers affirmatively with SANA-Video 2.0, a 5B hybrid-attention video diffusion transformer. The model keeps a mostly-linear backbone but inserts a small number of softmax anchors, achieving competitive generation quality while remaining fast on a single GPU (Figure 1): with 40-step sampling it reaches VBench Total 84.30 in 13.2s at $480\times832\times81$ on one H100, competitive with much larger softmax models at a fraction of their latency, and its DiT forward is $3.2\times$ faster than a matched full-softmax DiT at a 720p/60s shape with both compiled on their best kernels (Figure 1(c)), an advantage that widens over the evaluated durations. Unlike post-hoc linearization of a pretrained softmax model, we train the hybrid backbone directly. The design rests on three key components.

Hybrid Linear-Softmax Attention. We keep gated bilinear linear attention as the core token-mixing operation for its $O(N)$ cost, and interleave gated softmax anchors at a regular 3:1 ratio (25% softmax) placed at fixed depths. The anchors provide less rank-constrained token mixing, while the linear majority reduces reliance on quadratic global attention. Text enters through cross-attention, and a convolution-free SwiGLU FFN replaces SANA-Video's temporal-convolution FFN, whose measured overhead grows with duration (Appendix G.2), keeping the backbone easy to profile and fuse with off-the-shelf kernels.

Block Attention Residuals (AttnRes). Additive residuals alone force later layers to re-derive information computed upstream. Instead of this re-derivation, AttnRes routes completed block-feature summaries across depth, so later linear layers can reuse features from blocks containing softmax anchors. AttnRes targets that bottleneck directly: a same-checkpoint on/off probe over 24 prompts raises deep-layer state effective rank by $\sim$20%, at modest overhead in the reported probes. Isolating an end-task effect would need a second from-scratch production run, so we validate the mechanism the design targets. We adapt the routing to bidirectional video diffusion by sharing routing queries across depth and dropping the explicit timestep input that AdaLN makes redundant.

From-Scratch Training and Design. We select the architecture with short, reduced-resolution proxy studies that identify 25% softmax anchors as a practical quality–efficiency knee. The hybrid backbone is then trained from scratch through a complete, documented multi-stage pipeline, comprising diverse-source data curation and filtering, a low-to-high resolution and duration curriculum, structured captioning, and self-distillation, followed by preference-based post-training

(DPO and ReFL), so the hybrid learns motion and appearance without converting a pretrained softmax backbone. We report this full pipeline as a central, reproducible contribution.

In conclusion, SANA-Video 2.0 attains competitive VBench quality (84.30) at markedly lower latency than large softmax Video DiTs, and its mostly-linear backbone achieves increasing speedups over the matched all-softmax baseline across the evaluated durations. A separate 50-step Sol-Engine deployment measures a $3.58\times$ end-to-end speedup on B200, and quantization-aware training matches BF16 quality at MXFP4 weights and MXFP8 activations (Section 6). Rank and routing analyses reveal a clear division of labor, where linear layers provide inexpensive global mixing, softmax anchors provide less rank-constrained token mixing, and AttnRes carries block-level features across depth. We hope SANA-Video 2.0 offers a practical, efficient foundation for high-quality video generation that researchers and everyday users can run fast.

2 PRELIMINARIES

Video DiTs and flow matching. Video Diffusion Transformers (Peebles & Xie, 2023) can be trained with conditional optimal-transport flow matching (Lipman et al., 2023): they form a latent sequence $z_t = (1-t)z + t\epsilon$ ($\epsilon \sim \mathcal{N}(0, I)$, $t \sim$ logit-normal) and predict the conditional velocity $\epsilon - z$ with loss $\mathcal{L} = \mathbb{E}\|v_\theta - (\epsilon - z)\|^2$. Here $t \in [0, 1]$ denotes the normalized flow-matching timestep. Each block uses AdaLN-style timestep modulation, self-attention, text cross-attention, and a FFN. At 1080p a long clip spans tens of thousands of latent sites, making softmax's $O(N^2)$ cost prohibitive.

Gated linear attention. Linear attention compresses token interactions into a fixed-size state, reducing sequence complexity to $O(Nd^2)$ at the cost of a rank bottleneck. Our bidirectional operator is a gated bilinear form rather than a causal delta-rule recurrence; dropping the delta-rule update also makes it a natural initialization for a causal Gated DeltaNet, a direction we leave to future work. For normalized queries and keys, let q_j^r, k_n^r denote their RoPE-rotated forms and $\phi(\cdot) = \mathrm{ReLU}(\cdot)$. Each head computes

$$S = \sum_n v_n(\beta_n k_n^r)^\top, \qquad o_j = W_o\left[\mathrm{RMSNorm}\left(\frac{Sq_j^r}{\sum_n \phi(k_n)^\top \phi(q_j) + \epsilon}\right) \odot \sigma(g_j)\right], \quad (1)$$

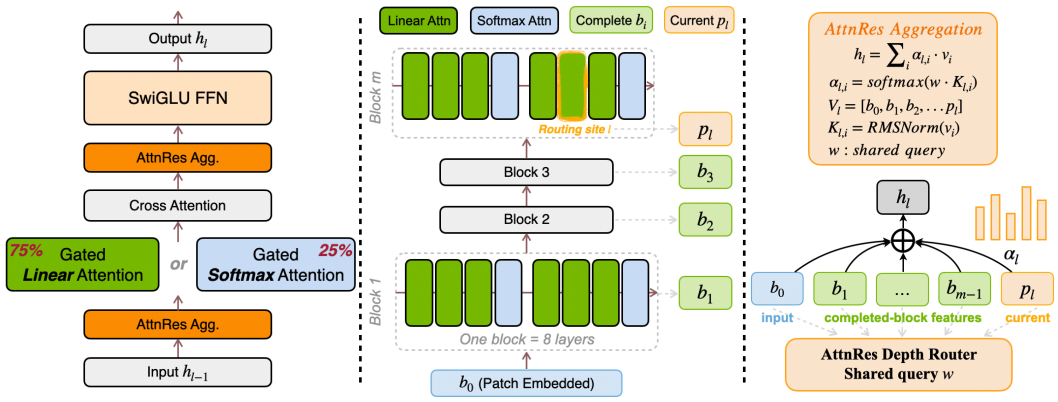
where β gates writes to the state and g gates the output. Periodic global softmax layers provide less rank-constrained token mixing at selected depths, while most layers use a linear-attention operator. These anchors use bidirectional SDPA with QK normalization, RoPE, and the sigmoid output gate studied by Qiu et al. (2025). We adopt the regular 3:1 hybrid layout of Qwen3-Next (Qwen Team, 2025); Kimi-Linear (Kimi Team, 2025) independently uses the same proportion with a different operator.

Attention Residuals. Standard additive residuals $h_l = h_{l-1} + f_l(h_{l-1})$ propagate information through every intervening depth. Attention Residuals (Kimi Team, 2026b) instead use learned depth-wise aggregation, $h_l = \sum_i \alpha_{i\to l} v_i$. Their Block AttnRes variant keeps completed block summaries and a running within-block residual, and applies separate routing before attention and FFN sublayers. In our hybrid stack, completed summaries include softmax-anchor updates and expose them to later mostly-linear layers. The original parameterizes routing queries per sublayer. Our video adaptation instead shares them across depth. Rather than assume the routing queries need a separate diffusion-timestep input, we test whether one is necessary and find it redundant with AdaLN.

3 SANA-VIDEO 2.0

3.1 OVERVIEW

SANA-Video 2.0 is a video DiT that combines hybrid sequence attention with block residual attention across depth. It operates on LTX-VAE 2.3 (HaCohen et al., 2024) latents and draws text features from Gemma-2-2B-IT (Gemma Team, 2024) through cross-attention at every layer. At each depth, AttnRes first routes a representation into the self-/cross-attention branch and, in a second step, into the SwiGLU FFN, with AdaLN-style modulation inside each branch, and



(a). Hybrid DiT Module (b). Residual Attention Aggregation (c). AttnRes Module

Figure 2: **SANA-Video 2.0 overview.** (a) Hybrid DiT layer. (b) Eight-layer blocks expose completed features across depth. (c) Shared-query AttnRes routing combines the input, completed block summaries, and the current partial sum before each sublayer.

a final aggregation precedes the output head. The local path is convolution-free throughout. Appendix B lists the full configuration.

3.2 Hybrid Attention Design

Building on the gated attention primitives (Section 2), we interleave bidirectional gated linear attention layers with gated softmax anchors at a 3:1 ratio, 75% linear, 25% softmax, placing a softmax anchor at every fourth layer. Most layers use token mixing with linear sequence complexity, while global softmax anchors provide interactions unconstrained by the bilinear operator's per-head feature-rank bound. This does not guarantee full rank for every input. The full hybrid retains a quadratic term in sequence length; reducing the number of global softmax layers lowers its measured cost. Linear and softmax heads use different head dimensions, trading efficiency against per-head capacity (Appendix B). This regular layout follows the hybrid linear attention of recent LLMs (Qwen3-Next (Qwen Team, 2025), Kimi-Linear (Kimi Team, 2025), and Kimi K3 (Kimi Team, 2026a) at trillion-parameter scale) and maps cleanly onto fused linear-attention kernels. We confirm the 25% anchor ratio as a quality–efficiency knee for the video regime by sweeping it from scratch rather than assuming the language-model value (Section 5.3.1). The two paths retain separate RoPE (Su et al., 2024) tensors and gating parameterizations: linear layers use both the write gate β and an output gate, whereas softmax anchors use the sigmoid output gate of Qiu et al. (2025).

3.3 Block Attention Residuals (AttnRes)

Figure 2 summarizes the hybrid layer and its block-level depth routing. Hybrid attention supplies less rank-constrained token mixing only at sparse depths; AttnRes complements it by exposing completed block summaries containing those updates to later layers. We build on the Block AttnRes (Kimi Team, 2026b), which Kimi K3 (Kimi Team, 2026a) pairs with hybrid linear attention at language-model scale, and adapt it to bidirectional video diffusion. The layers are grouped into consecutive blocks: each block that has finished by layer l leaves a single feature summary, and the block still in progress carries a running partial sum. The router at layer l therefore draws on a source set $\mathcal{V}_l = \{b_0, b_1, \ldots, p_l\}$ made of the initial token embedding b_0, the completed block summaries $b_1, b_2, \ldots$ from blocks that finished before layer l, and the current within-block partial sum p_l, the accumulation so far in the in-progress block (absent only at a block boundary, where it has just been frozen into a new summary and reset). For each video token x, the routed representation before sublayer type τ is

$$h_l(x) = \sum_{v_i \in \mathcal{V}_l} \alpha^{(\tau)}_{i \to l}(x)\, v_i(x), \qquad \alpha^{(\tau)}_{i \to l}(x) = \underset{i}{\mathrm{softmax}}\Big(\big(w^{(\tau)} + \phi_\tau(t)\big)^\top \mathrm{RMSNorm}(v_i(x)) \Big), \tag{2}$$

where $v_i(x)$ is the feature that source $v_i \in \mathcal{V}_l$ holds at token position x, $w^{(\tau)}$ is shared by all depths for $\tau \in \{\text{attn}, \text{ffn}\}$, and $\phi_\tau(t)$ is an optional, zero-initialized timestep offset. The softmax is over depth sources independently for every token. After the last block, a third shared query aggregates the initial embedding and completed block sums into the final representation passed to the output head.

Shared routing queries. The original AttnRes learns a separate routing query for every residual sublayer, a per-layer design that language models keep even at scale (e.g., Kimi K3 (Kimi Team, 2026a)). For our video model, one shared routing query per branch suffices: a single attention query and a single FFN query, reused at every depth. This matches per-layer routing in loss at a fraction of the memory (Section 5.3.2). Sharing the query does not make routing uniform. The source set $\mathcal{V}_l$ changes with depth, so the routing weights still vary from layer to layer, and because each source is normalized per token, they also vary from token to token. Sharing thus drops the per-depth query parameters while still routing the attention and FFN branches differently.

Block organization. We group every $S{=}8$ consecutive transformer layers into one block. As the forward pass moves through a block, each attention or FFN sublayer's output, the residual update it would otherwise add to the hidden state, is added into a running partial sum p_l, which the router also reads back as a source (Eq. 2). When the block ends, p_l is frozen as that block's completed summary b_k, and the next block starts a fresh partial sum. Because the router keeps only one summary per finished block rather than one feature per layer, the stored history shrinks from $O(NLd)$ to $O(N\lceil L/S\rceil d)$ for sequence length N, roughly an S-fold reduction. The span of $S{=}8$ covers two softmax-anchor cycles while keeping this history compact, and the ablation treats it as an engineering default rather than a universal optimum (Section 5.3.2). As in Block AttnRes, attention and FFN routers learn distinct preferences (Section F.5).

Timestep-independent final design. Although denoising timesteps have different semantic roles, the learned offset is nearly constant and changing or shuffling it has negligible validation effect. The final model therefore uses $\phi_\tau \equiv 0$ while retaining timestep modulation in the DiT blocks (Section 5.3.2, Appendix F.4).

4 Training Recipe

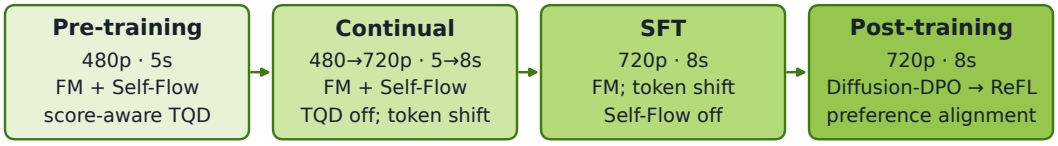


Figure 3: **Training curriculum for the 5B model.** Quality/motion-scored, captioned clips feed progressively stricter training pools; Appendix C details the data pipeline.

The 5B backbone trains from scratch with flow matching (Figure 3). Curation scores quality and motion on separate axes rather than collapsing them into one bar, which keeps selection from drifting toward visually clean but nearly static clips, and feeds a progressive funnel from broad pre-training coverage to a trusted SFT subset, with captions densifying along the way. Post-training applies Diffusion-DPO (Wallace et al., 2024), then online ReFL (Xu et al., 2023b).

The noise schedule follows the same logic. From the continual stage onward the flow shift is token-count-aware, interpolating log-linearly from 3 at 4,290 latent tokens to 6 at 23,000, so every resolution×duration combination lands on a principled shift rather than a discrete per-resolution constant; in pre-training, score-aware TQD (Luo et al., 2026) instead routes motion-rich clips toward high noise, where texture is masked anyway, and clean low-motion clips toward low noise. Self-Flow (Chefer et al., 2026) is used only before SFT. We monitor checkpoints with timestep-stratified validation rather than a scalar random-t mean, which cuts the standard deviation of the early-to-final estimate by $2.27\times$. This recipe was selected jointly, without component-wise ablations; Appendix H gives the full settings and protocols.

Table 1: **VBench quality comparison.** Green intensity ranks the top three results per metric. Superscripts after method names denote score sources and are defined in Appendix D. **Attention** gives each backbone's token-mixing scheme: full 3D softmax, mixture-of-transformers (MoT), all-linear, or our linear–softmax hybrid. Only the 81-frame SANA-Video 2.0 result is shown here; the appendix also provides the full protocols, extended baselines, and our 121- and 193-frame operating points.

Model	Params	Attention	Frames	Total↑	Quality↑	Semantic↑
LTX-2.3 (base)†	22B	Full 3D	121	81.95	83.85	74.35
Cosmos-3 Nano†	16B	Full 3D	93	83.13	84.29	78.50
Lance (Fu et al., 2026)†	7.1B	MoT	81	83.18	84.09	79.55
Wan 2.1	1.3B	Full 3D	81	83.31	85.23	75.65
HunyuanVideo⋆	13B	Full 3D	129	83.43	85.07	76.88
Wan 2.1	14B	Full 3D	81	83.69	85.59	76.11
SANA-Video	2B	Linear	81	84.17	84.85	81.46
Wan 2.2⋆	A14B	Full 3D	81	84.23	85.42	79.50
Bernini-R‡	14B	Full 3D	81	**84.64**	85.18	**82.49**
SANA-Video 2.0	5B	Hybrid	81	84.30	**85.61**	79.05

5 Experiments

5.1 Implementation Details

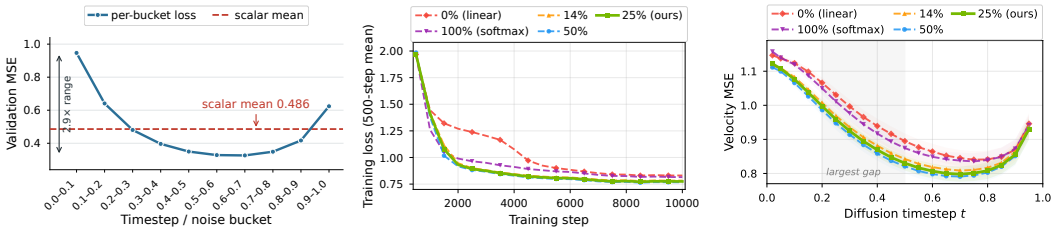
SANA-Video 2.0 uses a 5B hybrid backbone with 32 layers at width 2,560; Appendix Table 3 lists its architecture and training settings. The model uses LTX-VAE 2.3 latents, Gemma text features, flow matching, 25% softmax anchors, and Block AttnRes. To select the architecture and diagnose mechanisms rather than estimate final quality, we use short, reduced-resolution studies: the attention-ratio and early AttnRes sweeps use depth-28, width-3,072 architecture-search backbones at 256p, while the query/timestep ablations and mechanism probes use 5B architecture-family checkpoints (all on 8×H100 GPUs).

5.2 Main Results

Table 1 compares SANA-Video 2.0 on VBench (Huang et al., 2024); Figure 1(b) compares one-H100 pipeline latency including the accelerated Sol-Engine deployment, which the VBench scores evaluate separately. At its $480\times832\times81$ operating point, SANA-Video 2.0 reaches VBench Total 84.30 with the best Quality in Table 1 (85.61). Only the author-reported Bernini-R 14B (Bernini Team et al., 2026) scores higher on Total and Semantic ($84.64/82.49$ vs. $84.30/79.05$), yet our 5B model is $31.8\times$ cheaper at a matched shape, step count, and device (13.2s vs. 421s on one H100). The SANA-Video comparison is not controlled: it initializes from an image model, which favours text following, whereas SANA-Video 2.0 trains from scratch. Against SANA-Video, SANA-Video 2.0 is higher on Quality and on dynamic degree (78.9 vs. 69.2) and lower on Semantic, where 79.05 still exceeds LTX-2.3, Wan 2.1 at both scales, HunyuanVideo, and Cosmos-3 Nano. The same checkpoint is stronger at longer shapes: 121 frames reaches Total 85.29 and Quality 86.98, above every entry in Table 1, and 193 frames gives 84.48 (Appendix Table 5); these are not frame-matched, so the table leads with 81 frames. At a 720p-class shape it stays 3–4× faster than Cosmos-3 Nano and LTX-2.3 and roughly $50\times$ faster than Bernini-R and Wan 2.2-A14B. Full per-shape timings and the complete 16-dimension breakdown are in Appendix D.

5.3 Design Space Exploration

We first explore the sequence-attention ratio with depth-28, width-3,072 architecture-search proxies, trained from scratch at 256p/81 frames. The sweep fixes the backbone dimensions, patch size, attention head dimensions, data, optimizer settings, seed, and training recipe while changing the softmax-layer placement; we report training curves and validation loss on 200 held-out videos with text conditioning at 10K steps. Because validation loss is strongly non-uniform



(a) Loss varies with t. (b) Training loss vs. step. (c) Validation loss vs. t.

Figure 4: **Reading loss by timestep, then the softmax-ratio proxy.** (a) On a production checkpoint, validation loss spans a $\sim 3\times$ range across the noise axis, so the scalar random-t mean (dashed) hides structure. We therefore read every proxy comparison per timestep rather than as one scalar. (b,c) Fixed depth-28, width-3,072 architecture-search backbones (256p/81f): (b) 500-step mean training loss and (c) held-out normalized-latent velocity MSE at 10K (200 videos; pointwise 95% CIs) for five ratios.

across the noise axis (Figure 4a), we read every proxy comparison per timestep rather than as a single scalar. The subsequent AttnRes probes follow the per-panel protocols in Table 2; comparisons are made only within each matched panel.

5.3.1 Softmax Ratio

We train five attention-ratio variants from 0% (all-linear) to 100% (all-softmax), including three hybrids (Figure 4). The lowest-ratio hybrid has four periodic softmax anchors among 28 layers, giving a softmax fraction of $4/28{=}14.3\%$.

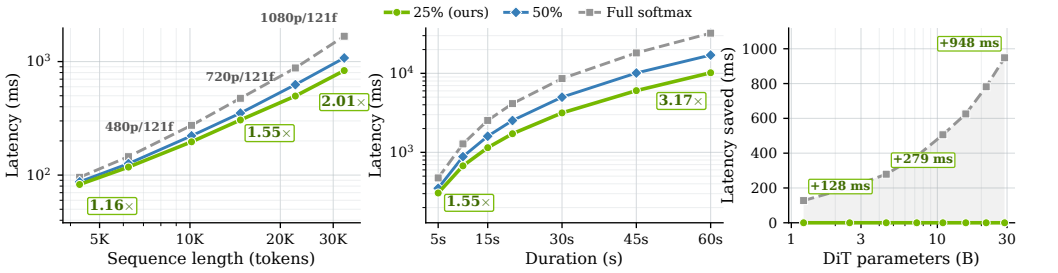
Finding: Hybrid ratios perform better than the all-linear and all-softmax endpoints in this fixed-backbone-dimension proxy search. Pure linear attention is the weakest endpoint (all-linear val loss 0.955), consistent with the absence of periodic softmax correction. All-softmax is also worse than the hybrids (0.945), with the two endpoints close and the linear one slightly worse. Among hybrids, 50% achieves the lowest loss (0.897) but at steep efficiency cost ($1.29\times$ latency over 25% hybrid at 1080p/121f; Figure 5(a)), while 25% (0.905) and the four-anchor (14.3%) hybrid (0.914) trail by only $\sim 1\%$. Together, Figures 4 and 5(a) identify 25% softmax as a practical Pareto knee rather than the minimum-loss point. Per-timestep analysis shows the hybrid–endpoint differentiation concentrates at mid noise ($t \approx 0.2$–0.5).

Lock-in: **25% softmax (3:1 ratio)** for all subsequent experiments. This is the measured proxy Pareto knee, aligns with AttnRes block boundaries ($S{=}8$), and matches the Qwen3-Next layout (Qwen Team, 2025); Kimi-Linear (Kimi Team, 2025) independently uses the same proportion with a different operator.

Scratch-training stability. All curves in Figure 4b are trained from scratch; the hybrid variants converge smoothly with no loss spike or collapse and continue into the production training curriculum. The 3:1 ratio is therefore learned directly rather than obtained by post-hoc linearization.

5.3.2 Residual Attention

With the 3:1 ratio fixed, Table 2 separates late-stage continuation from early design probes (protocols in Appendix I). Removing the explicit timestep input improves early loss, while shared queries match per-layer loss at one-quarter of the memory overhead. The block-span sweep gives similar loss point estimates and similar latency and memory overheads across spans; $S{=}8$ covers two anchor cycles at the middle of that range, without a statistically established optimum. In this continuation probe the mean held-out MSE is nearly tied (0.4851 vs. 0.4855), but read per bucket as above the direction is consistent: AttnRes is lower in 17 of 20 noise buckets. Both arms branch from one mature hybrid ancestor, so the probe still asks whether removing the router degrades a trained model rather than what training with it from scratch provides, which would need a second production run. We therefore also validate AttnRes against the deficiency it targets, the rank bottleneck of linear-attention states (Section 5.5).



(a) Compiled resolution scaling. (b) Compiled 720p/24fps scaling. (c) Compiled 720p/161f model-size scaling.

Figure 5: **Hybrid-attention DiT-forward scaling** (H100, bf16, batch 1; no AttnRes). (a) Compiled resolution scaling. (b) Compiled 720p/24fps duration scaling; its 121f point is shared with (a). (c) Full-softmax minus 25%-hybrid latency across model sizes at 720p/161f (19.3K tokens), with the same compiled backend in two run orders. Only (a,b) include 50% softmax.

Table 2: **AttnRes training and design probes.** (a) Late-stage continuation; (b–d) early short-run probes. Wins counts improved noise buckets; green shading marks the selected designs.

(a) Router effect

Design	MSE↓	Wins
No AttnRes	0.48547	—
+ AttnRes	**0.48506**	**17/20**

(b) Explicit t input

Setting	Loss↓
with t	0.962
without t	**0.920**

(c) Query sharing

Design	Loss↓	Mem
per-layer	0.496	+16%
shared	**0.495**	**+4%**

(d) Block span (S)

S	Loss↓	Lat.	Mem
4	0.965	+3.9%	+2.9%
8	0.962	+3.1%	+2.1%
16	0.962	+3.1%	+1.6%

5.4 Efficiency Scaling

Resolution and router scaling. The proxy studies select a shared, timestep-independent router with S=8. We compare the 25%-softmax backbone with full softmax under one compiled, no-AttnRes protocol in which every variant uses its best kernels (fused linear attention, FlashAttention for all softmax layers). This is the standard protocol for all DiT-forward comparisons in the paper.

Sequence- and model-size scaling. Figure 5(a) compares six 480p–1080p shapes and three attention ratios. Under the compiled best-kernels no-AttnRes protocol, full/25%-hybrid speedup rises from $1.16\times$ to $2.01\times$. At fixed 720p and 24fps it reaches $3.17\times$ at the 60s tensor shape (1,441 frames, Figure 5(b)). Under the same protocol, panel (c) fixes the 720p/10s shape and jointly grows width and depth from 1.2B to 28.9B. The hybrid is faster at all eight scales, and its absolute saving grows from 128 to 948ms. A controlled decomposition (Appendix G.1) separates the axes: depth leaves the speedup nearly flat ($1.41\times$), while width shifts it from $1.45\times$ to $1.33\times$ as width-bound FFN and projection cost outpaces attention. The hybrid advantage is thus governed by how sequence-bound the compute is, exactly the regime long video occupies.

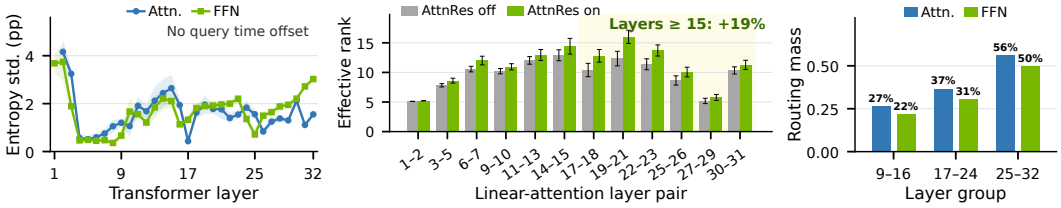
5.5 Mechanistic Analysis

Routing without an explicit timestep input. A late checkpoint trained with $\phi_\tau \equiv 0$ still varies its routing across denoising stages: over 20 captioned clips with clip and noise fixed across five timesteps, normalized routing entropy has temporal standard deviation 0.0164 ± 0.0011 (mean and across-clip standard deviation; Figure 6a; protocol in Appendix F.3). Separately, replacing the learned timestep offset in an ablation checkpoint by its mean or evaluating it at $1 - t$ changes validation loss by at most 0.0002 (Appendix F.4). These probes support dropping the explicit query offset while retaining timestep-dependent routing through DiT features.

Linear-state effective rank. The ratio sweep shows that periodic softmax anchors improve the proxy quality–efficiency tradeoff. We next test whether AttnRes realizes the complementary goal of carrying richer representations across depth. Using effective rank, the exponential entropy of the linear-state singular spectrum, we toggle depth aggregation on and off on the same production checkpoint under matched prompts and noise seeds. At maximum noise, averaged over 24 prompts (seeds 0–3 each), enabling AttnRes increases the mean effective rank of deep linear-attention states

by $19.7\% \pm 5.1\%$, with non-decreasing rank in 23 of 24 linear layers (Figure 6b). This metric characterizes the states' singular spectra; token-attention rank and generation quality require separate evaluation. The static router adds a negligible parameter cost (well below 0.001% of the model).

Cross-depth reuse. The ten-clip diagnostic assigns 56%/50% of deep-layer attention/FFN routing mass to completed blocks, with the nearest source receiving 26.0%/25.3% (Figure 6c). A 24-prompt production-checkpoint probe reproduces it (52.9%/46.9%). Removing completed sources cuts hidden-representation effective rank by 81–88% at block entries, where the partial sum has reset, but not mid-block (Appendix F.6), localizing the reuse to block boundaries. Because summaries mix attention and FFN updates, the intervention does not isolate the contribution of softmax alone.



(a) Timestep variation. (b) Same-checkpoint rank recovery. (c) Cross-block reuse.

Figure 6: **Evidence for the selected AttnRes design.** (a) Routing varies across timesteps in a checkpoint trained without an explicit query offset (20 clips; shading: across-clip standard deviation). (b) Enabling depth aggregation recovers effective rank in deeper linear layers. (c) Learned routing increasingly reuses completed blocks.

6 Deployment and Applications

Standard attention and FFN primitives let the backbone compose with Sol-Engine (Li et al., 2026b): kernel fusion, caching, and sparse anchors give $3.58\times$ end-to-end speedup at 720p/8s on B200 and 13.06s at 720p/5s on one H100, $120\times$ faster than Wan 2.2-A14B at the matched shape (Figure 1(b); VBench 84.30 is a separate base evaluation). MXFP4/MXFP8 QAT matches BF16 VBench Total with 67.9% less storage. Appendix K reports deployment, quantization, and Physical AI results.

7 Related Work

Open video generators are largely diffusion transformers with quadratic softmax token mixing (Wan Team et al., 2025; Kong et al., 2024; Yang et al., 2025; Zheng et al., 2025), while MAGI-1 and LTX-Video reduce effective sequence cost through temporal chunking or stronger latent compression (Sand.ai et al., 2025; HaCohen et al., 2024). Efficient token mixers instead replace the $N \times N$ map with a fixed-size state (Katharopoulos et al., 2020; Yang et al., 2024; Arora et al., 2024; Gu & Dao, 2024; Liu et al., 2024), though compressed states restrict representation rank (Fan et al., 2025); in visual generation this runs from image DiTs (Zhu et al., 2025) to pure-linear video DiTs (Chen et al., 2025; Zhu et al., 2026). Hybrid language models keep periodic softmax layers at a 3:1 ratio and route across depth with Attention Residuals (Qwen Team, 2025; Kimi Team, 2025; 2026a;b), whereas Attention Surgery linearizes a pretrained video DiT post hoc (Ghafoorian et al., 2025); sparse methods reduce the cost within each remaining softmax operator (Xi et al., 2025; Li et al., 2025; 2026a; Yuan et al., 2025). SANA-Video 2.0 brings hybrid attention with AttnRes to bidirectional video diffusion, trains it from scratch, and re-selects the softmax ratio for video (Appendix A).

8 Conclusion

We introduced SANA-Video 2.0, a scratch-trained 5B video diffusion transformer combining 75% gated linear attention, 25% softmax anchors, and depth-shared Block AttnRes. Proxy studies identify 25% softmax as a quality–efficiency knee, and same-checkpoint probes over 24 prompts show AttnRes enables cross-depth reuse and $\sim$20% higher deep linear-state effective rank. The model reaches competitive VBench quality with speedups over full softmax that grow with sequence length. Three limits remain: architecture selection uses short proxy studies, the longest-duration results are tensor-shape latency, and global softmax keeps a quadratic term.

THE USE OF LARGE LANGUAGE MODELS (LLMS)

LLMs were used as a writing and coding aid: polishing prose and grammar in text the authors had already written, and helping draft and check analysis, plotting, and verification scripts. The research ideas, model design, training recipe, and the interpretation of results are the authors' own. Separately, and as documented in the method, Gemini is used offline to rank sampled videos into the preference pairs used by DPO (Appendix H.4), and the model conditions on frozen Gemma-2-2B-IT text features. The authors take full responsibility for the paper.

REPRODUCIBILITY STATEMENT

We have aimed to make the work reproducible from the paper alone. The complete architecture and training hyperparameters for the 5B model are listed in Appendix B, and the staged training recipe, data-curation pipeline, noise schedule, and post-training procedure are described in Section 4 and Appendix C. The evaluation protocol, including sampling settings, the shapes at which each result is measured, and the timing methodology, is specified in Appendix D; all quality numbers use the public VBench benchmark. The profiling protocol for every latency measurement, including warmup counts, the number of timed iterations, and the kernels used for each attention variant, is given in Appendix D.2. The mechanism analyses report their sample sizes explicitly: the effective-rank and routing-ablation probes are averaged over 24 prompts, and the metric definitions and intervention procedures are given in Appendix F.7 and Appendix F.6.

ETHICS STATEMENT

This work studies the efficiency of video generation architectures and does not involve human subjects, personally identifying information, or sensitive attributes. The training corpus is filtered for quality and defects as described in Appendix C, and evaluation uses the public VBench benchmark. We note the dual-use considerations that apply to generative video models in general: lowering the cost of high-resolution video synthesis also lowers the cost of producing misleading or non-consensual synthetic media. The efficiency contributions here are orthogonal to content control, and we regard provenance measures such as watermarking and generated-content detection, along with the usage restrictions attached to released models, as necessary complements to the efficiency gains reported in this paper.

REFERENCES

Simran Arora, Sabri Eyuboglu, Michael Zhang, Aman Timalsina, Silas Alberti, Dylan Zinsley, James Zou, Atri Rudra, and Christopher Ré. Simple Linear Attention Language Models Balance the Recall-Throughput Tradeoff. In *ICML*, 2024.

Bernini Team, Chenchen Liu, Junyi Chen, Lei Li, Lu Chi, Mingzhen Sun, Zhuoying Li, Yi Fu, Ruoyu Guo, Yiheng Wu, Ge Bai, and Zehuan Yuan. Bernini: Latent Semantic Planning for Video Diffusion. arXiv preprint arXiv:2605.22344, 2026.

Shuo Cao, Jiayang Li, Xiaohui Li, Yuandong Pu, Kaiwen Zhu, Yuanting Gao, Siqi Luo, Yi Xin, Qi Qin, Yu Zhou, Xiangyu Chen, Wenlong Zhang, Bin Fu, Yu Qiao, and Yihao Liu. UniPercept: Towards Unified Perceptual-Level Image Understanding across Aesthetics, Quality, Structure, and Texture. In *ICML*, 2026.

Hila Chefer, Patrick Esser, Dominik Lorenz, Dustin Podell, Vikash Raja, Vinh Tong, Antonio Torralba, and Robin Rombach. Self-Supervised Flow Matching for Scalable Multi-Modal Synthesis. arXiv preprint arXiv:2603.06507, 2026.

Junsong Chen, Yuyang Zhao, Jincheng Yu, Ruihang Chu, Junyu Chen, Shuai Yang, Xianbang Wang, Yicheng Pan, Daquan Zhou, Huan Ling, Haozhe Liu, Hongwei Yi, Hao Zhang, Muyang Li, Yukang Chen, Han Cai, Sanja Fidler, Ping Luo, Song Han, and Enze Xie. SANA-Video: Efficient Video Generation with Block Linear Diffusion Transformer. arXiv preprint arXiv:2509.24695, 2025.

Yukang Chen, Luozhou Wang, Wei Huang, Shuai Yang, Bohan Zhang, Yicheng Xiao, Ruihang Chu, Weian Mao, Qixin Hu, Shaoteng Liu, Yuyang Zhao, Huizi Mao, Ying-Cong Chen, Enze Xie, Xiaojuan Qi, and Song Han. LongLive-2.0: An NVFP4 Parallel Infrastructure for Long Video Generation. arXiv preprint arXiv:2605.18739, 2026.

Qihang Fan, Huaibo Huang, and Ran He. Breaking the Low-Rank Dilemma of Linear Attention. In *CVPR*, 2025.

Fengyi Fu, Mengqi Huang, Shaojin Wu, Yunsheng Jiang, Yufei Huo, Hao Li, Yinghang Song, Fei Ding, Jianzhu Guo, Qian He, Zheren Fu, Zhendong Mao, and Yongdong Zhang. Lance: Unified Multimodal Modeling by Multi-Task Synergy, 2026. URL https://arxiv.org/abs/2605.18678.

Gemma Team. Gemma 2: Improving Open Language Models at a Practical Size. arXiv preprint arXiv:2408.00118, 2024.

Mohsen Ghafoorian, Denis Korzhenkov, and Amirhossein Habibian. Attention Surgery: An Efficient Recipe to Linearize Your Video Diffusion Transformer. arXiv preprint arXiv:2509.24899, 2025.

Albert Gu and Tri Dao. Mamba: Linear-Time Sequence Modeling with Selective State Spaces. In *Conference on Language Modeling (COLM)*, 2024.

Yoav HaCohen, Nisan Chiprut, Benny Brazowski, Daniel Shalem, et al. LTX-Video: Realtime Video Latent Diffusion. arXiv preprint arXiv:2501.00103, 2024.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench: Comprehensive Benchmark Suite for Video Generative Models. In *CVPR*, 2024.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention. In *ICML*, 2020.

Kimi Team. Kimi Linear: An Expressive, Efficient Attention Architecture. arXiv preprint arXiv:2510.26692, 2025.

Kimi Team. Kimi K3: Open Frontier Intelligence. Moonshot AI technical blog, https://www.kimi.com/blog/kimi-k3, 2026a.

Kimi Team. Attention Residuals. arXiv preprint arXiv:2603.15031, 2026b.

Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, et al. HunyuanVideo: A Systematic Framework For Large Video Generative Models. arXiv preprint arXiv:2412.03603, 2024.

Haopeng Li, Shitong Shao, Wenliang Zhong, Zikai Zhou, Lichen Bai, Hui Xiong, and Zeke Xie. PISA: Piecewise Sparse Attention Is Wiser for Efficient Diffusion Transformers. arXiv preprint arXiv:2602.01077, 2026a.

Kunchang Li, Xinhao Li, Yi Wang, Yinan He, Yali Wang, Limin Wang, and Yu Qiao. VideoMamba: State Space Model for Efficient Video Understanding. In *ECCV*, 2024.

Xingyang Li et al. Radial Attention: $O(n \log n)$ Sparse Attention with Energy Decay for Long Video Generation. arXiv preprint arXiv:2506.19852, 2025.

Yitong Li, Junsong Chen, Haopeng Li, Haozhe Liu, Jincheng Yu, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Sol Video Inference Engine: Agent-Native Full-Stack Acceleration Framework for Efficient Video Generation. arXiv preprint arXiv:2606.23743, 2026b.

Zhi Li, Anne Aaron, Ioannis Katsavounidis, Anush Moorthy, and Megha Manohara. Toward A Practical Perceptual Video Quality Metric. Netflix Technology Blog, https://netflixtechblog.com/toward-a-practical-perceptual-video-quality-metric-653f208b9652, 2016.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling. In *ICLR*, 2023.

Yijun Liu, Jie Huang, Zeyue Xue, Yuming Li, Ruizhe He, Haoran Li, Shijia Ge, and Siming Fu. HPSv3++: Scaling Reward Models Across the Full Spectrum of Diffusion Model Capabilities. arXiv preprint arXiv:2606.14657, 2026.

Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, Jianbin Jiao, and Yunfan Liu. VMamba: Visual State Space Model. In *NeurIPS*, 2024.

Xiangyang Luo, Qingyu Li, Yuming Li, Guanbo Huang, Yongjie Zhu, Wenyu Qin, Meng Wang, Pengfei Wan, and Shao-Lun Huang. Beyond the Golden Data: Resolving the Motion-Vision Quality Dilemma via Timestep Selective Training. In *CVPR*, 2026.

Shuailei Ma, Jiaqi Liao, Xinyang Wang, Jingjing Wang, Chaoran Feng, Zijing Hu, Chong Bao, Zichen Xi, Yuqi Gan, Weisen Wang, et al. Scaling mixture-of-experts video pretraining for embodied intelligence. *arXiv preprint arXiv:2607.07675*, 2026.

NVIDIA. Cosmos 3: Omnimodal World Models for Physical AI. *arXiv preprint arXiv:2606.02800*, 2026.

William Peebles and Saining Xie. Scalable Diffusion Models with Transformers. In *ICCV*, 2023.

Zihan Qiu, Zekun Wang, Bo Zheng, Zeyu Huang, Kaiyue Wen, Songlin Yang, Rui Men, Le Yu, Fei Huang, Suozhi Huang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Gated Attention for Large Language Models: Non-linearity, Sparsity, and Attention-Sink-Free. arXiv preprint arXiv:2505.06708, 2025.

Qwen Team. Qwen3-Next: Towards Ultimate Training & Inference Efficiency. Qwen Team blog, `https://qwen.ai/blog?id=qwen3-next`, 2025.

Sand.ai, Hansi Teng, Hongyu Jia, Lei Sun, Lingzhi Li, et al. MAGI-1: Autoregressive Video Generation at Scale. arXiv preprint arXiv:2505.13211, 2025.

Team Seedance et al. Seedance 2.0: Advancing Video Generation for World Complexity. arXiv preprint arXiv:2604.14148, 2026.

Tomáš Souček and Jakub Lokoč. TransNet V2: An Effective Deep Network Architecture for Fast Shot Transition Detection. arXiv preprint arXiv:2008.04838, 2020.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced Transformer with Rotary Position Embedding. *Neurocomputing*, 568:127063, 2024.

Yao Teng, Yue Wu, Han Shi, Xuefei Ning, Guohao Dai, Yu Wang, Zhenguo Li, and Xihui Liu. DiM: Diffusion Mamba for Efficient High-Resolution Image Synthesis. arXiv preprint arXiv:2405.14224, 2024.

Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam, Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion Model Alignment Using Direct Preference Optimization. In *CVPR*, 2024.

Wan Team. Wan2.2: Open and Advanced Large-Scale Video Generative Models. Official code and model release, `https://github.com/Wan-Video/Wan2.2`, 2025.

Wan Team, Ang Wang, Baole Ai, Bin Wen, et al. Wan: Open and Advanced Large-Scale Video Generative Models. arXiv preprint arXiv:2503.20314, 2025.

Haoning Wu, Erli Zhang, Liang Liao, Chaofeng Chen, Jingwen Hou, Annan Wang, Wenxiu Sun, Qiong Yan, and Weisi Lin. Exploring Video Quality Assessment on User Generated Contents from Aesthetic and Technical Perspectives. In *ICCV*, 2023.

Haocheng Xi et al. Sparse VideoGen: Accelerating Video Diffusion Transformers with Spatial-Temporal Sparsity. In *ICML*, 2025. arXiv:2502.01776.

Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, Fisher Yu, Dacheng Tao, and Andreas Geiger. Unifying Flow, Stereo and Depth Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(11):13941–13958, 2023a.

Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. ImageReward: Learning and Evaluating Human Preferences for Text-to-Image Generation. In *NeurIPS*, 2023b.

Shuai Yang, Wei Huang, Ruihang Chu, Yicheng Xiao, Yuyang Zhao, Xianbang Wang, Muyang Li, Enze Xie, Yingcong Chen, Yao Lu, Song Han, and Yukang Chen. LongLive: Real-Time Interactive Long Video Generation. In *ICLR*, 2026.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated Linear Attention Transformers with Hardware-Efficient Training. In *ICML*, 2024.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, et al. CogVideoX: Text-to-Video Diffusion Models with An Expert Transformer. In *ICLR*, 2025.

Zhiyuan You, Xin Cai, Jinjin Gu, Tianfan Xue, and Chao Dong. Teaching Large Language Models to Regress Accurate Image Quality Scores Using Score Distribution. In *CVPR*, 2025.

Jingyang Yuan et al. Native Sparse Attention: Hardware-Aligned and Natively Trainable Sparse Attention. arXiv preprint arXiv:2502.11089, 2025.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid Loss for Language Image Pre-Training. In *ICCV*, 2023.

Jintao Zhang et al. SpargeAttention: Accurate and Training-free Sparse Attention Accelerating Any Model Inference. In *ICML*, 2025. arXiv:2502.18137.

Yuyang Zhao, Yicheng Pan, Qiyuan He, Jincheng Yu, Junsong Chen, Tian Ye, Haozhe Liu, Enze Xie, and Song Han. SANA-Streaming: Real-time Streaming Video Editing with Hybrid Diffusion Transformer. arXiv preprint arXiv:2605.30409, 2026.

Zangwei Zheng, Xiangyu Peng, Yuxuan Lou, et al. Open-Sora 2.0: Training a Commercial-Level Video Generation Model in $200k. arXiv preprint arXiv:2503.09642, 2025.

Haoyi Zhu, Haozhe Liu, Yuyang Zhao, Tian Ye, Junsong Chen, Jincheng Yu, Tong He, Song Han, and Enze Xie. SANA-WM: Efficient Minute-Scale World Modeling with Hybrid Linear Diffusion Transformer. arXiv preprint arXiv:2605.15178, 2026.

Lianghui Zhu, Zilong Huang, Bencheng Liao, Jun Hao Liew, Hanshu Yan, Jiashi Feng, and Xinggang Wang. DiG: Scalable and Efficient Diffusion Models with Gated Linear Attention. In *CVPR*, 2025.

A Related Work

A.1 Video Diffusion and Efficient Sequence Modeling

Modern open video generators are largely built on diffusion transformers with full uniform softmax attention. Wan 2.1/2.2, HunyuanVideo, CogVideoX, and Open-Sora scale this design through larger backbones and improved training recipes, with Wan 2.2 adopting a mixture-of-experts denoiser (Wan Team et al., 2025; Wan Team, 2025; Kong et al., 2024; Yang et al., 2025; Zheng et al., 2025). Their global token mixing is expressive, but its quadratic cost becomes increasingly expensive as resolution and duration grow. Other systems reduce the sequence length instead: MAGI-1 generates video autoregressively in temporal chunks (Sand.ai et al., 2025), while LTX-Video relies on latent compression (HaCohen et al., 2024). LongLive combines frame-level causal autoregression with cache reuse for real-time interactive long video, and LongLive-2.0 adds sequence-parallel and low-precision training and inference infrastructure (Yang et al., 2026; Chen et al., 2026).

Efficient sequence models reduce backbone cost directly. Linear attention replaces the explicit $N \times N$ attention map with a fixed-size state (Katharopoulos et al., 2020; Yang et al., 2024; Arora et al., 2024), while state-space models use recurrent or selective state updates (Gu & Dao, 2024; Li et al., 2024; Teng et al., 2024; Liu et al., 2024). In vision, DiG applies gated linear attention to image diffusion transformers (Zhu et al., 2025); SANA-Video develops a pure-linear video DiT with a temporal-convolution FFN (Chen et al., 2025); and SANA-WM and SANA-Streaming extend efficient SANA-family backbones to world modeling and streaming video editing (Zhu et al., 2026; Zhao et al., 2026). Because a compressed state can restrict representation rank and direct token interaction (Fan et al., 2025), we keep full-sequence generation but periodically restore exact softmax interaction instead of relying on a pure-linear stack or an additional temporal-convolution path.

A.2 Hybrid and Sparse Attention

Hybrid language models combine efficient state-based layers with a smaller number of exact softmax layers. Qwen3-Next and Kimi-Linear use a regular 3:1 linear-to-softmax layout with different linear operators (Qwen Team, 2025; Kimi Team, 2025), and post-attention output gating provides a compatible stabilization mechanism for the softmax branch (Qiu et al., 2025). Attention Surgery instead linearizes a pretrained video DiT after training (Ghafoorian et al., 2025). We study a bidirectional video diffusion transformer trained from scratch with its hybrid layout fixed throughout training, and re-select the softmax fraction in the video regime rather than assuming that the language-model ratio transfers directly.

Sparse methods provide a complementary way to reduce the remaining softmax cost. SparseVideoGen and Radial Attention exploit structure in video attention maps, SpargeAttn and PISA provide training-free sparse or approximate kernels, and Native Sparse Attention learns hardware-aligned sparse patterns for long-context language modeling (Xi et al., 2025; Li et al., 2025; Zhang et al., 2025; Li et al., 2026a; Yuan et al., 2025). These methods reduce the work within each softmax layer, whereas our hybrid design reduces how many layers invoke softmax. In our deployment study, sparsity is therefore applied only to the fixed softmax anchors, leaving the linear majority and trained architecture unchanged (Section 6).

A.3 Cross-Layer Residual Routing

Standard transformer residual streams pass features only through adjacent layers. Attention Residuals allow a layer to retrieve completed block features from earlier depths, and Kimi K3 combines this mechanism with hybrid attention at language-model scale (Kimi Team, 2026b;a). Video diffusion introduces different requirements: features are bidirectional, every block is modulated by a denoising timestep, and the spatial–temporal token set is much larger. We therefore route over completed block summaries and the current partial block, share one query projection across depth, and retain timestep information through the ordinary DiT features rather than a separate router-timestep projection. The routing and rank analyses in this paper test whether the AttnRes path reuses the representations refreshed by the softmax anchors.

B Model and Training Details

B.1 Model Configuration

Table 3 records the architecture and training hyperparameters of the 5B SANA-Video 2.0 model.

Table 3: **Full 5B configuration of SANA-Video 2.0.** Architecture and production training settings.

Setting	5B
Backbone architecture	
Depth	32
Hidden dimension	2,560
Linear attention	20 heads (d_h=128)
Softmax attention	10 heads (d_h=256)
Attention operators	Gated bidirectional linear / gated softmax; self- and cross-attention QK normalization
Softmax anchors	8 layers; 25% (3:1)
AttnRes	S=8, shared projection, no time conditioning
FFN	SwiGLU, ratio 4.0
Position encoding	Wan RoPE; separate linear/softmax head dimensions
Patch size	$(1, 1, 1)$
Parameters	$\sim$4.5B (5B class)
Training configuration	
Resolution / frames	480p–720p; 81/121/193
VAE	LTX-VAE 2.3, 128 channels, causal encode
VAE stride	8×32×32 (temporal × spatial × spatial)
Text encoder	Gemma-2-2B-IT, 300 tokens; normalized features, scale 0.01
Objective	Flow matching; stage-specific TQD / Self-Flow
Self-Flow	student/teacher 9/25, weight 0.8, R_M=0.1
Noise distribution	shift 1 then token-aware 3–6, σ=0.95
Optimizer	AdamW, lr 10^{-4} / 5×10^{-5}
Schedule / clipping	constant + 500-step warmup; clip 0.1
Per-rank batch	1 video
Precision / sharding	bf16, FSDP
Weight EMA	enabled
Training hardware	64×H100

B.2 Self-Flow Distillation and Dual-Timestep Tokens

Self-Flow (Chefer et al., 2026) is an auxiliary training aid applied only during pre-training and continual training (disabled for SFT). It adds a feature-distillation loss from a shallow student readout toward a deeper (or, in later continual training, EMA) teacher, and a dual-timestep schedule that assigns a small token partition (R_M=0.1) to a second timestep. All tokens stay in the flow-matching loss, so it changes conditioning rather than reducing backbone compute; the exact layer indices and weights are in Table 3.

B.3 Content-aware Flow-Shift

TQD (Luo et al., 2026) uses curation scores to place supervision where each pre-training clip is most informative. With base flow-shift 1, clips above the fps-normalized UniMatch threshold of 30 but not the DOVER quality threshold receive a $+1.1$ logit bias toward high noise; clips above the DOVER threshold of 0.91 but not the motion threshold receive a -1.1 bias toward low noise; clips satisfying both or neither receive zero bias. The two biased branches have median timesteps of approximately 0.75 and 0.25, corresponding to effective shifts of about 3 and $1/3$, respectively (Figure 7). From continual training onward TQD is disabled and a tail-floored token-aware density with shift 3–6 is used. Shift 3 aligns approximately with the high-motion branch's median, but the distributions and the other TQD branches differ, so this is not a per-sample continuous handoff.

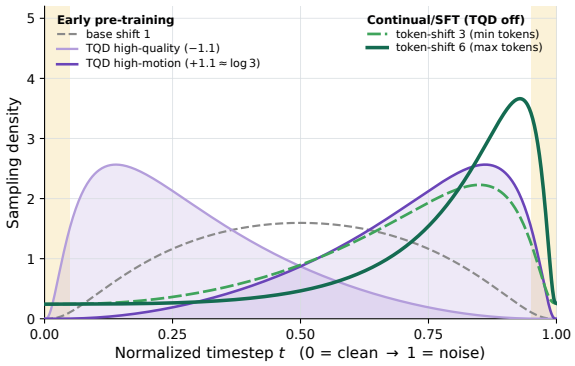
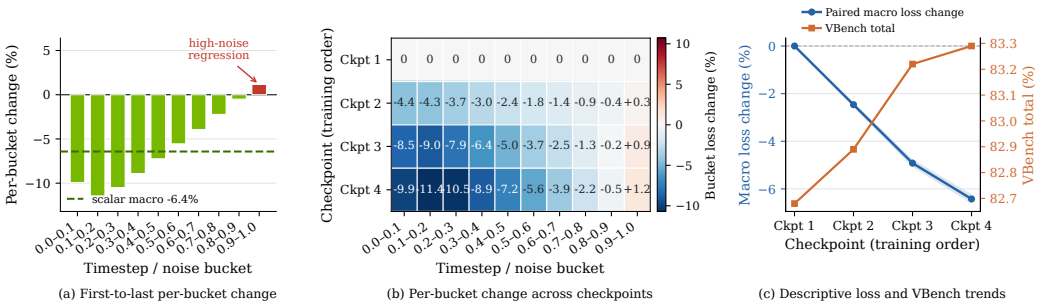


Figure 7: **Timestep densities used in successive stages.** Pre-training TQD applies opposite offsets to the mutually exclusive high-quality and high-motion branches. Continual training and SFT disable TQD and use a tail-floored token-count shift spanning 3–6; shift 3 approximately aligns with the high-motion branch's median but is not a continuous per-sample handoff.

B.4 Timestep-Stratified Checkpoint Monitoring

A random-timestep mean can hide improvements in one noise regime behind regressions in another. We therefore divide the 1,000-step noise axis into ten equal buckets and compare four checkpoints from one continuous trajectory. Every checkpoint reuses the same 100 clips, text features, noise tensors, and timestep assignments, covering each integer timestep exactly once. The equal-bucket average is a monitoring statistic, not the training objective.

From the first to last checkpoint, macro MSE falls by 6.4%, but the bucket view reveals the useful detail: low-noise MSE improves by as much as 11.4%, while the highest-noise bucket regresses by 1.2% (Figure 8). Stratification also reduces the standard deviation of the change estimate by $2.3\times$ relative to IID timestep sampling. VBench Total rises from 82.68 to 83.29 over these checkpoints; with four observations, this is context rather than a correlation claim.



(a) First-to-last per-bucket change (b) Per-bucket change across checkpoints (c) Descriptive loss and VBench trends

Figure 8: **Timestep-stratified checkpoint monitoring.** Four matched checkpoints from one run. (a,b) Bucketwise loss changes expose a large low-noise improvement and a small high-noise regression that the scalar mean blurs together. (c) Paired macro change with 95% intervals and the descriptive VBench trend.

C Data Curation and Progressive Selection

We process the corpus with a uniform six-stage pipeline. Its purpose is not only to remove defective clips, but to construct the progressive supervision used by the training curriculum: pre-training favors coverage, continual training raises the quality and motion-consistency floors, and SFT uses the most trusted subset. The same processing logic is used across these pools, with stage-dependent selection thresholds.

Table 4: **Signals used by the in-house quality/motion funnel.** Selection thresholds tighten progressively across the pre-training, continual, and SFT stages.

Signal family	Examples	Role
Visual quality	DOVER, blur/exposure statistics	Fidelity and aesthetics
Motion	optical flow, VMAF motion	Richness and consistency
Color	saturation and luminance statistics	Natural appearance
Consistency/alignment	image–video, vision–language similarity	Conditioning reliability
Shot integrity	cut and artifact detectors	Temporal continuity

Decode, segment, and clean. We first reject undecodable or undersized clips, normalize metadata, and split long videos into coherent shots with TransNetV2 (Souček & Lokoč, 2020). Temporally stable crops remove letterboxing and subtitles when possible; clips dominated by overlays, unstable cuts, severe blur, broken exposure, or temporal artifacts are discarded.

Score complementary properties. The remaining clips are evaluated along separate quality, motion, color, and consistency axes rather than collapsed into one score (Table 4). DOVER measures visual quality, UniMatch and VMAF-derived features describe motion, and SigLIP features measure visual–text consistency (Wu et al., 2023; Xu et al., 2023a; Li et al., 2016; Zhai et al., 2023). Decisions are aggregated over time, and optical flow is normalized by frame rate so informative low-motion clips are not removed simply for being slow.

Build the curriculum. Progressively stricter per-axis thresholds form a broad pre-training pool, a cleaner continual-training pool, and a small trusted SFT subset. Captions follow the same progression, ending in a structured description of subjects, actions, camera motion, scene, lighting, interactions, and temporal evolution. Quantized motion descriptors are appended when available so motion remains explicit to the text conditioner.

Resolution, FPS, and duration schedule. The run advances resolution ($480p \rightarrow 720p$), frame rate, and duration together, with the 720p share increasing in later stages (Table 7). Frame counts are chosen so each clip lands on an integer number of LTX-VAE latent frames. Because FSDP runs at per-rank batch size one, the sampler buckets each rank into a single (aspect-ratio, frame-count) tier per step, so resolution/duration mixing does not desynchronize ranks. Image batches are interleaved with video at a fixed cadence to inject appearance quality into a motion-biased corpus.

D Evaluation and Measurement Protocols

D.1 Sampling and VBench Evaluation

The reported SANA-Video 2.0 operating points use 40 flow-DPM-Solver steps, motion bucket 30, seed 0, and Gemma-2-2B-IT text features (Gemma Team, 2024). The 81-frame, 16-fps setting uses classifier-free guidance 6.0 and flow-shift 6.0; the 121/193-frame, 24-fps settings use guidance 8.0 and flow-shift 12.0. The 81-frame result comes from one late-stage production checkpoint, while the 121/193-frame results share a second late-stage checkpoint. Baselines keep their default settings.

We evaluate the full VBench (Huang et al., 2024) text-to-video suite with official prompts over all 16 dimensions. Quality averages seven and Semantic nine; Total uses the default 4:1 quality–semantic weighting. Table 5 abbreviates the dimensions as SC/BC (subject/background consistency), TF (temporal flickering), MS (motion smoothness), DD (dynamic degree), AQ/IQ (aesthetic/imaging quality), OC (object class), MO (multiple objects), HA (human action), CO (color), SR (spatial relationship), SE (scene), AS/TS (appearance/temporal style), and OV (overall consistency).

Score sources. All baseline scores are either official results ($\star$: official 720p leaderboard, ‡: author-reported) or measured by us with each model's official pipeline and default inference settings (†). Unmarked scores use the common $480\times832\times81$ protocol. Table 1 shows the compact selection with the 81-frame SANA-Video 2.0 result, and Table 5 adds the full baseline set and our 121-/193-frame operating points. Wan 2.2 is A14B active (27B total) (Wan Team, 2025).

Table 5: **Complete VBench comparison across all 16 dimensions** (%). Green intensity ranks the top three results per column (best in bold); dashes mark models with only aggregate scores reported. Appendix D defines the score-source markers and protocols.

Model	Aggregates			Quality dimensions							Semantic dimensions								
	Total↑	Quality↑	Semantic↑	SC	BC	TF	MS	DD	AQ	IQ	OC	MO	HA	CO	SR	SE	AS	TS	OV
LTX-2.3 (base)†	81.95	83.85	74.35	93.10	95.77	98.92	98.71	71.39	62.13	67.51	88.89	60.26	95.80	81.31	68.59	51.38	21.11	24.31	25.91
CogVideoX 1.5	82.17	82.78	79.76	96.87	97.35	98.88	98.31	50.93	62.79	65.02	87.47	69.65	97.20	87.55	**80.25**	52.91	**24.89**	25.19	27.30
Cosmos-3 Nano†	83.13	84.29	78.50	95.05	97.26	99.08	**99.06**	60.28	62.71	69.37	89.29	78.03	96.60	89.20	72.72	54.56	20.98	24.49	26.78
Lance (Fu et al., 2026)†	83.18	84.09	79.55	94.68	94.95	98.34	98.92	71.11	63.71	67.71	94.18	76.69	**97.60**	81.33	79.56	52.83	23.12	24.58	26.89
Wan 2.1 (1.3B)	83.31	85.23	75.65	97.56	97.93	**99.55**	98.52	65.19	65.46	67.01	88.81	74.83	94.00	89.20	73.04	41.96	21.81	23.13	25.50
HunyuanVideo*	83.43	85.07	76.88	97.22	97.60	99.39	99.05	71.94	60.28	67.24	83.48	66.71	94.40	**89.79**	72.13	54.46	22.21	24.52	26.95
Wan 2.1 (14B)	83.69	85.59	76.11	97.52	**98.09**	99.46	98.30	65.46	66.07	69.43	86.28	69.58	95.40	88.59	75.39	45.75	22.64	23.19	25.91
SANA-Video	84.17	84.85	81.46	97.13	97.71	98.37	98.20	69.17	68.27	64.82	**95.39**	**85.05**	95.40	89.43	78.00	57.83	22.43	24.31	27.04
Wan 2.2 (A14B)*	84.23	85.42	79.50	97.29	97.39	99.22	98.16	61.02	67.22	**71.75**	94.06	82.10	96.40	87.43	78.39	56.80	20.39	23.64	26.12
Open-Sora 2.0	84.34	85.40	80.12	97.71	98.00	99.40	98.69	71.39	64.39	65.66	94.50	77.72	95.40	85.98	76.18	52.71	22.98	**25.91**	**27.50**
Bernini-R‡	84.64	85.18	**82.49**	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
SANA-Video 2.0 (81f)	84.30	85.61	79.05	97.34	97.33	97.84	98.31	78.89	67.99	66.51	92.80	76.75	94.40	88.05	71.85	**59.88**	21.35	24.05	26.93
SANA-Video 2.0 (121f)	**85.29**	**86.98**	78.51	97.44	97.09	97.58	98.40	**92.50**	**68.85**	68.39	92.39	74.38	95.00	89.17	69.34	58.81	21.26	24.46	26.64
SANA-Video 2.0 (193f)	84.48	86.51	76.37	**98.00**	96.94	97.94	98.52	88.33	68.10	66.31	91.76	72.39	92.60	88.21	65.32	54.68	21.27	23.48	26.08

Latency protocol. End-to-end latency covers text encoding, denoising, and VAE decoding at batch size one, excluding checkpoint loading and video writing. Unless noted otherwise, we report the mean of three steady-state videos after one full warmup on a single H100. Baselines use their official pipelines and native 720p-class shapes; Wan 2.2-A14B, Bernini-R, and the 5B SANA-Video 2.0 model use 40 steps in the headline comparison.

At $736\times1280\times81$, the 5B pipeline takes 30.9s; at $480\times832\times81$, it takes 13.2s compared with 421s for Bernini-R. For context, Chen et al. (2025) reports 36s for the 720p SANA-Video-2B pipeline.

D.2 Latency and Profiling Protocols

The standard architecture profiles use one H100 80GB, bf16, batch size one, eight warmups, and 20 CUDA-event timed forwards (median). Every attention ratio receives its best implementation: fused linear attention, FlashAttention for softmax, and `torch.compile`. Figures 5(a,b) vary resolution and duration with AttnRes disabled; panel (c) follows the same compiled protocol while scaling matched backbones at a fixed 19.3K-token shape. These are DiT-forward measurements and exclude text encoding and VAE decoding.

The module-composition figure is a separate eager diagnostic with the selected shared router enabled. It attributes non-overlapping CUDA-event intervals to top-level modules at 22.1K and 111.3K tokens; it is used only to identify bottlenecks, not as a second speedup protocol or as evidence about learned routing. Results from different protocols are never multiplied unless a table explicitly defines the composition.

E Qualitative Results

E.1 Additional Qualitative Samples

Figure 13 shows six additional 720p text-to-video clips (1280×736, 193 frames, 8s at 24fps) spanning people, wildlife, urban scenes, and stylized content. Each row unrolls one clip as five frames at $0/2/4/6/8$s, generated with the same 40-step 193-frame sampling recipe as above; the generating prompt is reproduced beneath each row. Four further showcase prompts from the same batches (a violinist in light rain, an octopus changing color, a drone light show, and a desert lightning storm) are held out for the cross-method comparison below (Figure 15).

E.2 Qualitative Effect of Online RL

Figure 14 complements the training-time reward trajectories in Figure 19 with four matched-prompt, matched-CFG comparisons between the SFT and RL checkpoints. Five evenly spaced frames expose changes throughout each eight-second clip; these prompt-level examples provide qualitative context and are not an aggregate evaluation.

E.3 Qualitative Comparison with Other Methods

Figure 15 compares SANA-Video 2.0 with Wan 2.2-A14B, Bernini-R 14B, Cosmos-3 Nano, and LTX-2.3 on four prompts chosen to expose temporal behavior: sustained action, color change, shape

transformation, and repeated events. Each method uses its official pipeline, native 720p-class shape, and step count on one H100 (seed 42); SANA-Video 2.0 uses 40 steps and 193 frames. Because durations differ, we show each output at normalized start, midpoint, and end rather than at absolute timestamps. These are qualitative examples, not an aggregate ranking.

Figures 16 and 17 extend the comparison to image-conditioned generation. Every method receives the same first frame and caption from VBench-I2V, and each row shows normalized progress at $0/25/50/100\%$. SANA-Video 2.0 generates 121 frames with its image-conditioned checkpoint; LTX-2.3, Wan 2.2 TI2V-5B, and Cosmos-3 use their official first-frame conditioning paths. Bernini-R is omitted because its public pipeline has no corresponding mode. All methods run at native 720p-class shapes with seed 42 on one H100. The examples illustrate differences in motion and composition retention, but do not substitute for a quantitative TI2V benchmark.

F AttnRes Routing and Representation Analysis

This section explains what the selected router learns and why its final form shares one query across depth without an explicit timestep input. Unless noted otherwise, the analyses use 5B architecture-family checkpoints; the continued-training comparison below uses the mature 4.5B production ancestor.

Among completed blocks, the nearest receives 26.0%/25.3% of attention/FFN routing mass, the next 15.5%/13.2%, and the third 14.5%/11.1%. This recency gradient holds across depth and complements the aggregate reuse in Figure 6c.

F.1 Late-Stage Continued-Training Probe

To test the router after useful video representations have formed, we continue a block-level AttnRes run from the same mature 4.5B hybrid ancestor as the no-router trajectory. Near-matched checkpoints are evaluated on the same 100 held-out clips and 20 noise levels. Their mean MSE is effectively tied (0.4851 with AttnRes versus 0.4855 without it), although 17 of 20 buckets favor AttnRes. The buckets share the same clips, so we read this as a direction check rather than an independent test. We use this probe only to rule out degradation; the evidence for cross-depth reuse comes from the routing and representation interventions below, not from this narrow loss difference.

F.2 Routing Selectivity Metric

For each layer l, AttnRes computes aggregation weights $\alpha_l = \mathrm{softmax}(q_l \cdot K)$ over N_l depth sources (initial embedding b_0, completed block summaries $b_1, b_2, \ldots$, and the current partial sum p_l). We measure selectivity via normalized Shannon entropy:

$$\hat{H}_l = \frac{H(\alpha_l)}{\log N_l} = \frac{-\sum_{j=1}^{N_l} \alpha_{l,j} \log \alpha_{l,j}}{\log N_l} \tag{3}$$

For layer l, the attention branch has $N_l^{\mathrm{attn}} = \lceil l/S \rceil + \mathbf{1}[(l-1) \bmod S > 0]$ sources, while the FFN branch has $N_l^{\mathrm{ffn}} = \lceil l/S \rceil + 1$. We omit the first attention routing site ($N = 1$) from normalized-entropy summaries. The normalization otherwise makes source sets of different sizes comparable: $\hat{H}_l = 1$ is uniform and $\hat{H}_l \to 0$ is selective.

F.3 Timestep Adaptivity

We evaluate a late 5B checkpoint trained natively without the explicit router timestep offset. The probe uses 20 captioned clips from the anime data source and five noise fractions $t \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. Each clip's latent, text features, and Gaussian noise tensor are cached once and reused across all five timesteps, so changes in clip identity or sampled noise cannot explain the measured variation. This is a within-checkpoint representation diagnostic, not a held-out quality benchmark or a matched comparison of query-sharing designs.

For each clip c and routing site s, we define timestep adaptivity as

$$\mathrm{Adaptivity}_{c,s} = \sigma_t[\hat{H}_{c,s}(t)]. \tag{4}$$

We then average over attention and FFN routing sites within each clip, excluding the single-source first attention site and the final output aggregator. The mean over clips is 0.0164, with an across-clip sample standard deviation of 0.0011. Figure 6a shows per-layer attention and FFN means with across-clip standard deviations, in percentage points. The result demonstrates timestep-dependent routing in a model trained without query offsets; it does not establish a quality gain or superiority to per-layer queries. DiT timestep modulation remains enabled throughout.

F.4 Timestep Conditioning Probe

To determine whether the explicit router timestep projection learns timestep-specific information or merely a constant bias, we combine a behavioural probe with a direct weight-space analysis of its learned offset $\phi_\tau(t)$.

Behavioural probe. Let q_0 be the base routing query. We evaluate the ablation checkpoint under three conditions: **Normal**, $q = q_0 + \phi_\tau(t)$; **Mean**, $q = q_0 + \bar{\phi}_\tau$, where $\bar{\phi}_\tau$ averages 21 uniformly spaced timesteps; and **Shuffle**, $q = q_0 + \phi_\tau(1 - t)$ for AttnRes while AdaLN still receives the correct t. Their respective aggregate validation-loss point estimates are 0.4905, 0.4906, and 0.4907, a maximum difference of 0.0002 in this probe.

Weight-space probe (Figure 9). Across 21 timesteps, the learned offset remains nearly constant (mean cosine 0.979), and its varying component is only 13% of the constant component's norm. Together with the negligible behavioural change above, this motivates removing the explicit router-timestep projection while retaining timestep modulation inside every DiT block.

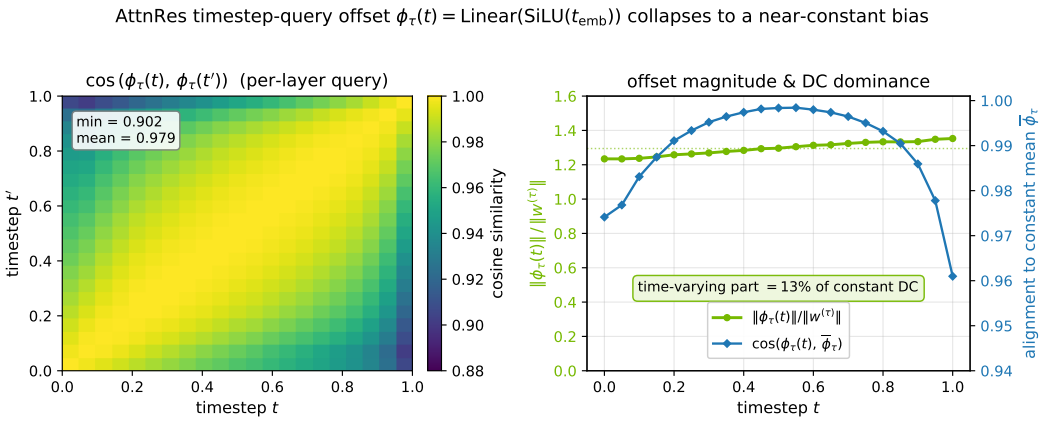


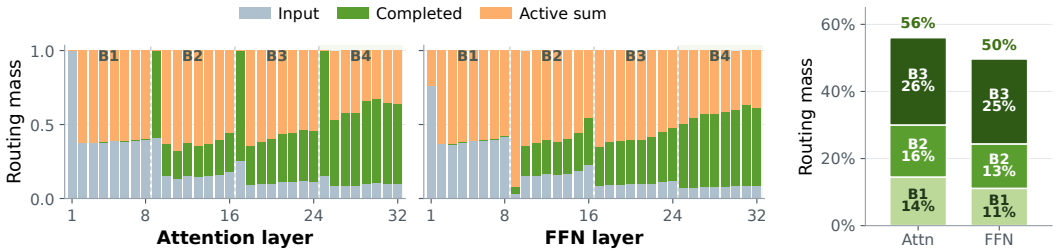
Figure 9: **The explicit router offset is nearly timestep-invariant.** Across 21 timesteps, $\phi_\tau(t)$ remains strongly aligned with its mean and its time-varying component is small.

F.5 Depth-Routing Patterns

Figures 6c and 10 average routing over ten validation clips. Completed summaries receive roughly half of the routing mass in layers 25–32 (56% for attention and 50% for FFN), and the detailed view favors the most recently completed block. A 24-prompt replication with real text conditioning on the production checkpoint reproduces both effects, with 52.9% of attention and 46.9% of FFN mass on completed summaries and the nearest completed block again taking the largest share. The difference at block entries follows operation order: attention routing occurs before the new partial sum exists, whereas FFN routing occurs afterward. Because each summary contains attention, cross-attention, and FFN updates, this evidence supports cross-depth feature reuse without assigning it to a single operator.

F.6 Completed-Block Routing Ablation

We zero the routing weights assigned to completed block sums, renormalize the remaining sources, and measure the effective rank of h_l across 24 prompts at $t \in \{0.3, 0.5, 0.7\}$ (Figure 11a). At block-



(a) Layerwise routing; B1–B4 mark the 8-layer blocks.

(b) Completed-source detail in layers 25–32.

Figure 10: **AttnRes routing-source composition** (ten-clip mean). In (a), dashed lines mark block resets; the first attention layer carries only the input source.

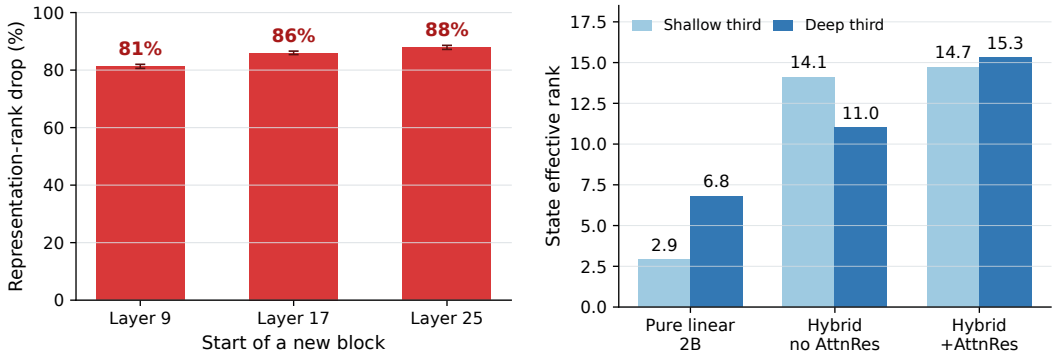
entry layers $(9, 17, 25)$, the current partial sum has reset, so removal leaves the input embedding and reduces rank by 81–88%, with an across-prompt standard deviation below 1% at every entry. Once the partial sum has rebuilt within a block, the removal no longer reduces rank: at mid-block layers it changes rank by $-14.6\% \pm 11.6\%$, a slight increase rather than a drop. This localizes completed-block reuse to the boundary where it is needed most, without attributing the recovered rank to one operator inside the block.

F.7 Same-Checkpoint Rank Probe

For the gated, RoPE-rotated linear state $S = \sum_n v_n(\beta_n k_n^r)^\top$, we define

$$r_{\text{eff}}(S) = \exp\left(-\sum_i p_i \log p_i\right), \qquad p_i = \frac{\sigma_i(S)}{\sum_j \sigma_j(S)}. \tag{5}$$

In implementation, we first average corresponding singular values across attention heads, then normalize the averaged spectrum and compute r_{eff}. We toggle depth aggregation on and off on the same production checkpoint across 24 prompts, pairing noise seeds 0–3 between the two conditions for each prompt. Figure 6b reports each linear layer's mean and across-prompt standard deviation, and the deep-layer summary (layers ≥ 15) rises by $19.7\% \pm 5.1\%$ (mean and standard deviation across prompts). No weights or activations other than the AttnRes path are changed, isolating a representation-level effect of the trained router. For context (Figure 11b), we place this result beside a public 2B pure-linear model and a separately trained no-AttnRes model. These differ in width, training horizon, and checkpoint, so the comparison is contextual rather than an AttnRes effect estimate.



(a) Earlier-block feature removal.

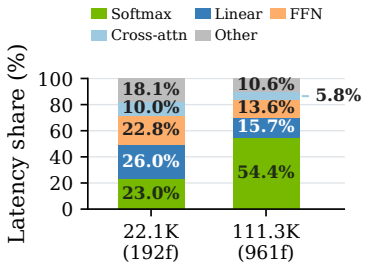
(b) Step-unmatched rank context.

Figure 11: **Routing intervention and rank context.** (a) Representation-rank drop after removing completed-block sources at block entries. (b) Descriptive comparison across separately trained models that differ in width, training horizon, and checkpoint.

Table 6: **Controlled FFN latency swap** at 720p (one H100; ms). T-conv denotes temporal convolution; Δ is its overhead; † marks the excluded 60s pair.

Backbone	FFN	5s	15s	20s	30s	45s	60s
L=20, d=2240	SwiGLU	52	126	161	242	367	510
	T-conv	64	157	202	302	455	627
	Δ	+12	+31	+42	+60	+88	+117
L=32, d=2560	SwiGLU	306	906	1258	2036	3416	4994†
	T-conv	396	1141	1573	2499	4091	10043†
	Δ	+90	+236	+315	+463	+677	—†

Figure 12: **Eager module composition** at 22.1K/111.3K tokens under the controlled forward-profile protocol.



G Efficiency and Deployment Analysis

These controlled analyses explain the scaling in the main efficiency figure and complement it with deployment measurements. Appendix D.2 specifies each protocol. Results are combined across studies only when a table reports a direct combined measurement.

G.1 Model-Size Scaling at Fixed Sequence Length

Figure 5(c) fixes a 720p/10s sequence while jointly increasing width and depth. Absolute time saved by the 25% hybrid grows from 128 to 948ms, while Full/Hybrid speedup narrows from 1.88× to 1.45×. A controlled decomposition explains the trend: depth scales both layouts nearly equally, whereas width shifts more compute into shared FFN/projections. These matched forward profiles isolate systems scaling.

G.2 Temporal-Convolution FFN Ablation

SANA-Video uses an additional temporal-convolution path inside its FFN, whereas SANA-Video 2.0 relies on attention for spatiotemporal mixing and keeps a standard SwiGLU FFN. To measure the cost of that choice, we hold depth and width fixed, change only the FFN path, and sweep 720p duration at two model scales (Table 6). The variants are architecture-matched rather than parameter-matched because the temporal convolution adds weights. Its overhead grows with duration and reaches 20–29% at the larger scale. The large-scale 60s temporal-conv timing is an isolated outlier and is excluded together with its paired baseline. Removing the temporal path simplifies the backbone and preserves the scaling benefit of the selected attention ratio.

G.3 Module-Level Composition

Figure 12 reports mutually exclusive top-level module shares from an eager shape diagnostic (modules are non-overlapping, so the shares are additive). From 22.1K to 111.3K tokens, the eight softmax anchors grow from 23.0% to 54.4% of forward time, while linear self-attention falls from 26.0% to 15.7%, which motivates anchor-specific kernels for long shapes.

G.4 Compile-Friendly AttnRes

The original inference path mutates a source buffer and reduces a growing prefix, which is difficult for `torch.compile` to fuse. We instead expose the at-most-five sources as a static list and accumulate their weighted sum in fp32 without stacking all sources. Relative to the buffer path, full-forward drift is negligible in fp32 and 0.68% in bf16. The rewrite improves compiled/eager speedup from 1.47× to 2.11× at 8s and from 1.21× to 1.49× at 60s. These numbers compare router implementations, not complete model deployments.

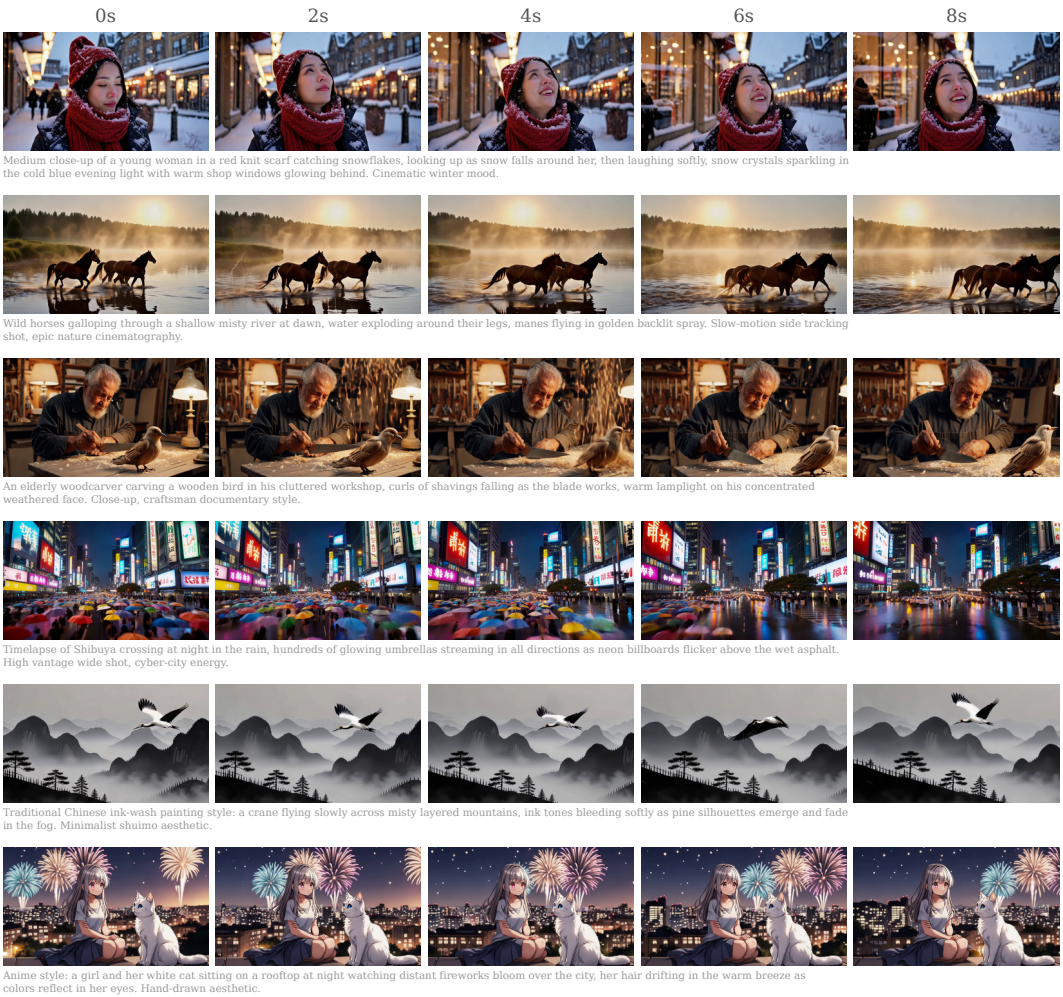


Figure 13: **Additional 720p text-to-video samples.** Each row unrolls one 8s 1280×736 clip as five frames at $0/2/4/6/8$s; the generating prompt is shown beneath each row.

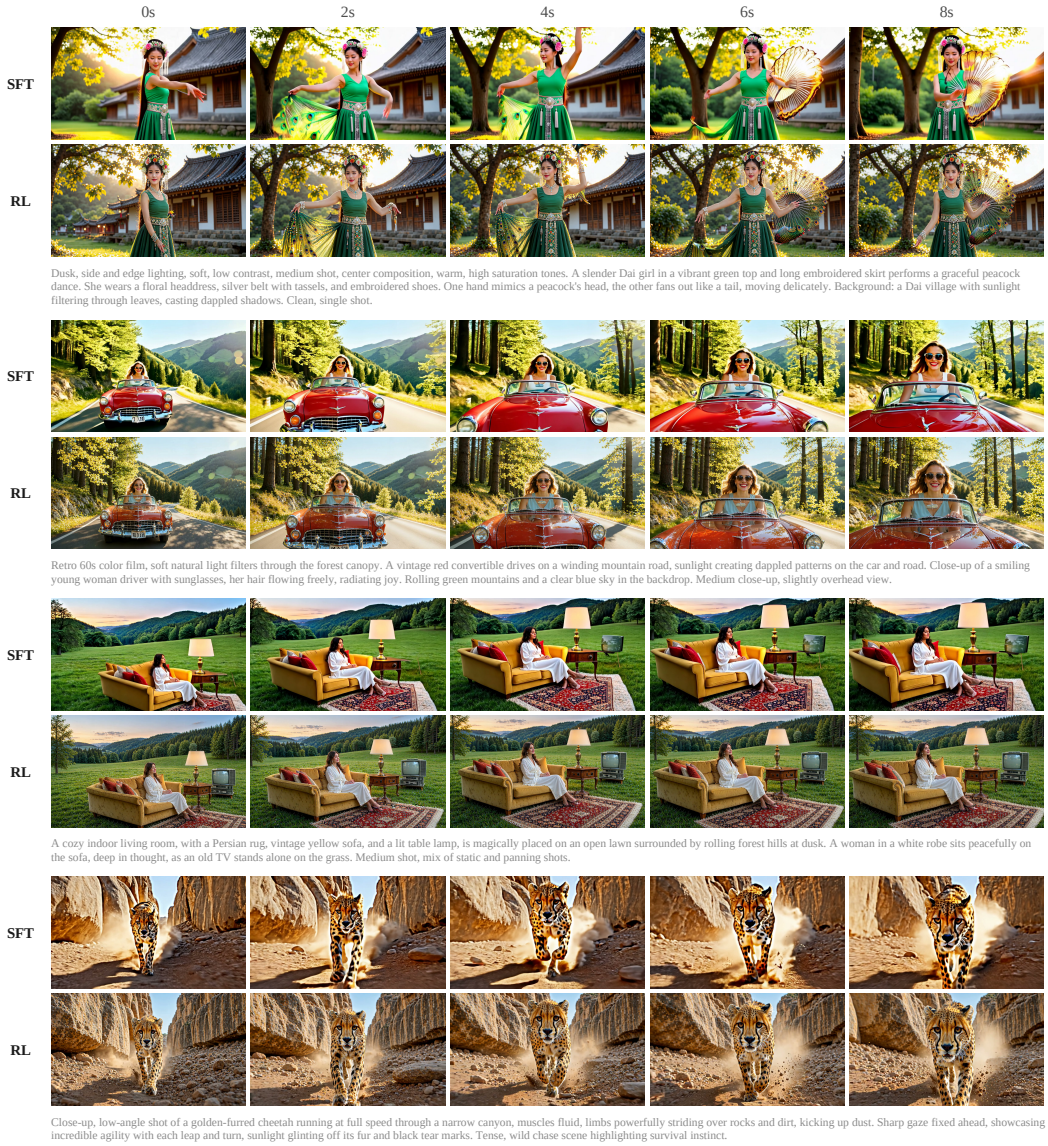


Figure 14: **Qualitative effect of online RL across four prompts.** Each block compares generations from the SFT and RL checkpoints using the same prompt and classifier-free guidance scale (CFG 8). Columns show frames at $0/2/4/6/8$s, and the complete generating prompt is reproduced beneath each pair.

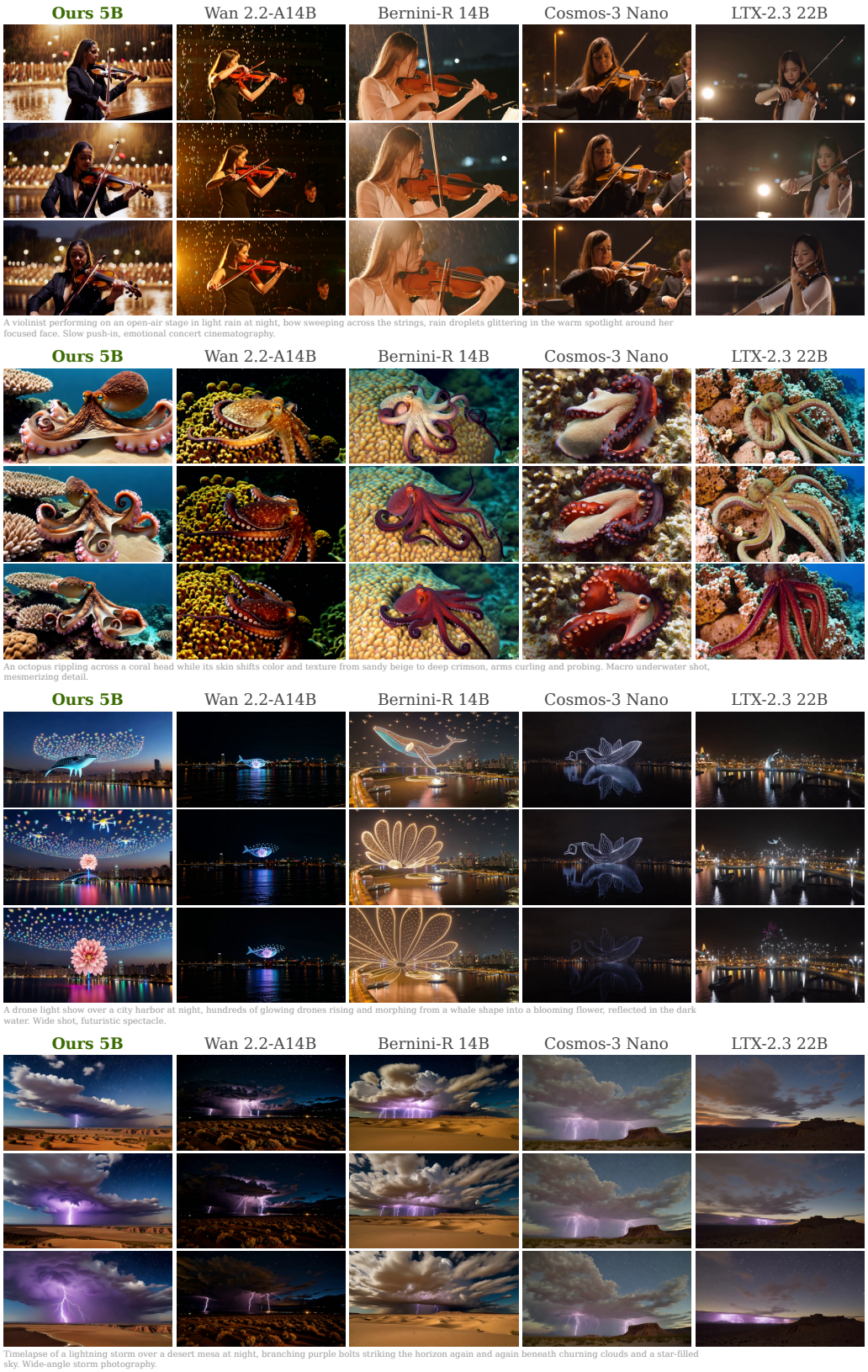


Figure 15: **Cross-method 720p text-to-video comparison** on four held-out prompts. Columns are methods and rows show $0/50/100\%$ of each clip; prompts appear below the corresponding blocks.

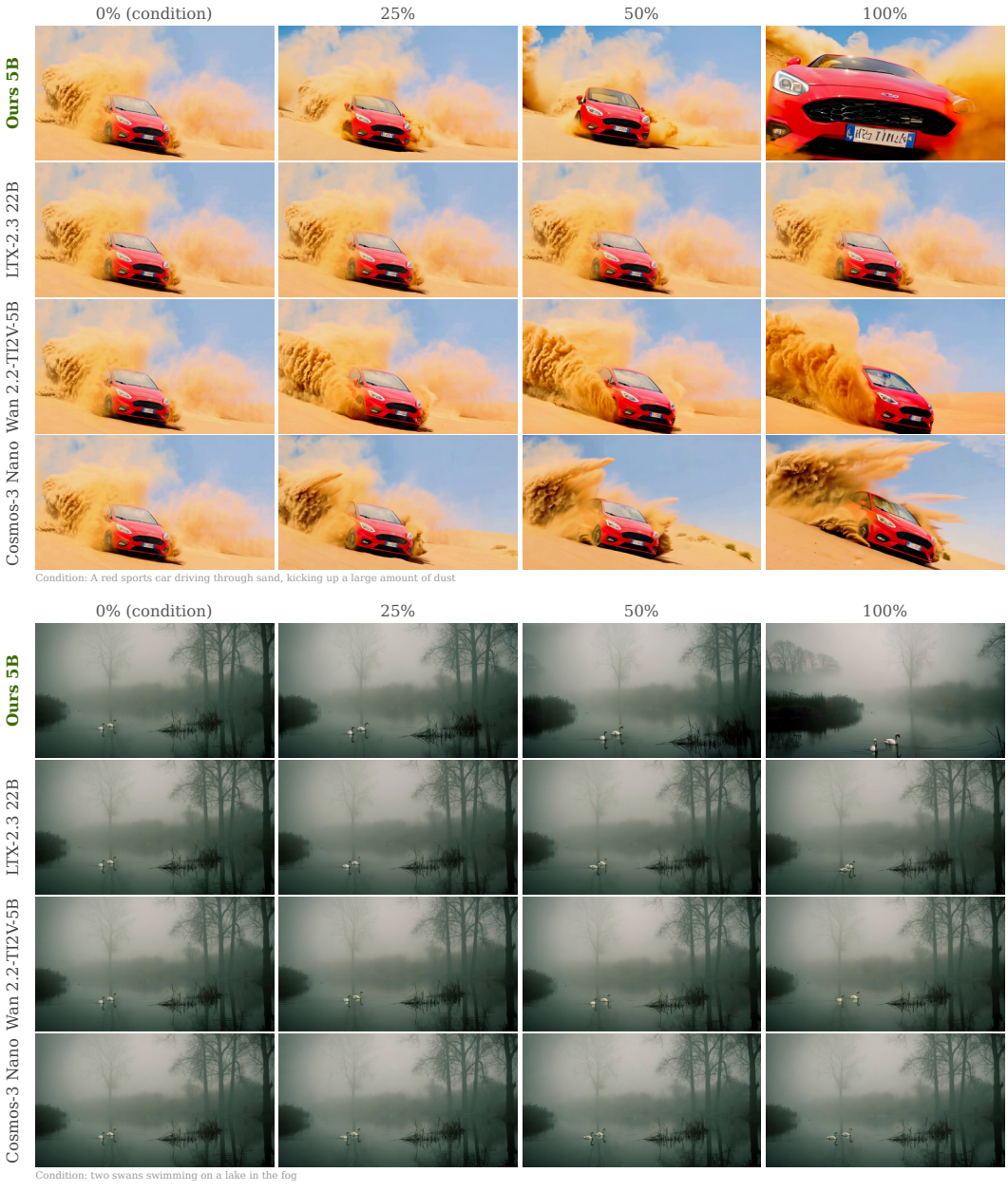


Figure 16: **First-frame-conditioned video comparison, pairs 1–2.** Each row shows one method at $0/25/50/100\%$ of its clip; the first frame is shared. Protocol details are in Appendix E.3.

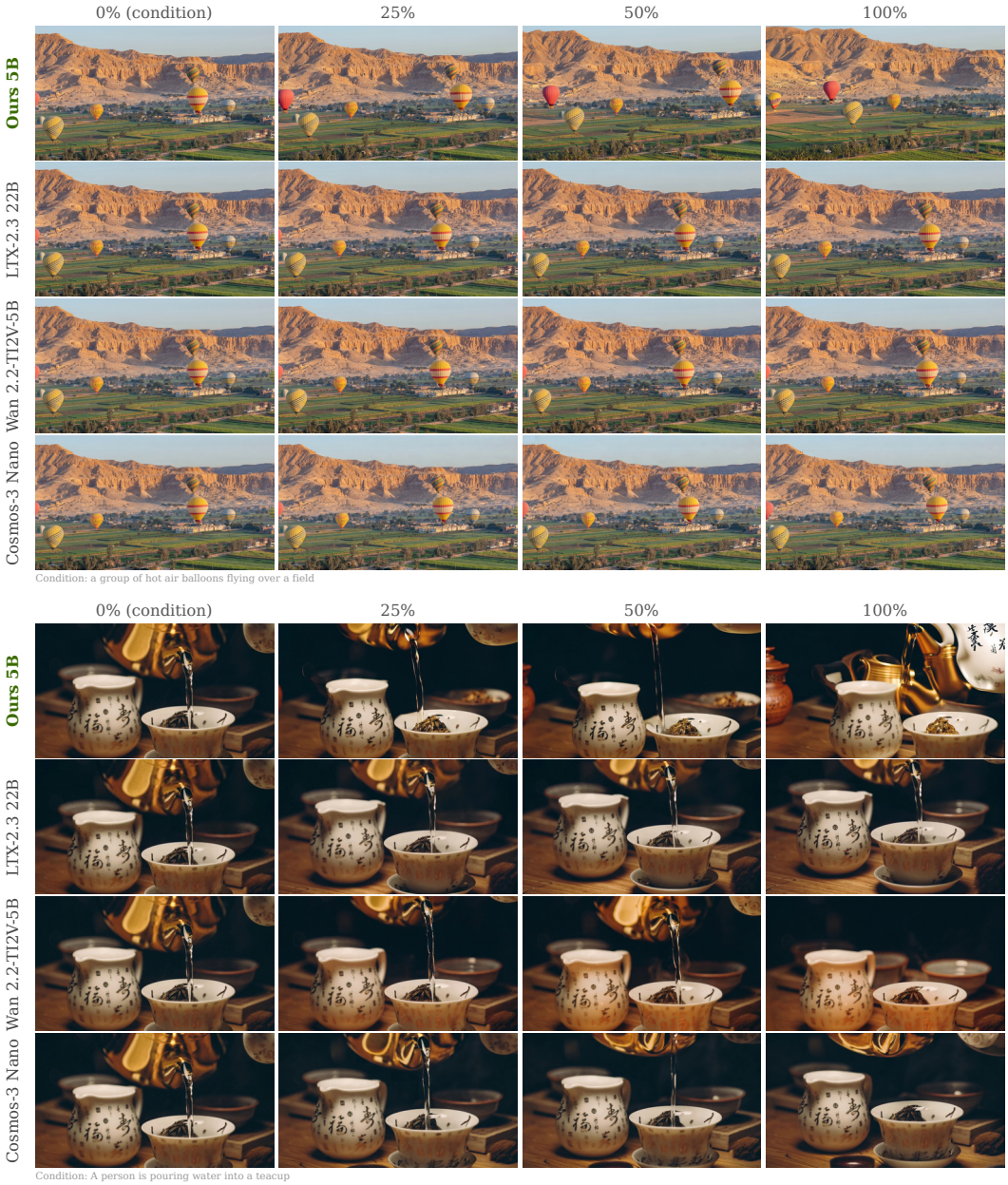


Figure 17: **First-frame-conditioned comparison, pairs 3–4.** Protocol as in Figure 16.

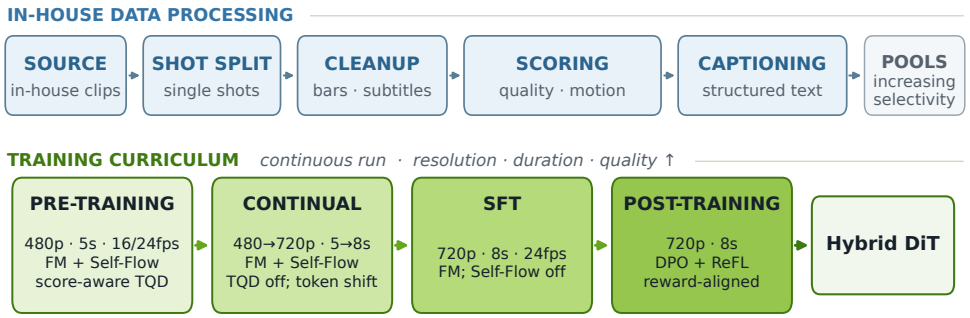


Figure 18: **Training and data pipeline for the 5B checkpoint.** Curated in-house clips feed pre-training, continual training, SFT, and preference-based post-training (DPO + ReFL), with stage-specific data selection, video shapes, and training objectives per stage.

H Full Training Recipe

The 5B production recipe for SANA-Video 2.0 spans data curation, a resolution/duration curriculum, and preference-based post-training (Figure 18). Beyond the architecture, the recipe specifies how the corpus is transformed into progressively cleaner supervision, how noise levels are sampled, and how resolution and duration are increased. We describe these components because they are part of the method. The recipe was selected jointly rather than component-wise ablated. Appendix B lists the architecture and training settings.

H.1 Data Curation and a Quality/Motion Funnel

Starting from the raw video corpus, we apply a uniform processing pipeline: shot segmentation, black-bar and subtitle cleanup, removal of low-resolution/blurry/static/exposure-defective clips, and multi-axis scoring rather than a single aggregate quality score. Keeping the axes separate is deliberate: it prevents selection from collapsing to visually clean but nearly static clips while still retaining high-quality low-motion content. Appendix C gives the complete six-stage pipeline and scoring signals (Table 4). The resulting clips form a progressive quality/motion funnel: pre-training uses a broad pool under permissive thresholds, continual training raises the quality and motion floors, and SFT concentrates on a small trusted subset. Captions progress from mixed short/long descriptions to dense structured ones, and quantized motion descriptors are appended when available to turn the selection signal into a conditioning cue. The same quality and motion axes also drive the pre-training timestep policy below, establishing broad coverage early and focusing later stages on fidelity and prompt adherence.

H.2 Training Objectives and Timestep Sampling

Training uses flow matching with a logit-normal timestep density. In later continual and SFT stages we tail-floor this density, reserving probability mass near the clean and pure-noise extremes that a plain logit-normal undersamples. These regimes respectively correspond to final refinement and global structure. From the continual stage onward we make flow-shift token-count-aware: rather than a fixed per-resolution constant, we anchor the shift at two endpoints and interpolate in log-shift space by latent token count, running log-linearly from 3 at 4,290 latent tokens (480p, 81 frames) to 6 at 23,000 tokens (720p, 193 frames). Each shift is realized by offsetting the (tail-floored, σ=0.95) logit-normal mean by $\log(\text{shift})$, so every intermediate resolution×duration lands on a principled shift and more probability mass moves to high noise as clips lengthen (Table 7). Since the curriculum varies resolution and duration jointly, token count spans a continuous range that discrete per-resolution shifts miss. Each rank therefore computes its shift directly from the local latent shape; because production uses batch size one per rank, the mapping is per-sample.

Content-aware flow-shift (TQD). High-quality video data is scarce, so we want each clip to contribute where it teaches best. This runs into a motion–quality dilemma: motion-rich clips often carry artifacts that lower their per-frame aesthetics, whereas the cleanest clips tend to be nearly static, so a single quality bar sacrifices either motion or fidelity. TQD (Luo et al., 2026) resolves this by routing each clip to the noise regime where its content matters and its weaknesses do not:

Table 7: **Production training curriculum.** Clip counts are reported after filtering and objective add-ons are stage-specific; model and optimizer settings are listed in Table 3.

Stage	Clips	Resolution	FPS	Duration	Flow-shift	LR	Objective add-on
Pre-train	$\sim$30M	480p	16/24	5s	1 (+TQD)	10^{-4}	TQD + Self-Flow
Continual	$\sim$10M	480$\rightarrow$720p	16/24	5s$\rightarrow$8s	3–6 (by tokens)	10^{-4}	Self-Flow
SFT	$\sim 10^4$	720p	24	8s	3–6 (by tokens)	5×10^{-5}	standard FM

high noise governs global structure and motion and masks fine texture, so it receives motion-rich clips, while low noise governs fine detail and receives aesthetically clean, low-motion clips. In pre-training, clips that pass only the high-motion threshold receive a $+1.1$ logit offset, clips that pass only the high-quality threshold receive a -1.1 offset, and clips satisfying both or neither keep the base density. From continual training onward the TQD offset is disabled and the token-count shift takes over. Its minimum shift of 3 approximately matches the median of the high-motion TQD branch, but the change is a stage-level replacement rather than a per-sample continuous handoff. Appendix B.3 gives the bias magnitudes, thresholds, and full sampling densities.

Weight EMA and Self-Flow. We keep an exponential moving average (EMA) of the model weights active throughout training, including SFT. Alongside it, pre-training and continual training additionally carry Self-Flow (Chefer et al., 2026), an auxiliary feature-distillation objective with within-clip dual-timestep conditioning that we use as a training aid rather than a studied component. Self-Flow (Appendix B.2) leaves every latent token supervised and is turned off for SFT while the weight EMA stays on.

H.3 Timestep-Stratified Validation

During training we track checkpoint quality by validation loss, but a single random-t mean is a weak monitor: because loss varies several-fold across the noise axis, one scalar blurs which noise regimes improve or regress and carries high sampling variance. We therefore stratify validation into ten equal 100-step noise buckets, which exposes structure the mean hides. Across four paired checkpoints the bucket macro falls by 6.42%, but that net figure separates a large 11.44% low-noise improvement from a small 1.16% high-noise regression (Figure 8). Stratifying the fixed 100-evaluation budget also cuts the standard deviation of this early-to-final estimate by $2.27\times$ relative to IID timestep draws, giving a more reliable monitor. VBench Total rising from 82.68 to 83.29 is consistent with this net improvement, read descriptively given only four points. Appendix B.4 details the matched-sample protocol and uncertainty analysis.

H.4 Post-Training Stage

H.4.1 Direct Preference Optimization

We employ Diffusion-DPO (Wallace et al., 2024) as a standard post-training step for visual-preference alignment. For each prompt c, Gemini ranks videos sampled with different random seeds by visual fidelity, prompt alignment, motion quality, and temporal coherence. After filtering unreliable rankings, the highest- and lowest-ranked videos form a preferred–rejected pair (x^+, x^-, c). This ranking is performed offline; Gemini is not part of the training loop.

We initialize both the trainable policy θ and a frozen reference model θ_{ref} from the SFT checkpoint. At each training step, the two videos in a pair share the same sampled timestep t and Gaussian noise ϵ. Let q_t denote the forward noising operator and v_ϕ the flow-velocity prediction of model ϕ. The per-video flow-matching errors are

$$x_t^\pm = q_t(x^\pm, \epsilon), \qquad u^\pm = \epsilon - x^\pm, \qquad e_\phi^\pm = \frac{1}{D}\left\| v_\phi(x_t^\pm, t, c) - u^\pm \right\|_2^2, \tag{6}$$

where D is the number of latent elements. Sharing t and ϵ ensures that pairwise differences reflect the videos rather than mismatched corruption levels. We define the preferred–rejected error gap as $\Delta_\phi = e_\phi^- - e_\phi^+$ and optimize

$$\mathcal{L} = -\mathbb{E}[w \log \sigma(\beta(\Delta_\theta - \Delta_{\theta_{\text{ref}}}))] + \lambda\, \mathbb{E}[e_\theta^+]. \tag{7}$$

The first term is Diffusion-DPO, with pair weight w derived from the judge-score gap and preference strength controlled by β. The preferred-sample flow-matching regularizer stabilizes post-training. We update only the policy and keep the reference model fixed.

H.4.2 Online Reinforcement Learning

We further align the model through online reward optimization using Reward Feedback Learning (ReFL) (Xu et al., 2023b). For each text prompt, ReFL first rolls out the denoising trajectory without gradients to a randomly sampled update step. At that state, our diffusion model predicts the clean latent $\hat{x}_0$, which is decoded into a video. We select the first, middle, and last frames of the decoded video and evaluate them with frozen image reward models, using the resulting frame-level feedback to optimize the video. Gradients from the rewards are propagated through the decoder and the model evaluation, avoiding backpropagation through the complete sampling trajectory.

Our joint reward combines HPSv3++ (Liu et al., 2026), DeQA-Score (You et al., 2025), and UniPercept (Cao et al., 2026). The three models provide complementary supervision. HPSv3++ targets wide-spectrum human preference alignment across the capability–iteration spectrum, with particular emphasis on *text fidelity* and *aesthetic quality*. DeQA-Score provides a no-reference perceptual-quality signal: it models continuous human quality judgments through a predicted *score distribution*, making it sensitive to fidelity loss and visible image degradations. UniPercept supplies a finer-grained perceptual profile spanning Image Aesthetics Assessment (IAA), Image Quality Assessment (IQA), and Image Structure and Texture Assessment (ISTA). After normalization and capping, we combine the three reward signals with an HPSv3++:DeQA-Score:UniPercept weight ratio of $4{:}4{:}1$. We additionally apply a velocity-space mean-squared-error regularizer against the frozen base model, balancing preference optimization against preservation of the generative prior acquired during pre-training and supervised fine-tuning. Section J evaluates this online stage.

I AttnRes Router Ablations

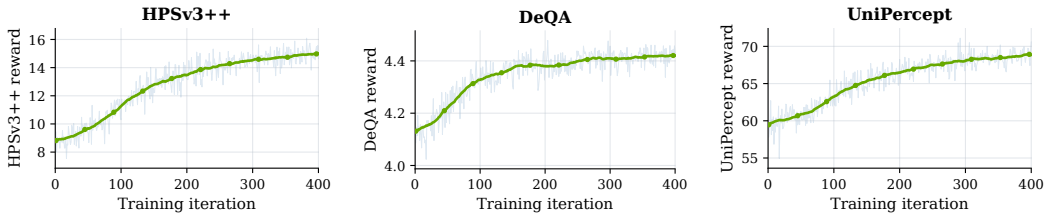
With the 3:1 hybrid ratio fixed, we study AttnRes and select its router design (Table 2).

Router across training (Table 2a,b). During early training, short-run probes select the router formulation: removing its explicit timestep input improves loss from 0.962 to 0.920. After the hybrid backbone has matured, a continued-training comparison finds AttnRes and its removal reach comparable held-out MSE (0.4851 vs. 0.4855, with 17 of 20 noise buckets slightly favoring AttnRes). We do not read a quality advantage from this narrow margin; we adopt AttnRes for the cross-depth reuse it provides and use a timestep-independent router, verifying the intended behavior through the rank and routing probes in Figure 6.

Query design (Table 2c). Per-layer and shared projections have nearly identical loss (0.496 vs. 0.495), but sharing reduces memory overhead $4\times$ (+4% vs. +16%). A separate late-checkpoint probe verifies that routing remains timestep-dependent when the model is trained without an explicit query offset (Figure 6a; protocol in Appendix F.3); this within-checkpoint diagnostic does not compare the quality of query-sharing designs.

Block size (Table 2d). $S \in \{4, 8, 16\}$ yield similar loss point estimates (0.962–0.965), so the short sweep does not identify a preferred span. Memory scales with block count: $S{=}4$ uses +2.9% while $S{=}16$ uses only +1.6%. We keep $S{=}8$ as an engineering default because it gives two regular 3:1 attention cycles per aggregation block, preserves more intermediate depth sources than $S{=}16$, and keeps overhead near $S{=}16$.

The selected shared router remains adaptive without an explicit timestep input: its routing entropy varies across denoising stages through the timestep-conditioned source features. Direct perturbation and weight-space analyses provide the complementary explanation. Replacing the learned router offset by its mean or evaluating it at the wrong timestep changes validation loss by at most 0.0002, while the offset has 0.979 mean cross-timestep cosine similarity and a time-varying residual only 13% as large as its constant component (Appendix F.4).



(a) HPSv3++ preference reward. (b) DeQA perceptual quality. (c) UniPercept mean score.

Figure 19: **All three reward signals improve steadily during online RL training.** Over 400 ReFL iterations, HPSv3++, DeQA-Score, and UniPercept each exhibit a consistent upward trend; pale curves show per-iteration scores, and green curves show their smoothed trajectories.

J Online RL Post-Training Results

We evaluate the online ReFL stage described in Section H.4.2 by tracking its three frozen reward signals throughout training. Across the reported 400-iteration run, the smoothed HPSv3++, DeQA-Score, and UniPercept trajectories all rise together (Figure 19), showing that this run improves all three logged components of the joint objective rather than trading one against another. Appendix E.2 complements these curves with matched-prompt, matched-CFG comparisons between the SFT and RL checkpoints across four examples, providing qualitative context for the changes in visual quality and temporal behavior.

K Deployment and Applications

Kernel-Friendly Deployment with Sol-Engine. The backbone stays close to standard attention and FFN primitives: fixed softmax anchors and a conv-free SwiGLU FFN, with no convolution-adapter path. Removing SANA-Video's temporal-convolution FFN avoids a measured 20–29% overhead at the larger scale (Appendix G.2) and leaves the attention ratio as the main sequence-scaling variable; accordingly, the DiT-forward comparisons in Section 5.4 use equally optimized compiled implementations (FlashAttention for softmax and fused linear attention). Because the backbone maps onto these standard, well-optimized kernels, it composes directly with a production deployment stack. We optimize SANA-Video 2.0 through a three-stage Sol-Engine stack (Li et al., 2026b) on NVIDIA B200 at $736\times1280\times193$ (720p, 8s): kernel and execution optimization first reduces end-to-end latency from 62.65 to 30.74s, residual reuse then evaluates 33 of 50 denoising steps and reaches 20.89s, and sparse attention on the softmax anchors, which dominate module time at long sequences (Appendix Figure 12), reaches 17.52s, for a directly measured $3.58\times$ speedup (Table 8). At the shorter 720p/5s operating point on one H100, the same stack brings the complete 5B pipeline to 13.06s, $120\times$ faster than Wan 2.2-A14B at the matched shape (Figure 1(b)). VBench 84.30 evaluates the base 480p/81-frame, 40-step configuration; it does not measure the quality of the accelerated 720p deployment. These are orthogonal deployment improvements rather than modeling contributions: fusion preserves the computation, while caching and sparse attention introduce approximation. Figure 20 shows matched qualitative outputs from the baseline and the complete accelerated pipeline for the same prompts.



Figure 20: **Qualitative outputs before and after Sol-Engine acceleration.** Two matched prompts compare baseline outputs with the measured 3.58× pipeline at matched points within each clip.

Table 8: **Full stack acceleration of Sol-Engine** on one B200 and one H100 at 720p resolution and 8s duration. Latency includes denoising and VAE decoding. Speedups are relative to each GPU's own baseline.

Stage	NFEs	B200		H100	
		Latency (s)	Speedup	Latency (s)	Speedup
Baseline	50	62.65	1.00×	95.08	1.00×
+ Kernel Optimization	50	30.74	2.04×	59.32	1.60×
+ Diffusion Cache	33	20.89	3.00×	40.07	2.37×
+ Sparse Attention	33	**17.52**	**3.58**×	**33.43**	**2.84**×

Low-Precision Inference with QAT. We quantize the SANA-Video 2.0 5B backbone using MXFP4 weights and MXFP8 activations (Table 9). QAT matches the BF16 baseline on VBench Total, while PTQ is slightly lower. On an NVIDIA GB200, one CFG-packed backbone forward for an 81-frame video at 832×480 resolution decreases from 203.08 to 191.30ms (5.8%); static model storage decreases from 8.94 to 2.87GB (67.9%), and peak allocated memory from 10.74 to 4.63GB (56.9%). The latency gain is modest because the selected GEMMs occupy only 16.2% of BF16 runtime: W4A8 reduces them from 33.15 to 22.16ms, and therefore saves only about 11ms overall. We currently quantize only linear GEMMs; the linear-attention and softmax-attention operators remain in BF16.

Table 9: **QAT quality and low-precision efficiency.** VBench scores (%) and median systems measurements for one CFG-packed 81-frame backbone forward at 832×480 on NVIDIA GB200. QAT matches the BF16 baseline, while PTQ is slightly lower; PTQ and QAT share the low-precision inference configuration. Memory uses decimal GB.

		VBench				GB200 systems		
Method	Precision	Quality	Semantic	Total	Δ	Latency (ms)	Static (GB)	Peak (GB)
BF16	BF16	83.98	80.14	83.22	—	203.08	8.94	10.74
PTQ	MXFP4-W/MXFP8-A	83.53	80.23	82.87	−0.35	**191.30**	**2.87**	**4.63**
QAT	MXFP4-W/MXFP8-A	83.95	80.43	**83.25**	+0.03			



Figure 21: **Qualitative comparison with Physical AI models.** In the presented comparison, SANA-Video 2.0 outperforms the similar-sized Cosmos3-Edge (NVIDIA, 2026) (4B) and delivers competitive results against much larger models, including Cosmos3-Nano (NVIDIA, 2026) (16B) and Lingbot-Video (Ma et al., 2026) (30B).

Physical AI. SANA-Video 2.0 holds strong potential for Physical AI scenarios, especially efficiency-sensitive tasks in robotics and self-driving. We fine-tune SANA-Video 2.0 on roughly 5,000 hours of publicly available real-robot and egocentric videos for 100k iterations at a learning rate of 1×10^{-4}. As shown in Figure 21, SANA-Video 2.0 produces realistic and physically plausible robot-manipulation videos. It outperforms the similar-sized Cosmos3-Edge (NVIDIA, 2026) (4B) in the presented comparison and delivers competitive results against much larger models, including Cosmos3-Nano (NVIDIA, 2026) (16B) and Lingbot-Video (Ma et al., 2026) (30B).

L Future Work

Two directions follow most directly from this design. The measured efficiency advantage increases with duration over the profiled range (Section 5.4). Extending the curriculum beyond the current 8s horizon toward minute-scale training would test whether these computational savings translate into efficient generation with long-range temporal consistency. Second, our operator is bidirectional, whereas robotics, autonomous driving, and world models require streaming, action-conditioned rollouts: as noted in Section 2, dropping the delta-rule update makes it a natural initialization for a causal Gated DeltaNet, so pairing a causal delta-rule linear operator in the Kimi-Linear (Kimi Team, 2025) / Kimi K3 (Kimi Team, 2026a) family with the same hybrid-plus-AttnRes layout would carry this scratch-trained, kernel-friendly recipe into causal generation and closed-loop world models, complementing the Physical AI study in Section 6. Further extensions include anchor-specific sparse kernels and calibrated low-precision (NVFP4) formats, few-step distillation to complement the training-free deployment caching, and learned, content-adaptive anchor placement and AttnRes routing.