SPARKDIFFUSION: A UNIFIED SPARSE WARM-UP AND FEW-STEP DISTILLATION FRAMEWORK FOR MITIGATING THE HIGH-SPARSITY TRAP

Anonymous authors
Paper under double-blind review

ABSTRACT

Video diffusion transformers are central to visual generation, but inference is dominated by attention over long spatiotemporal sequences. While video attention's sparse-plus-low-rank structure suggests high sparsity is possible, existing models degrade beyond $\sim$ 90% sparsity even after step-local training converges. We identify this failure regime as the *high-sparsity trap*: step-local objectives leave terminal-visible velocity errors that accumulate during sampling without direct gradient signals to correct them. We propose SPARKDIFFUSION (Terminal-Aligned Sparse Acceleration), a unified framework whose central principle is to first adapt a dense video DiT into a high-sparsity coarse prior through sparse warm-up, and then correct its terminal trajectory through few-step trajectory-mixed distillation; fused FP8 kernels provide the deployment layer. Our idealized analysis isolates the mechanism and limits, while experiments validate the staged framework across multiple backbones, tasks, resolutions, and hardware platforms.



Figure 1: **SparkDiffusion teaser.** From one pretrained dense video DiT, SPARKDIFFUSION delivers high-quality frames across Wan2.1/2.2, T2V/I2V, and 480P/720P in 3 steps, sustaining 97% attention sparsity on long-sequence 720P models and 90% on compact Wan2.1-T2V-1.3B-480P.

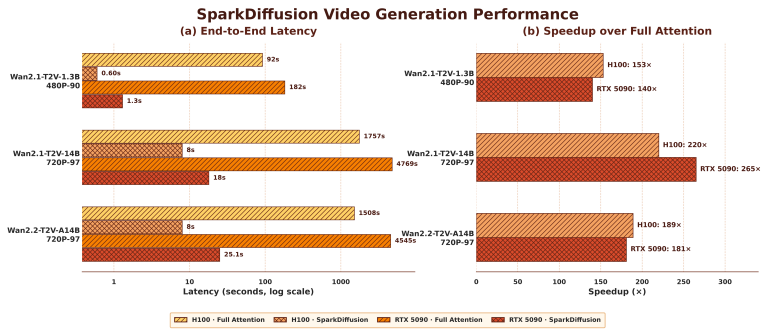

Figure 2: Generation latency and full-stack speedup of SPARKDIFFUSION over Full Attention across Wan models on H100 and RTX 5090 GPUs. Suffixes "90" and "97" denote sparsity levels. These ratios combine few-step CFG-free distillation, sparse attention, and FP8 deployment and therefore should not be interpreted as the gain from sparsity alone; NFE follows footnote 1.

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.

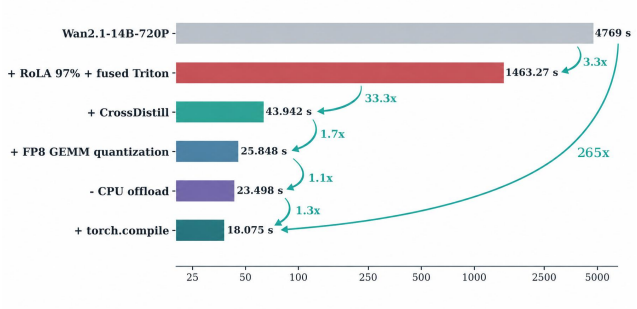

Figure 3: Full-stack efficiency decomposition on Wan2.1-T2V-14B-720P. The reported $265\times$ is $4769/18.0$: it combines 100-to-3 NFE reduction, 97% sparse attention, and FP8 deployment, rather than measuring sparsity alone. At matched 3-step inference, $90\% \rightarrow 97\%$ sparsity contributes $23.7/18.0 = 1.32\times$ (footnote 1).

1 INTRODUCTION

Video diffusion transformers (Peebles & Xie, 2023) are a central architecture for high-quality text-to-video and image-to-video synthesis (Wan et al., 2025; Kong et al., 2024; Yang et al., 2025), but their inference cost is dominated by attention over long spatiotemporal sequences at every sampling step. Sparse attention reduces this cost by skipping redundant query–key interactions, yet conventional designs lose global context and degrade structure when pushed to very high sparsity (Xi et al., 2025; Yang et al., 2026; Zhang et al., 2025a). Compensated sparse attention mitigates this problem by coupling a high-energy sparse branch with a lightweight low-rank or linear branch (Zhang et al., 2026a;b; Fang et al., 2026; Liu et al., 2026c), enabling around 90% attention sparsity.

Since block-sparse attention computes only a fraction $1 - s$ of query–key blocks, the sparse-branch computation scales roughly linearly with $1 - s$. Increasing sparsity from 90% to 97% reduces the retained sparse blocks from 10% to 3%, cutting the block-wise sparse-attention computation to about 30% of that at 90% sparsity. This reduction translates directly into end-to-end latency gains: on Wan2.1-T2V-14B-720P with 3-step inference, the $90\% \rightarrow 97\%$ stretch alone reduces latency from $10.5\,\text{s}$ to $8.0\,\text{s}$ on H100 and from $23.7\,\text{s}$ to $18.0\,\text{s}$ on RTX 5090 (Fig. 2)—an additional 1.31–$1.32\times$ speedup that is available only if quality survives extreme sparsity.[1]

However, extreme sparsity also exposes a second barrier: a sparse model may have fast inference and converged step-local training loss, while its terminal videos still exhibit broken structure, semantic drift, mosaic textures, and temporal flicker. We refer to this regime as the *high-sparsity trap*: a sparse generator can converge under step-local supervision while still failing terminally.

This raises the central question:

> **Can a pretrained visual generator maintain terminal quality at $90\%+$ attention sparsity, and what post-training sequence makes this possible?**

Figure 4 provides a mechanistic diagnosis: at extreme sparsity, replacing the sparse student's velocity with the dense teacher's velocity only in a high-noise window substantially reduces the terminal error, whereas the same correction in a low-noise window yields limited gain (setup and details in Sec. 3). Video attention exhibits a strong sparse-plus-low-rank structure that makes 90%+ sparsity reachable in representation (Liu et al., 2026c); the remaining bottleneck is supervision (evidence in Sec. 3). Sparse video diffusion models typically inherit flow matching (Lipman et al., 2022), which is *step-local*: it supervises velocities at individual noise levels but does not directly constrain the terminal sample, so a highly sparse model can minimize step-local loss while still producing terminal samples misaligned with the data distribution.

This observation motivates SPARKDIFFUSION, a staged post-training framework. A short sparse warm-up adapts the dense backbone to the sparse architecture and provides a trainable coarse prior. Trajectory-mixed distillation then corrects the terminal distribution by applying high-noise structural alignment and low-noise distribution matching. Finally, fused FP8 deployment converts the reduced

[1] By default, Full Attention results use 50 steps with classifier-free guidance (2 evaluations/step, NFE $= 50 \times 2 = 100$); accelerated results are CFG-free in 3 steps (NFE $= 3$). The baseline runs in BF16 using standard dense attention kernels—FlashAttention-2 on RTX 5090 and FlashAttention-3 on H100

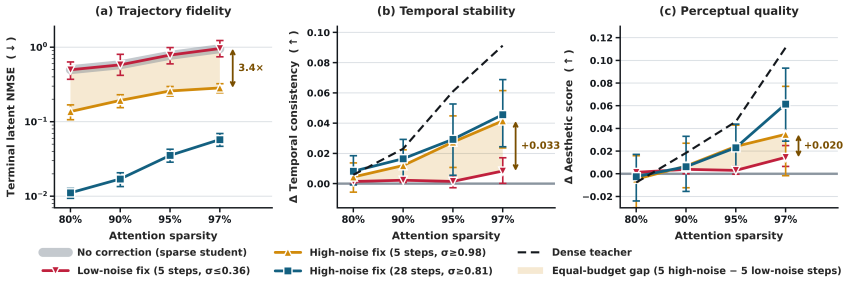

Figure 4: **The high-sparsity trap is rooted in high-noise structure errors.** *Oracle probe*: the sparse model's guided velocity is replaced by the dense teacher's within a window of sampling steps (Wan2.1-T2V-14B-480P, 50-step sampler, 32 prompts × 4 seeds, error bars: 95% CI; both models use CFG, isolating sparsity-induced error). (a) Terminal error to the dense reference (latent MSE, ↓). (b,c) Gains over the uncorrected sparse model (↑; dashed: teacher). At 97% sparsity, fixing the five highest-noise steps removes most of the terminal error, while the equal-budget low-noise fix leaves it essentially unchanged (beige band); a wider 28-step high-noise window recovers nearly all of it.

computation into measured wall-clock speedup. The same staged recipe applies to T2V/I2V tasks and to both dense Wan2.1 and MoE-based Wan2.2 backbones (Sec. 5).

Contributions.

- **Unified staged framework.** We introduce SPARKDIFFUSION, which first uses a short sparse warm-up to adapt the dense backbone into a usable high-sparsity prior, then uses few-step trajectory-mixed distillation to correct the terminal distribution. This staging, rather than a new sparse selector or distillation primitive, is the core framework novelty; FP8 fused kernels provide deployment.
- **The high-sparsity trap and its remedy.** We identify a regime in which step-local loss decreases while terminal generation stagnates or degrades, localize its dominant error to high-noise structure generation, and show that terminal-aligned post-training mitigates it where extended step-local training does not.
- **Empirical validation.** Across Wan2.1/Wan2.2, T2V/I2V, and 480P/720P, SPARKDIFFUSION sustains up to 97% attention sparsity with competitive quality. The 265× result is a full-stack measurement combining step reduction, sparsity, and FP8, not a sparsity-only claim.

Extension to auto-regressive video diffusion. The framework is also compatible with auto-regressive (AR) video diffusion Yin et al. (2025). Compensated sparse attention can be restricted to a causal window. Trajectory-mixed distillation acts on the noise axis rather than the temporal axis, so it can be combined with self-forced AR training Huang et al. (2026). Since high-noise structural errors can propagate across chunks, terminal alignment is arguably even more important in AR generation.

2 PRELIMINARIES

The following ingredients are central to our discussion: flow-matching training, block-sparse attention, and compensated sparse attention.

Flow matching. Following flow matching (Lipman et al., 2022), we consider a video diffusion transformer that predicts a velocity field $F_\theta(x, t)$ on latent tokens $x \in \mathbb{R}^{L\times d}$, where L is the spatiotemporal token length. Let F_{θ_0} denote a pretrained dense checkpoint. Given a clean sample $x_0 \sim p_{\text{data}}$ and Gaussian noise $\epsilon \sim \mathcal{N}(0, I)$, flow matching forms the interpolated noisy sample

$$x_t = (1-t)x_0 + t\epsilon, \qquad t \in [0, 1],$$

and trains the model to predict the path velocity $\epsilon - x_0$:

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{x_0,\epsilon,t}\left[w(t)\,\|F_\theta(x_t, t) - (\epsilon - x_0)\|_2^2\right], \tag{1}$$

where $w(t)$ is an optional timestep weighting function. This objective supervises velocities at sampled noise levels; the terminal sample $\hat{x}_0$ is obtained only after composing many sampling steps.

Block sparsity. Here, *attention sparsity* refers to *block sparsity*: the attention matrix is partitioned into contiguous query–key blocks, and a fraction s of them are skipped during computation. We report sparsity as the fraction of blocks skipped by the sparse branch.

Compensated sparse attention. A major cost in video DiTs is dense self-attention, scaling as $O(L^2)$ with token length L. Compensated sparse attention replaces it with a gated sum of two branches: a sparse branch A_M computing exact softmax on a $1 - s$ fraction of selected query–key blocks, and a lightweight compensation branch C_ψ (linear/low-rank attention or pooled summaries) recovering discarded global context, with the gate initialized near zero (details in Sec. 4.1).

Training sparse attention. Starting from the dense checkpoint F_{θ_0}, we insert the compensated sparse attention modules into the backbone and train the resulting model with the same task loss as in (1). Some variants replace the target $\epsilon - x_0$ with the dense teacher prediction $F_{\theta_0}(x_t, t)$, but the objective remains a step-local velocity-regression loss.

3 THE HIGH-SPARSITY TRAP

We use the term *high-sparsity trap* to describe a specific failure regime at extreme sparsity. Mitigating the trap unlocks the largest remaining latency gain in sparse attention (Sec. 1).

> **The high-sparsity trap.** We say that a sparse video DiT falls into the high-sparsity trap when, under a compensated sparse architecture, increasing attention sparsity beyond a threshold causes terminal generation quality to stagnate or degrade while the step-local validation loss remains low or continues to decrease, and the terminal error persists even under substantially extended step-local training budgets.

Generality across sparse designs. RoLA (Zhang et al., 2026d) shares its sparse branch with VSA (Zhang et al., 2026c) and SLA (Zhang et al., 2026a): all compute exact softmax attention on a retained block subset and differ only in the compensation mechanism. Appendix D reproduces the trap on VSA- and SLA-style designs, and the staged recipe of Sec. 4 restores terminal quality on both.

Terminal error. In this section, *terminal error* denotes the paired MSE between sparse and dense terminal outputs under matched initial noise; by default, it is measured in latent space. Both models use identical settings—50 steps with CFG (scale 5.0)—so the terminal error reflects sparsity-induced error alone, not CFG removal or step reduction.

3.1 EMPIRICAL SIGNATURE: TERMINAL DRIFT AT EXTREME SPARSITY

High sparsity is not ruled out by architecture alone: video attention exhibits a strong sparse-plus-low-rank structure that supports 90%+ sparsity Liu et al. (2026c), and indeed the 80% and 90% models remain close to the dense baseline under the same training (Figure 6); the failure appears only when the same architecture family is pushed to 95% and 97%. Two observations identify the training signal, rather than the representation, as the bottleneck. First, widening the compensation branch from rank-64 to full rank (r=128) leaves the terminal error essentially unchanged (0.124 → 0.121 paired latent MSE, VBench 80.92 → 81.05; Wan2.1-T2V-14B-480P, 10,000 steps): the model can already express far richer corrections than the ones it converges to. Second, the objective it is given is still being optimized: the step-local validation loss continues to decrease while terminal quality stalls (Figure 7).

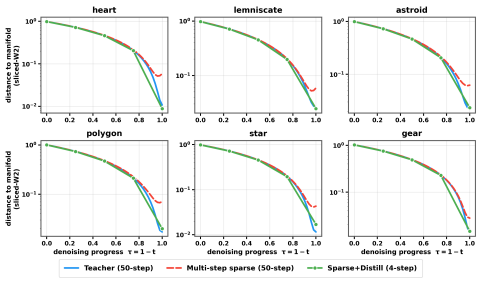

Figure 5: Sliced-W_2 distance to the data distribution along the denoising trajectory on six toy 2D sequence manifolds. The sparse model is trained until validation loss plateaus, while the distilled student is obtained by trajectory-mixed distillation.

Evidence in a controlled setting. On toy 2D sequence manifolds, a 95% sparse model trained to convergence still exhibits structural deviations from the high-noise stage; these amplify as denoising proceeds (Figure 5). Figure 16 shows the qualitative counterpart.

Real-video evidence under controlled training budgets. On a real video backbone, all sparse models are trained with the same step-local task loss (1) from the same dense checkpoint on an enlarged training set of ≈30,000 videos (15× the production warm-up set), and training is deliberately extended to 10,000 steps—40× the step-local budget of the production warm-up (≈250 steps), and more than the total step count of the full two-stage pipeline of Sec.4 (8,250 steps; Tables 7 and 8).



Figure 6: Video frame comparison on Wan2.1-T2V-14B-480P under attention sparsity levels 80%, 90%, 95%, and 97%.

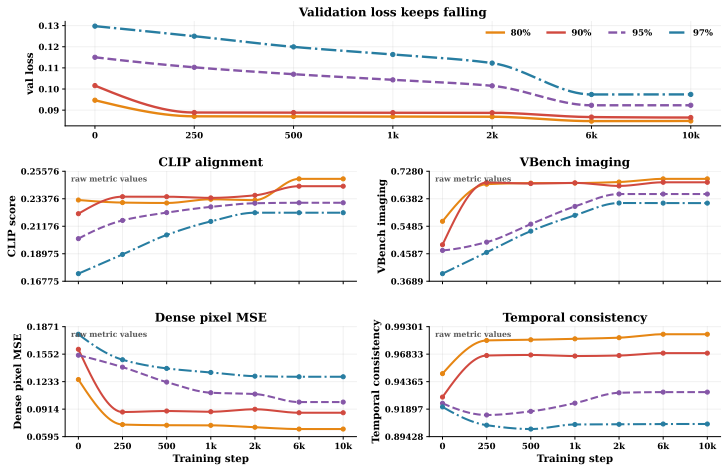

Figure 7: Validation loss and raw quality metrics for Wan2.1-14B across sparsity levels. The "Dense pixel MSE" panel reports the terminal error of Sec.3 after VAE decoding, sharing the pixel-space footing of the other quality panels.

Where the error is injected. Figure 4 localizes the terminal error with an oracle intervention: for a fixed initial noise, the sparse student's velocity is replaced by the dense teacher's velocity on a contiguous noise interval, while the sparse student is kept elsewhere. At 97% sparsity, correcting the five highest-noise steps removes most of the terminal error, whereas the equal-budget low-noise correction leaves it essentially unchanged; a wider high-noise window recovers nearly all of it. Structural errors are therefore injected during high-noise structure generation and then amplified by later steps. These diagnostics show step-local validation loss is a poor proxy for terminal quality at extreme sparsity.

3.2 Why step-local supervision may be insufficient

Intuitively, the step-local loss measures per-step error magnitudes, whereas the terminal sample depends on how those errors are transported and summed along the sampling trajectory. Appendix C formalizes this mechanism in a deliberately simplified surrogate. After a step-local optimum, a terminal-aligned signal can reduce the surviving terminal error at first order through the same trainable parameters, but it cancels the terminal-visible projection of the error rather than removing the underlying velocity error.

3.3 From diagnosis to a staged remedy

We call a training signal *terminal-aligned* if its supervision target at noise level t is constructed from the sampling trajectory *beyond* t—the model's prediction at a later trajectory state, a teacher-composed trajectory segment, or the terminal output itself—so that the loss constrains how the prediction at t *composes* with the remainder of the trajectory toward the terminal state. In contrast, step-local losses such as flow matching supervise the velocity at (x_t, t) against a closed-form, pointwise target that is independent of the sampling trajectory, and are therefore agnostic to how per-step errors compose and accumulate (Appendix C). Under this definition, consistency objectives are terminal-aligned toward

the terminal state of their noise interval, and distribution-matching objectives are terminal-aligned toward the global terminal output $\hat{x}_0$. Sparse training with step-local loss provides a coarse prior but does not anchor the terminal distribution at extreme sparsity. Applying few-step distillation post-training to the sparse model with this prior is therefore a natural way to mitigate the high-sparsity trap. The remedy is staged: a short sparse warm-up adapts the high-sparsity architecture into a usable prior, and trajectory-mixed distillation (Sec. 4) then aligns the teacher's velocity field in the high-noise segment and matches the data distribution in the low-noise segment.
At 97% sparsity, the few-step model after trajectory-mixed distillation is visually much closer to the dense teacher than the multi-step sparse model (Appendix B.1, Fig. 8).

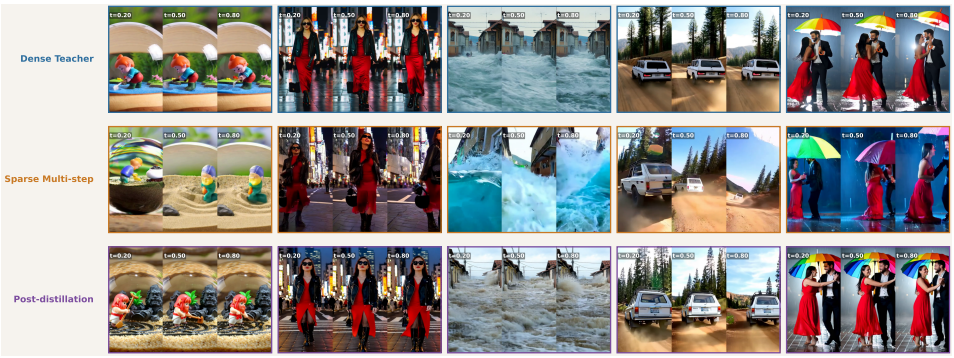

Figure 8: Qualitative comparison at 97% attention sparsity on Wan2.1-T2V-14B-720P. We compare the dense teacher, the multi-step sparse model, and the few-step sparse model obtained after trajectory-mixed distillation.

4 METHOD

SPARKDIFFUSION consists of two post-training stages and one deployment-time step. Figure 9 gives an overview of the framework. Together, they convert a pretrained dense video DiT into a fast, high-sparsity generator without retraining it from scratch. Stage 1 performs a sparse-attention warm-up to obtain a coarse prior (Sec 4.1); Stage 2 applies trajectory-mixed distillation to reduce sampling steps and correct the terminal distribution (Sec 4.2); Stage 3 uses FP8 quantization to convert reduced computation into wall-clock speedup (Sec 4.3). Each stage consumes the checkpoint of the previous one; the full schedule is in Sec 4.4.
The framework is agnostic to the specific sparse selector, compensation branch, and distillation objective. Our default instantiation uses RoLA (Zhang et al., 2026d) as the sparse module and the CrossDistill schedule (Liu et al., 2026b) for distillation.

4.1 COMPENSATED SPARSE ATTENTION

Our sparse attention module follows RoLA (Zhang et al., 2026d). A block-sparse branch retains high-energy query–key blocks, computing $O_s = A_M(Q, K, V)$ on the selected blocks, while a low-rank branch recovers the discarded global context: it applies low-rank projections, a SiLU nonlinearity, and a rank-truncated 3D RoPE rotation to the backbone's pre-attention normalized queries and keys, and is therefore linear in sequence length while preserving relative spatiotemporal position:

$$\tilde{q}_i = R_r(p_i)\,\phi(P_q q_i^{\mathrm{norm}}), \qquad \tilde{k}_j = R_r(p_j)\,\phi(P_k k_j^{\mathrm{norm}}),$$

$$C = \sum_j \tilde{k}_j v_j^\top, \qquad O_{lr,i} = \mathrm{RMSNorm}(\tilde{q}_i^\top C),$$

where ϕ is SiLU, and $R_r(\cdot)$ reuses the first r rotary coordinates of the pretrained 3D RoPE schedule. A token-wise gate g_i initialized near zero fuses the two branches:

$$O_i = O_{s,i} + g_i\, r_i\, O_{lr,i},$$

where r_i is the RMS of the sparse output $O_{s,i}$. Thus the module starts close to the sparse branch and opens the compensation path only where needed.

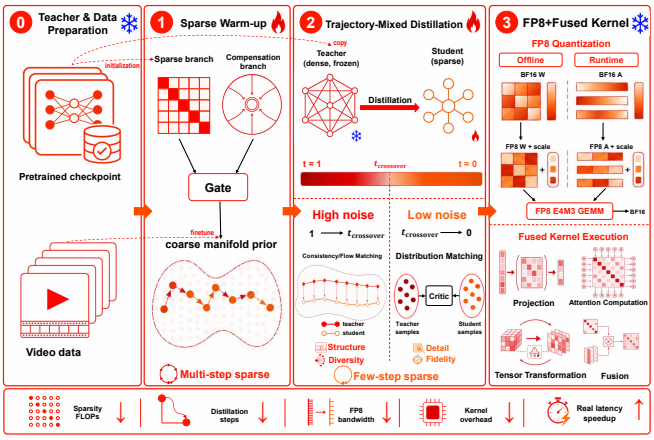

Figure 9: Overview of the SPARKDIFFUSION framework (stages detailed in Sec. 4.1–4.3).

4.2 TRAJECTORY-MIXED DISTILLATION

Stage-2 distillation must meet two distinct requirements. First, escaping the high-sparsity trap requires a terminal-aligned signal (Sec. 3); consistency-style and distribution-matching objectives are both terminal-aligned, so either suffices in principle (Sec. 5.3). Second, the choice among them is governed by the fidelity–diversity trade-off: pure distribution matching is mode-seeking and loses seed-level diversity (Chen et al., 2026b), while pure consistency matching preserves the teacher's trajectory but never aligns the terminal distribution.

We therefore adopt the trajectory-mixed schedule of CrossDistill (Liu et al., 2026b), which splits the sampling trajectory at a noise crosspoint: the high-noise segment is trained with PCM-styleWang et al. (2024) consistency to preserve structure, motion, and diversity, while the low-noise segment is trained with DMD-styleYin et al. (2024b) distribution matching to sharpen detail and correct terminal-visible errors. We use a 3-step student—one high-noise PCM step and two low-noise DMD steps—and follow the default crosspoint, loss weighting, and optimization schedule of CrossDistill. This assignment mirrors the diagnosis of Sec. 3: terminal-visible errors originate in high-noise structure generation (Fig. 4), so the high-noise step stays anchored to the teacher's coarse trajectory while the low-noise steps supply the terminal-aligned correction.

4.3 FP8 QUANTIZATION

The deployment step converts the sparse model's reduced FLOPs into wall-clock speedup. We quantize the linear projections in attention and the feed-forward layers to W8A8 FP8 (E4M3) (Shen et al., 2024): weights carry static per-channel scales calibrated once offline, and activations are scaled dynamically per token at runtime. Normalization layers, gating operations, and sparse mask selection remain in BF16. The per-token activation quantization path is fused into a single custom kernel so that scaling and casting happen in one pass over memory. This transform is applied purely at deployment time, after all training is complete. All SPARKDIFFUSION quality numbers in this paper are measured on the deployed FP8 model; the BF16-to-FP8 delta stays within 0.1 VBench points (Appendix B.2).

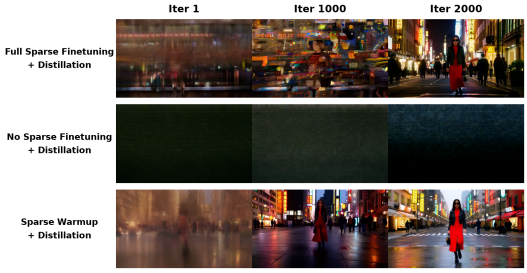

Figure 10: **Sparse warm-up provides the coarse prior for terminal-aligned distillation.** On Wan2.1-T2V-14B-480P, all models are followed by the same trajectory-mixed distillation stage; a short sparse warm-up establishes the usable coarse prior on which distillation builds.

4.4 TRAINING AND DEPLOYMENT

Stage 1 (sparse warm-up). We insert the compensated sparse attention of Sec 4.1 into the dense backbone and fine-tune briefly with the step-local objective of Sec. 2 (Eq. (1), including its teacher-velocity variant) alone. This stage adapts the backbone to the sparse architecture and establishes a coarse generative prior for the terminal-aligned stage that follows. Figure 10 shows that a brief warm-up is sufficient to establish a usable prior for distillation, so we keep this stage short.

Table 1: Generation quality and efficiency on VBench (Huang et al., 2024) and VBench-2.0 (Zheng et al., 2025) (overall score plus five capability categories). Wan2.1-T2V-1.3B is evaluated at 480×832, Wan2.1-T2V-14B and Wan2.2-T2V-A14B at 720×1280, all with 81 frames; NFE conventions follow footnote 1. All SPARKDIFFUSION rows are measured with the full deployment stack (W8A8 FP8 with fused kernels) for both quality and latency; baselines use their official inference pipelines; Full Attention uses our strongest dense implementation (BF16, FlashAttention-2/3). **Bold** marks the best accelerated result within each model block. Further implementation notes: Appendix E.

Model	Method	VBench	VBench-2.0							Latency (s)	
		Total ↑	Total ↑	Creat. ↑	Commo ↑	Control ↑	Human Fid. ↑	Physics ↑	Sp.	5090 ↓	H100 ↓
Wan2.1-T2V 1.3B	Full	83.21	56.02	54.73	57.38	34.96	78.71	54.30	0%	182	92
	FastWan (VSA)	82.37	54.63	52.66	56.52	32.27	79.69	52.01	90%	2.8	1.2
	TurboDiffusion	82.52	54.61	51.72	56.94	31.65	79.55	53.21	90%	2.0	1.0
	SparkDiffusion (Ours)	**82.64**	**55.85**	**53.99**	**57.13**	**33.75**	**80.34**	**54.02**	90%	**1.3**	**0.6**
Wan2.1-T2V 14B	Full	83.69	60.20	55.25	63.98	37.32	81.60	62.84	0%	4769	1757
	FastWan (VSA)	82.72	58.04	54.97	59.54	35.75	81.72	58.20	90%	54.1	20.5
	TurboDiffusion	82.88	57.98	54.88	59.77	34.98	81.51	58.74	90%	25.3	16.0
	SparkDiffusion (Ours)	**83.42**	**59.36**	**55.22**	**60.88**	**37.41**	**83.51**	**59.77**	90%	23.7	10.5
	SparkDiffusion (Ours)	83.15	58.05	54.81	59.02	36.01	81.77	58.63	97%	**18.0**	**8.0**
Wan2.2-T2V A14B	Full	84.21	60.36	55.60	64.50	37.40	81.30	63.00	0%	4545	1508
	SparkDiffusion (Ours)	**83.75**	**59.77**	**55.51**	**61.36**	**37.55**	**83.88**	**60.53**	90%	32.4	10.5
	SparkDiffusion (Ours)	83.36	58.46	55.09	59.32	35.67	81.92	60.30	97%	**25.1**	**8**

Stage 2 (trajectory-mixed distillation). Starting from the warm-up checkpoint, we apply the trajectory-mixed distillation of Sec. 4.2, with the dense multi-step model frozen as the teacher.

Deployment. The few-step model is quantized with the FP8 scheme of Sec 4.3 and used directly for inference. For mixture-of-experts backbones such as Wan2.2, where separate expert groups are specialized for different noise ranges, each expert group is warmed up and distilled on its own noise range before quantization.

5 EXPERIMENTS

5.1 MAIN QUANTITATIVE COMPARISON

Table 1 compares Full Attention, TurboDiffusion, FastWan (VSA), and SPARKDIFFUSION under the same hardware and generation settings.[2] All baselines are evaluated at their official released operating point (90% sparsity); we report SPARKDIFFUSION both at matched 90% and at 97%. At matched 90% sparsity, SPARKDIFFUSION improves over all baselines on every reported metric on Wan2.1-T2V-14B-720P. On the compact short-sequence Wan2.1-T2V-1.3B-480P, SPARKDIFFUSION at the same 90% sparsity improves generation quality and inference speed simultaneously over both baselines: it attains the best VBench-2.0 at the lowest latency (1.3 s on RTX 5090 and 0.6 s on H100, 1.5–2.2× faster than the baselines).

On the large long-sequence models, we push sparsity to 97%. All baselines are evaluated at their strongest official configuration (90% sparsity); since quality degrades monotonically with sparsity (Fig. 6), training the baselines at 97% could only widen the quality gap in our favor. At 97%, SPARKDIFFUSION attains slightly better aggregate quality than the strongest 90% baselines (on Wan2.1-T2V-14B-720P, VBench and VBench-2.0 relative to TurboDiffusion) at clearly lower latency, while remaining closest to the dense model in diversity (Table 2). End-to-end latency (excluding text encoding and VAE decoding) and speedups are shown in Fig. 2; the isolated $90\% \rightarrow 97\%$ stretch contributes 1.31–1.32× at matched 3-step inference (Sec. 1)—the regime where step-local-only recipes degrade (Sec. 3).

5.2 DIVERSITY PRESERVATION UNDER FEW-STEP DISTILLATION

Table 2 quantifies seed-level diversity, complementing the qualitative check in Appendix B.1 (Fig. 17). Following Shaul et al. (2026), we encode each video with two frozen video encoders, V-JEPA

[2]TurboDiffusion and FastWan release official weights only for Wan2.1 backbones, so they are omitted for Wan2.2-T2V-A14B. FastWan is the few-step sparse model released in the official VSA repository (Zhang et al., 2026c); we treat them as one baseline, denoted FastWan (VSA).

Table 2: Quantitative diversity comparison on Wan2.1-T2V-14B-720P. Each video is encoded by two frozen video encoders, V-JEPA 2 (Assran et al., 2025) and VideoMAE V2 (Wang et al., 2023); we report the average pairwise cosine and ℓ_2 distances among $N=5$ videos sampled from the same prompt with 5 noise seeds (1,000 prompts; protocol follows Shaul et al. (2026)).

		V-JEPA 2		VideoMAE V2	
Method	**Sp.**	**Cos ↑**	**ℓ_2 ↑**	**Cos ↑**	**ℓ_2 ↑**
Full Attention	0%	0.125	27.15	0.0252	2.83
FastWan (VSA)	90%	0.075	21.31	0.0117	2.05
TurboDiffusion	90%	0.078	21.67	0.0125	2.21
SparkDiffusion (Ours)	97%	**0.087**	**23.04**	**0.0142**	**2.47**

2 (Assran et al., 2025) and VideoMAE V2 (Wang et al., 2023), and report average pairwise cosine and ℓ_2 distances among videos sampled from the same prompt with different noise seeds; higher values indicate more diverse samples, and the dense model serves as a reference. Despite operating at 97% sparsity, SPARKDIFFUSION stays closest to the dense reference, whereas the few-step baselines lose a visible fraction of seed-level variation, consistent with the mode-seeking tendency of distribution matching (Chen et al., 2026b). This supports the trajectory-mixed design of Sec. 4.2: high-noise consistency matching preserves coarse structure and diversity, while low-noise distribution matching improves terminal fidelity.

5.3 DISTILLATION OBJECTIVE AT EXTREME SPARSITY

Table 3 ablates the Stage-2 objective at 97% sparsity, with all students initialized from the same Stage-1 warm-up checkpoint and sampled with three CFG-free steps. Both single-objective variants already recover most of the quality lost at 97% sparsity, corroborating that the trap is one of supervision: any terminal-aligned objective mitigates it. The two pure objectives, however, fail in opposite directions along the fidelity–diversity axis: PCM-only training tracks the teacher's trajectory but never aligns the terminal distribution, yielding the weakest scores, while DMD-only training aligns the terminal distribution but is mode-seeking, losing coarse structure and seed-level diversity (cf. Table 2). The trajectory-mixed objective attains the best of both segments; together with the warm-up ablation (Fig. 10), this supports the staging principle of Sec. 3.

Table 3: Distillation-objective ablation at 97% attention sparsity (Wan2.1-T2V-14B-720P; 3-step CFG-free student; All 3-step student rows initialize from the same Stage-1 warm-up checkpoint.)

Stage-2 objective	**VBench ↑**	**VBench-2.0 ↑**
Full Attention (dense, 50-step)	83.69	60.20
PCM only (3-step)	81.94	56.41
DMD only (3-step)	82.56	57.38
CrossDistill (Ours, 3-step)	**83.15**	**58.05**

5.4 CROSS-MODEL QUALITATIVE VALIDATION

Appendix B.1 reports qualitative comparisons across model scales (1.3B/14B/A14B), tasks (T2V/I2V), and resolutions (480P/720P), all at 3-step inference with baselines reproduced from official weights (Figures 11–15). SPARKDIFFUSION maintains coherent structure, semantic alignment, and temporal detail even at 97% sparsity.

6 CONCLUSION

We identified the *high-sparsity trap*: at extreme sparsity, step-local training can converge while terminal quality degrades, indicating that supervision is a major bottleneck. This diagnosis yields SPARKDIFFUSION's central contribution: *first adapt the sparse architecture into a coarse prior, then correct the terminal distribution through few-step distillation*. The framework sustains 97% attention sparsity with competitive quality across backbones, tasks, and resolutions. Its reported 265× result is the full-stack outcome of step reduction, sparse attention, and FP8 deployment, not an isolated sparsity speedup. Extending the framework to auto-regressive video diffusion and omni-modal generative models is left as future work.

REPRODUCIBILITY STATEMENT

All training configurations, hyperparameters, optimization settings, and evaluation protocols are detailed in Appendix E. The diagnostic experiments of Section 3 use publicly available datasets and

the Wan2.1/Wan2.2 checkpoints. We report model scales, sparsity levels, batch sizes, learning rates, training steps, GPU hours, and wall-clock training time for both Stage 1 sparse warm-up (Table 7) and Stage 2 trajectory-mixed distillation (Table 8). Inference latency measurements are obtained on NVIDIA H100 and RTX 5090 GPUs with the deployment stack of Section 4.3. Code, model weights, and training scripts will be released upon publication to facilitate reproduction and extension of this work.

ETHICS STATEMENT

This work substantially reduces video-generation inference latency, enabling high-quality text-to-video and image-to-video synthesis on consumer hardware. While this democratizes access to generative video technology for creative and educational applications, it also lowers the barrier for potential misuse, including the creation of deepfakes, synthetic misinformation, and non-consensual content. We acknowledge these dual-use risks and recommend that practitioners deploying SPARKDIFFUSION or similar methods incorporate provenance tracking (e.g., C2PA metadata), visible or invisible watermarking, and content authenticity verification into their inference pipelines. The methods presented here are architectural and algorithmic optimizations applicable to any video diffusion backbone; responsibility for safe deployment lies with system integrators and platform operators. We encourage the research community to continue developing robust detection methods for AI-generated video content alongside generation advances.

AI ASSISTANCE AND DISCLOSURE STATEMENT

We used large language models to assist with literature review, code debugging, and proofreading of manuscript drafts. All experimental results, algorithmic designs, theoretical analysis, and core technical contributions were developed independently by the authors without AI generation.

REFERENCES

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

Pengtao Chen, Xianfang Zeng, Maosen Zhao, Mingzhu Shen, Wei Cheng, Gang Yu, and Tao Chen. Sparse-vdit: Unleashing the power of sparse attention to accelerate video diffusion transformers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 2957–2965, 2026a.

Siyi Chen, Shaowei Liu, Yixuan Jia, Zian Wang, Huan Ling, Qing Qu, and Jun Gao. Data-forcing distillation: Restoring diversity and fidelity in few-step video generation. *arXiv preprint arXiv:2606.18478*, 2026b.

Tongcheng Fang, Hanling Zhang, Ruiqi Xie, Zhuo Han, Xin Tao, Tianchen Zhao, Pengfei Wan, Wenbo Ding, Wanli Ouyang, Xuefei Ning, et al. Salad: Achieve high-sparsity attention via efficient linear attention tuning for video diffusion transformer. *arXiv preprint arXiv:2601.16515*, 2026.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *Advances in Neural Information Processing Systems*, 38:167283–167308, 2026.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. Vbench: Comprehensive benchmark suite for video generative models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21807–21818. IEEE, 2024.

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

Xingyang Li, Muyang Li, Tianle Cai, Haocheng Xi, Shuo Yang, Yujun Lin, Lvmin Zhang, Songlin Yang, Jinbo Hu, Kelly Peng, et al. Radial attention: $\mathcal{O}(n \log n)$ sparse attention with energy decay for long video generation. *Advances in Neural Information Processing Systems*, 38:16822–16852, 2026.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Yuxi Liu, Yipeng Hu, Zekun Zhang, Kunze Jiang, and Kun Yuan. Mixture of distributions matters: Dynamic sparse attention for efficient video diffusion transformers. *arXiv preprint arXiv:2601.11641*, 2026a.

Yuxi Liu, Haoyu Li, Yixiang Cai, Tengxu Sun, Zekun Zhang, Baole Ai, Ang Wang, Jiamang Wang, Lin Qu, Kun Yuan, and Kai Zhang. Crossdistill: Balancing quality and diversity via trajectory-level hybrid few-step distillation. *arXiv preprint arXiv:2609.14725*, 2026b.

Yuxi Liu, Zekun Zhang, Yixiang Cai, Renjia Deng, Yutong He, and Kun Yuan. Ropeslr: 3d rope-driven sparse-lowrank attention for efficient diffusion transformers. *arXiv preprint arXiv:2605.20659*, 2026c.

Yihong Luo, Tianyang Hu, Jiacheng Sun, Yujun Cai, and Jing Tang. Learning few-step diffusion models by trajectory distribution matching. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 17719–17728. IEEE, 2025.

Kepan Nan, Rui Xie, Penghao Zhou, Tiehan Fan, Zhenheng Yang, Zhijie Chen, Xiang Li, Jian Yang, and Ying Tai. Openvid-1m: A large-scale high-quality dataset for text-to-video generation. In *International conference on learning representations*, volume 2025, pp. 1045–1064, 2025.

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

Neta Shaul, Chao Liu, Arash Vahdat, and Julius Berner. Parallel decoding distillation for fast image and video generation. *arXiv preprint arXiv:2607.26004*, 2026.

Haihao Shen, Naveen Mellempudi, Xin He, Qun Gao, Chang Wang, and Mengni Wang. Efficient post-training quantization with fp8 formats. *Proceedings of Machine Learning and Systems*, 6: 483–498, 2024.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023.

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

Fu-Yun Wang, Zhaoyang Huang, Alexander W Bergman, Dazhong Shen, Peng Gao, Michael Lingelbach, Keqiang Sun, Weikang Bian, Guanglu Song, Yu Liu, et al. Phased consistency models. *Advances in neural information processing systems*, 37:83951–84009, 2024.

Limin Wang, Bingkun Huang, Zhiyu Zhao, Zhan Tong, Yinan He, Yi Wang, Yali Wang, and Yu Qiao. Videomae v2: Scaling video masked autoencoders with dual masking, 2023. URL https://arxiv.org/abs/2303.16727.

Jianzong Wu, Liang Hou, Haotian Yang, Ye Tian, Pengfei Wan, Di ZHANG, and Yunhai Tong. Vmoba: Mixture-of-block attention for video diffusion models. In *International Conference on Learning Representations*, volume 2026, pp. 132534–132548, 2026.

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

Yifei Xia, Suhan Ling, Fangcheng Fu, Yujie Wang, Huixia Li, Xuefeng Xiao, and Bin Cui. Training-free and adaptive sparse attention for efficient long video generation. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15982–15993. IEEE, 2025.

Shuo Yang, Haocheng Xi, Yilong Zhao, Muyang Li, Jintao Zhang, Han Cai, Yujun Lin, Xiuyu Li, Chenfeng Xu, Kelly Peng, et al. Sparse videogen2: Accelerate video generation with sparse attention via semantic-aware permutation. *Advances in Neural Information Processing Systems*, 38:96965–96991, 2026.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. In *International Conference on Learning Representations*, volume 2025, pp. 83048–83077, 2025.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and William T Freeman. Improved distribution matching distillation for fast image synthesis. *Advances in neural information processing systems*, 37:47455–47487, 2024a.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6613–6623. IEEE, 2024b.

Tianwei Yin, Qiang Zhang, Richard Zhang, William T Freeman, Fredo Durand, Eli Shechtman, and Xun Huang. From slow bidirectional to fast autoregressive video diffusion models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22963–22974. IEEE, 2025.

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

Jintao Zhang, Kaiwen Zheng, Kai Jiang, Haoxu Wang, Ion Stoica, Joseph E Gonzalez, Jianfei Chen, and Jun Zhu. Turbodiffusion: Accelerating video diffusion models by 100-200 times. *arXiv preprint arXiv:2512.16093*, 2025b.

Jintao Zhang, Haoxu Wang, Kai Jiang, Shuo Yang, Kaiwen Zheng, Haocheng Xi, Ziteng Wang, Hongzhou Zhu, Min Zhao, Ion Stoica, et al. Sla: Beyond sparsity in diffusion transformers via fine-tunable sparse–linear attention. In *International Conference on Learning Representations*, volume 2026, pp. 138288–138305, 2026a.

Jintao Zhang, Haoxu Wang, Kai Jiang, Kaiwen Zheng, Youhe Jiang, Ion Stoica, Jianfei Chen, Jun Zhu, and Joseph E Gonzalez. Sla2: Sparse-linear attention with learnable routing and qat. *arXiv preprint arXiv:2602.12675*, 2026b.

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

Peiyuan Zhang, Yongqi Chen, Haofeng Huang, Will Lin, Zhengzhong Liu, Ion Stoica, Eric Xing, and Hao Zhang. Faster video diffusion with trainable sparse attention. *Advances in Neural Information Processing Systems*, 38:152509–152534, 2026c.

Zekun Zhang, Yixiang Cai, Yuxi Liu, Tengxu Sun, Tianle Liu, Zhoutong Wu, Haoyu Li, Baole Ai, Ang Wang, Jiamang Wang, Lin Qu, and Kun Yuan. Rola: Rotary-positioned low-rank linear attention for efficient diffusion transformers, 2026d. URL https://arxiv.org/abs/2609.06712.

Dian Zheng, Ziqi Huang, Hongbo Liu, Kai Zou, Yinan He, Fan Zhang, Lulu Gu, Yuanhan Zhang, Jingwen He, Wei-Shi Zheng, et al. Vbench-2.0: Advancing video generation benchmark suite for intrinsic faithfulness. *arXiv preprint arXiv:2503.21755*, 2025.

Kaiwen Zheng, Yuji Wang, Qianli Ma, Huayu Chen, Jintao Zhang, Yogesh Balaji, Jianfei Chen, Ming-Yu Liu, Jun Zhu, and Qinsheng Zhang. Large scale diffusion distillation via score-regularized continuous-time consistency. In *International Conference on Learning Representations*, volume 2026, pp. 2582–2603, 2026.

A Related Work

High-sparsity attention for video diffusion. Sparse attention is a widely used route for reducing the quadratic attention cost of video DiTs. Training-free methods exploit recurring spatiotemporal structure, search structured sparse patterns, or construct sparse masks dynamically during inference (Xi et al., 2025; Yang et al., 2026; Chen et al., 2026a; Liu et al., 2026a; Zhang et al., 2025a; Li et al., 2026; Zhang et al., 2025c; Xia et al., 2025), while trainable approaches learn adaptive sparse computation from data (Zhang et al., 2026c; Wu et al., 2026). To retain global context under aggressive sparsification, compensation-based methods such as SLA (Zhang et al., 2026a), SLA2 (Zhang et al., 2026b), and SALAD (Fang et al., 2026) combine sparse attention with lightweight linear branches. RoPeSLR (Liu et al., 2026c) further identifies a 3D-RoPE-aware sparse-plus-low-rank structure in video attention, where high-energy semantic interactions are captured by a sparse branch and the remaining background context is modeled by a low-rank MLP. These methods motivate the architectural feasibility of high sparsity, but they primarily focus on sparse-plus-compensation designs and do not directly address the terminal-quality failure that can appear when sparsity is pushed to the extreme regime studied in this paper.

Sparse attention and few-step distillation. Few-step diffusion generation has been widely explored through distribution-matching and consistency-based approaches (Yin et al., 2024b;a; Song et al., 2023; Wang et al., 2024; Luo et al., 2025; Zheng et al., 2026). Recent works combine sparse attention with few-step distillation to reduce both per-step computation and sampling steps. VSA (Zhang et al., 2026c) jointly trains sparse attention with DMD-style distillation, but DMD-style objectives are known to reduce sample diversity due to mode-seeking behavior (Chen et al., 2026b). TurboDiffusion (Zhang et al., 2025b) trains its sparse-attention (SLA)Zhang et al. (2026a) and consistency (rCM)Zheng et al. (2026) components separately and then merges the parameter updates. SparkDiffusion instead takes a staged route motivated by the diagnosis in Sec.3: sparse architecture adaptation and terminal alignment address two distinct failure modes—the former produces a usable generative prior under extreme sparsity, the latter corrects the terminal distribution, so we decouple them into two sequential phases, each with a single standard objective.

B Additional Results

B.1 Additional Qualitative Results

This appendix reports additional qualitative results across model scales, tasks, resolutions, and attention-sparsity levels. Full Attention is evaluated with 50 sampling steps and classifier-free guidance (NFE $= 100$); all accelerated models are evaluated with three CFG-free inference steps (NFE=3). Where included, TurboDiffusion and FastWan are reproduced from their officially released weights and inference scripts. The same prompts, random seeds, and displayed temporal positions are used across all comparisons.



Figure 11: **Wan2.1-T2V-1.3B-480P.** Qualitative results at 480×832 resolution with 81 frames; TurboDiffusion and FastWan and SparkDiffusion operate at 90% attention sparsity.

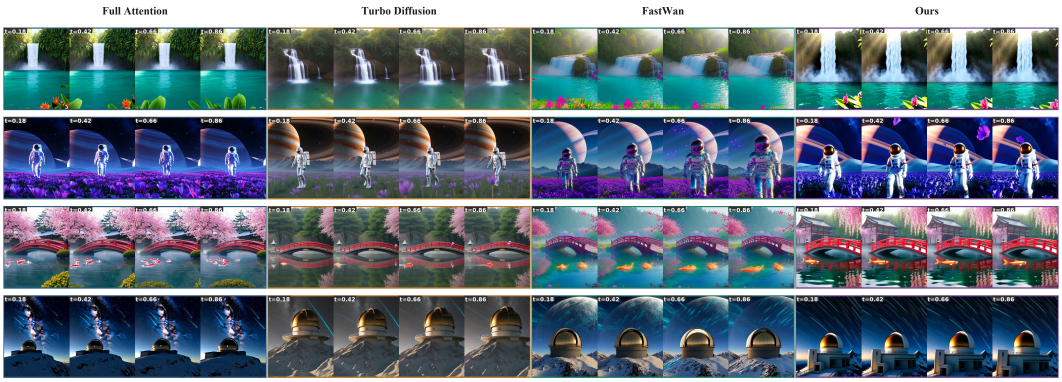

Figure 12: **Wan2.1-T2V-14B-480P.** Qualitative results at 480×832 resolution with 81 frames; all methods operate at 90% attention sparsity.

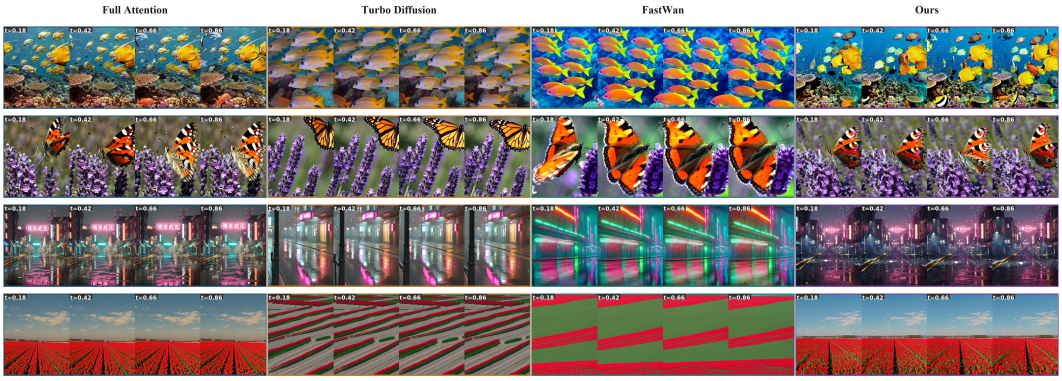

Figure 13: **Wan2.1-T2V-14B-720P.** Qualitative results at 720×1280 resolution with 81 frames; TurboDiffusion and FastWan operate at 90% attention sparsity, SPARKDIFFUSION at 97%.

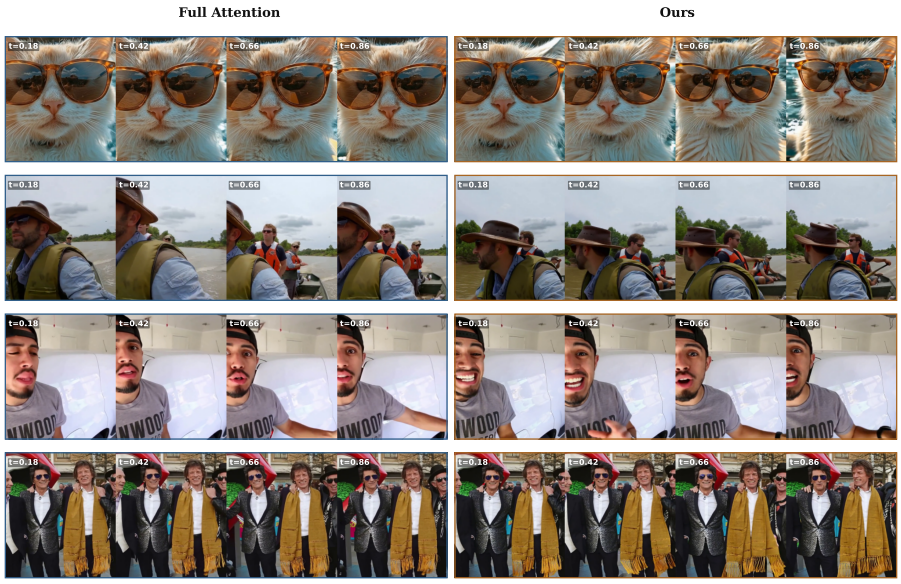

Figure 14: **Wan2.1-I2V-14B-720P.** Qualitative results at 720×1280 resolution with 81 frames; SPARKDIFFUSION operates at 97% attention sparsity.

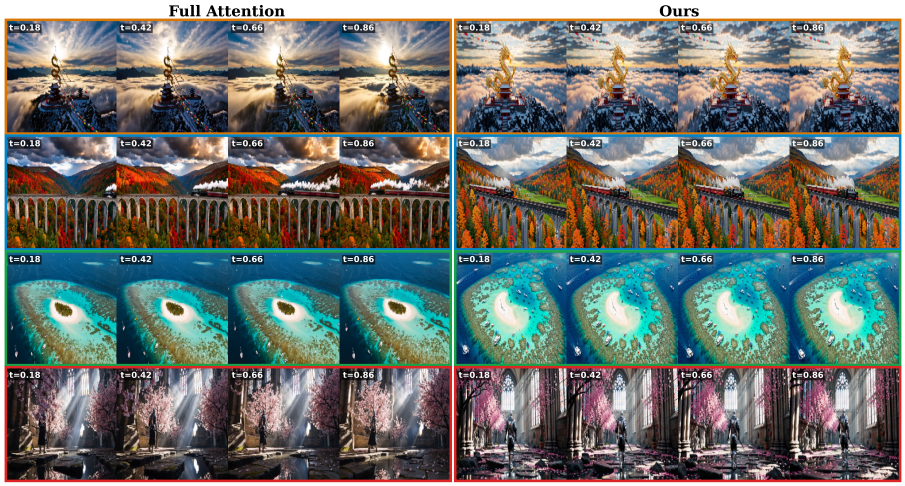

Figure 15: **Wan2.2-T2V-A14B-720P.** Qualitative results at 720 × 1280 resolution with 81 frames; SPARKDIFFUSION operates at 97% attention sparsity.

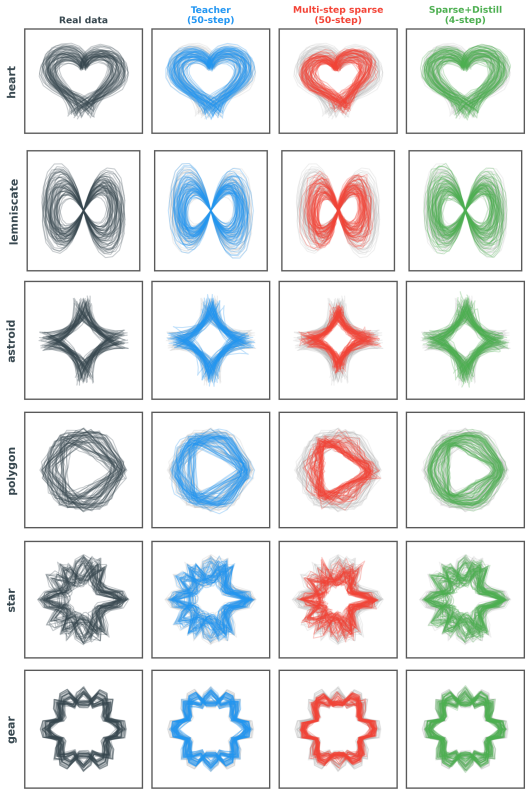

Figure 16: Qualitative sample overlays for the same six toy sequence manifolds. In this toy setting, the multi-step sparse model can drift away from the data distribution, producing distorted or misaligned contours, while the 3-step distilled student produces more globally consistent shapes.

B.2 FP8 VERSUS BF16 QUALITY ABLATION

Table 4 reports the quality impact of the W8A8 FP8 deployment step of Sec. 4.3, comparing each deployed model against its BF16 counterpart under identical 3-step CFG-free inference. Across backbones and sparsity levels, FP8 quantization changes VBench and VBench-2.0 by at most 0.07 points, confirming that with few-step sampling the quantization error has little room to accumulate



Figure 17: Few-step distillation preserves sample diversity. On Wan2.1-T2V-14B-480P, for each of two prompts, one frame per video under three models (dense teacher; sparse model with 90% block sparsity and rank-64 compensation; 3-step student distilled from it) using the same four noise seeds.

and that the speedups in Fig. 2 come at negligible quality cost. All SPARKDIFFUSION quality numbers in the main text are therefore reported directly on the deployed FP8 model.

Table 4: FP8 (W8A8) versus BF16 generation quality on deployed SPARKDIFFUSION models (3-step CFG-free inference).

Model	Sp.	VBench (BF16)	VBench (FP8)	VBench-2.0 (BF16)	VBench-2.0 (FP8)
Wan2.1-T2V-1.3B-480P	90%	82.68	82.64	55.91	55.85
Wan2.1-T2V-14B-720P	90%	83.47	83.42	59.41	59.36
Wan2.1-T2V-14B-720P	97%	83.22	83.15	58.12	58.05
Wan2.2-T2V-A14B-720P	97%	83.41	83.36	58.53	58.46

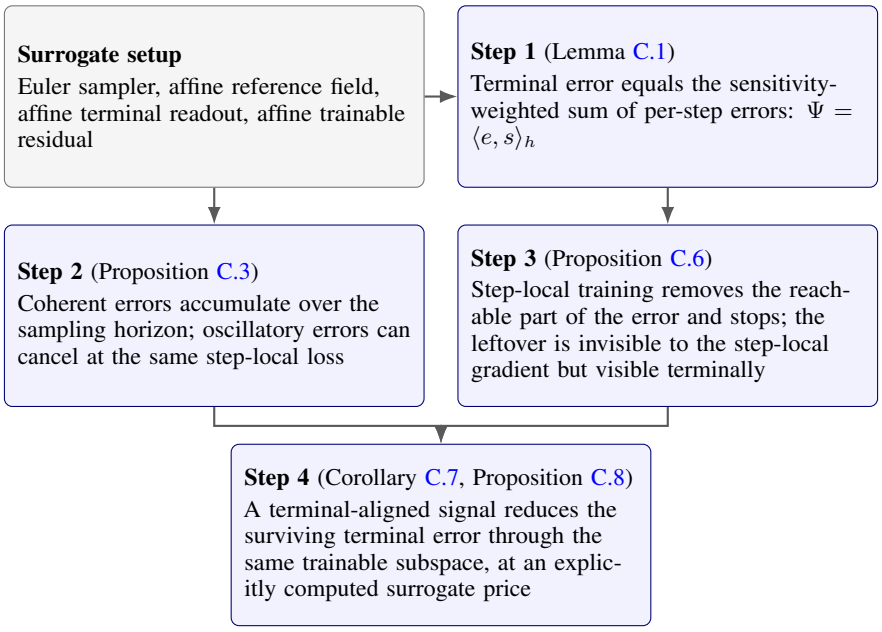

Figure 18: **Roadmap of the mechanism.** Each box states one step of the argument and where it is proved. The proof is Hilbert-space geometry: a quadratic step-local loss and a linear terminal functional see the same residual differently. Sparsity enters only through empirical inputs—the residual left by sparse training and the trainable correction subspace—not as a mathematical object in the proof.

C Idealized Mechanism: Why Step-Local Training Can Leave a Terminal Error

In three sentences. The flow-matching loss sums the *squared magnitudes* of per-step velocity errors, whereas the terminal sample depends on a *signed, transported sum* of the same errors along the sampling trajectory. Therefore two models can have identical step-local training loss but very different terminal outputs: one whose per-step errors point consistently and accumulate, and one whose errors oscillate and cancel. In a deliberately small surrogate, we show that the terminal-visible error left by a step-locally optimal model can still be reduced through the *same* trainable parameters by a terminal-aligned signal, and we compute the surrogate price of doing so.

C.1 Roadmap

Figure 18 summarizes the logical flow of the appendix.
The argument has four steps:

Step 1. The terminal error is exactly the sensitivity-weighted sum of per-step velocity errors.

Step 2. If the per-step errors are coherently aligned with the terminal sensitivity, they accumulate; if they oscillate, they can cancel, while the step-local loss remains unchanged.

Step 3. Step-local training removes exactly the part of the error that lies in the trainable subspace and leaves the orthogonal leftover. At the step-local optimum, the loss gradient vanishes, but the terminal error need not vanish.

Step 4. A terminal-aligned objective can reduce this surviving terminal error using the same trainable subspace. The first-order terminal gain costs only second-order step-local suboptimality.

What sparsity does and does not do here. Sparsity is *not* a mathematical object in this appendix. The analysis takes as input that high-sparsity training leaves a residual that is (i) nonzero, (ii) coherently aligned with the terminal sensitivity rather than zero-mean jitter, and (iii) partly outside the span of the trainable correction directions. We do not prove that aggressive sparsification produces these properties. The conclusion is conditional: *if* high-sparsity training leaves such

Table 5: Notation used in the appendix. The key objects are the per-step velocity error e, the terminal sensitivity s, the trainable correction subspace $\mathcal{U}$, and the two objectives: the step-local loss $\mathcal{L}$ and the terminal error Ψ.

Symbol	Meaning
N, h, T	number of sampler steps, step size, horizon $T = Nh$
$x_k^\star$	reference trajectory generated by $v_\star$
e_k	per-step velocity error at step k
$e = (e_0, \dots, e_{N-1})$	full velocity-error sequence
$R(x) = c^\top x$	linear terminal observable
Ψ	terminal error $R(x_N) - R(x_N^\star)$
s_k	terminal sensitivity transported back to step k
$\langle \cdot, \cdot \rangle_h$	step-weighted inner product $h \sum_k a_k^\top b_k$
$\mathcal{U}$	subspace of residual corrections reachable by training
$P_\mathcal{U}$, $P_{\mathcal{U}^\perp}$	orthogonal projections onto $\mathcal{U}$ and its orthogonal complement
$s_\mathcal{U}$, $s_{\mathcal{U}^\perp}$	reachable and unreachable parts of the terminal sensitivity
ℓ	leftover residual $P_{\mathcal{U}^\perp} e^0$ after step-local training
β	norm of the leftover residual, $\beta = \|\ell\|_h$
μ, ϵ	coherence constant and per-step error floor
$\mathcal{L}$	step-local quadratic loss $\frac{1}{2}\|e\|_h^2$
Δ	step-local sub-optimality relative to the step-local optimum
$\mathcal{C}$	surrogate price of cancelling the terminal error

a residual, *then* step-local training cannot remove its terminal-visible part, whereas a terminal-aligned signal can reduce it at first order through the same trainable parameters.

C.2 Setup and notation

We use a deliberately small surrogate in which every object has a simple geometric meaning. Table 5 summarizes the notation.

Sampler, reference field, and terminal observable. Fix $N \in \mathbb{N}$, step size $h > 0$, horizon $T = Nh$, and times $t_k = kh$. The index k runs along the sampling recursion; if the deployed sampler traverses noise levels in the opposite direction, relabel t_k accordingly—nothing below depends on the direction. An Euler sampler with velocity field v generates

$$x_0^v = z, \qquad x_{k+1}^v = x_k^v + h\, v(x_k^v, t_k).$$

The reference field $v_\star$ generates the reference trajectory $x^\star$ in the same way. We assume $v_\star(\cdot, t)$ is affine, so its one-step Jacobian

$$A_k := I_d + h\, \partial_x v_\star(\cdot, t_k)$$

does not depend on the state. Let

$$\Pi_j := A_{N-1} \cdots A_j, \qquad \Pi_N := I_d.$$

The terminal observable is affine:

$$R(x) = c^\top x, \qquad c \neq 0.$$

For a model field v, the terminal error is

$$\Psi(v) := R(x_N^v) - R(x_N^\star). \tag{2}$$

For a parametrized model we write $\Psi(a) := \Psi(v(a))$.

Velocity error. Let e_k denote the model's velocity error at step k. In the main surrogate, e_k is a per-step bias: it depends on the step index but not on the state. The full residual is

$$e = (e_0, \dots, e_{N-1}), \qquad e_k \in \mathbb{R}^d.$$

We equip residual sequences with the step-weighted inner product

$$\langle a, b \rangle_h := h \sum_{k=0}^{N-1} a_k^\top b_k, \qquad \|a\|_h^2 = h \sum_{k=0}^{N-1} \|a_k\|_2^2. \tag{3}$$

Up to the factor $T/2$, $\frac{1}{2}\|e\|_h^2$ is the usual mean-squared velocity error.

Trainable residual corrections. We model the effect of training as adding trainable directions to a baseline residual. Let e^0 be a baseline residual and let U be a linear map whose columns are trainable correction directions. Write

$$e(a) = e^0 + Ua, \qquad \mathcal{U} := \mathrm{range}(U), \tag{4}$$

where a denotes the coordinates of the trainable correction. As a ranges over all coordinates, Ua ranges over all of $\mathcal{U}$, so the set of reachable residuals is exactly $e^0 + \mathcal{U}$. Both the step-local stage and the terminal-aligned stage move the same parameters, so they act through the same subspace $\mathcal{U}$. Let $P_{\mathcal{U}}$ and $P_{\mathcal{U}^\perp}$ denote orthogonal projections onto $\mathcal{U}$ and $\mathcal{U}^\perp$ with respect to $\langle\cdot,\cdot\rangle_h$.

Terminal sensitivity. Define the terminal sensitivity vectors

$$s_k := \Pi_{k+1}^\top c, \qquad s = (s_0, \dots, s_{N-1}), \tag{5}$$

and split them into a reachable and an unreachable part,

$$s_{\mathcal{U}} := P_{\mathcal{U}} s, \qquad s_{\mathcal{U}^\perp} := P_{\mathcal{U}^\perp} s, \qquad s = s_{\mathcal{U}} + s_{\mathcal{U}^\perp}. \tag{6}$$

The vector s_k measures how strongly the terminal readout $R(x_N)$ reacts to a small velocity perturbation injected at step k. It is determined by the sampler and the reference field; it does not depend on the model parameters.

Two objectives. The step-local loss is the quadratic

$$\mathcal{L}(a) := \frac{1}{2}\|e(a)\|_h^2. \tag{7}$$

The terminal error, viewed as a functional of the residual, is the signed, sensitivity-weighted sum

$$\Psi(a) := \langle e(a), s\rangle_h, \tag{8}$$

which Lemma C.1 shows to coincide with (2). The key contrast is:

$$\mathcal{L} \text{ sees only } \|e_k\|_2^2, \qquad \Psi \text{ sees the signed sum } h\sum_k s_k^\top e_k.$$

Useful expansion. For any perturbation δa,

$$\mathcal{L}(a+\delta a) = \mathcal{L}(a) + \langle e(a), U\delta a\rangle_h + \frac{1}{2}\|U\delta a\|_h^2, \tag{9}$$

and

$$\Psi(a+\delta a) - \Psi(a) = \langle U\delta a, s\rangle_h. \tag{10}$$

C.3 Step 1: Terminal error is the sensitivity-weighted sum of per-step errors

Lemma C.1 (Pairing identity)**.** *Under the affine surrogate, for every parameter a,*

$$\Psi(a) = h\sum_{k=0}^{N-1} s_k^\top e_k(a) = \langle e(a), s\rangle_h. \tag{11}$$

Proof. Let

$$z_k := x_k^{v(a)} - x_k^\star$$

be the state deviation at step k, with $z_0 = 0$. Subtracting the reference Euler update from the model Euler update gives

$$z_{k+1} = z_k + h\big[v(a)(x_k^{v(a)}, t_k) - v_\star(x_k^\star, t_k)\big].$$

Because $v_\star$ is affine,

$$v_\star(x_k^{v(a)}, t_k) - v_\star(x_k^\star, t_k) = \partial_x v_\star(\cdot, t_k) z_k.$$

By definition of the residual $e_k(a)$,

$$v(a)(x_k^{v(a)}, t_k) - v_\star(x_k^\star, t_k) = \partial_x v_\star(\cdot, t_k) z_k + e_k(a).$$

Therefore

$$z_{k+1} = A_k z_k + h e_k(a).$$

Unrolling this recursion with $z_0 = 0$ yields

$$z_N = h \sum_{k=0}^{N-1} \Pi_{k+1} e_k(a).$$

Since $R(x) = c^\top x$,

$$\Psi(a) = c^\top z_N = h \sum_{k=0}^{N-1} c^\top \Pi_{k+1} e_k(a) = h \sum_{k=0}^{N-1} (\Pi_{k+1}^\top c)^\top e_k(a) = h \sum_{k=0}^{N-1} s_k^\top e_k(a).$$

This is exactly $\langle e(a), s \rangle_h$. □

Intuition. The terminal error is not the sum of error magnitudes. It is a signed sum: each per-step error e_k is weighted by how sensitive the final output is to a perturbation at that step. The step-local loss discards this sign and transport information; it only sees $\|e_k\|_2^2$.

C.4 Step 2: Coherent errors accumulate; oscillatory errors cancel

By Lemma C.1, the terminal effect of a residual is the single number

$$\Psi = \langle e, s \rangle_h.$$

Whether the N per-step contributions add or cancel depends on their alignment with s.

Definition C.2 (Coherent alignment)**.** Let $\epsilon > 0$ and $\mu \in (0, 1]$. We say that a residual e is (μ, ϵ)-*coherent* with respect to the terminal sensitivity s if

$$\langle e, s \rangle_h \geq \mu \sum_{k=0}^{N-1} h \, \|s_k\|_2 \, \|e_k\|_2, \qquad \|e_k\|_2 \geq \epsilon \quad \text{for every } k. \tag{12}$$

The first inequality says that the per-step errors are not random jitter: they have a systematic component aligned with the direction that the terminal readout is sensitive to. The second inequality says that the error is not concentrated on only a few steps.

Proposition C.3 (Accumulation versus cancellation)**.** *Let*

$$s_{\min} := \min_k \|s_k\|_2, \qquad s_{\max} := \max_k \|s_k\|_2, \qquad e_{\max} := \max_k \|e_k\|_2.$$

Then the terminal error $\Psi = \langle e, s \rangle_h$ satisfies the following.

(a) ***Upper bound.*** *For every residual e,*

$$|\Psi| \leq \sum_{k=0}^{N-1} h \, \|s_k\|_2 \, \|e_k\|_2 \leq s_{\max} \, e_{\max} \, T. \tag{13}$$

(b) ***Coherent errors accumulate.*** *If e is (μ, ϵ)-coherent, then*

$$\Psi \geq \mu \, s_{\min} \, \epsilon \, T. \tag{14}$$

(c) ***Magnitude-only losses cannot distinguish accumulation from cancellation.*** *There exist residuals with the same per-step magnitudes, hence the same step-local loss, but different terminal errors. In the simplest constant-sensitivity case $s_k \equiv s_0$, alternating signs can make the terminal error vanish while keeping all $\|e_k\|_2$ fixed.*

Proof. (a) By the triangle inequality and Cauchy–Schwarz at each step,

$$|\Psi| = \left| h \sum_{k=0}^{N-1} s_k^\top e_k \right| \leq h \sum_{k=0}^{N-1} \|s_k\|_2 \, \|e_k\|_2.$$

Using $\|s_k\|_2 \leq s_{\max}$, $\|e_k\|_2 \leq e_{\max}$, and $hN = T$ gives

$$|\Psi| \leq h \sum_{k=0}^{N-1} s_{\max} e_{\max} = s_{\max} e_{\max} T.$$

(b) By coherence,

$$\Psi = \langle e, s \rangle_h \geq \mu \sum_{k=0}^{N-1} h \, \|s_k\|_2 \, \|e_k\|_2.$$

Since $\|s_k\|_2 \geq s_{\min}$ and $\|e_k\|_2 \geq \epsilon$,

$$\Psi \geq \mu \sum_{k=0}^{N-1} h \, s_{\min} \epsilon = \mu \, s_{\min} \epsilon \, T.$$

(c) Take the simplest case $d = 1$, $s_k \equiv 1$, and N even. Let $e_k = \epsilon$ for even k and $e_k = -\epsilon$ for odd k. Then every step has the same magnitude $\|e_k\|_2 = \epsilon$, so the step-local loss is the same as for the constant residual $e_k \equiv \epsilon$. But

$$h \sum_{k=0}^{N-1} e_k = 0,$$

whereas the constant residual gives terminal error $T\epsilon$. Thus a loss that depends only on $\|e_k\|_2^2$ cannot distinguish the two cases. □

Intuition. The step-local loss sees only the size of each per-step error. It does not see whether those errors point in the same direction and accumulate, or alternate and cancel. For a highly sparse model, the concern is not merely that the residual error is nonzero, but that it may be *coherent*: systematically aligned with the terminal sensitivity. Such an error grows with the sampling horizon, while an oscillatory error of the same magnitude may cancel.

C.5 Step 3: Step-local training removes the reachable part and stops

We now analyze what step-local training can and cannot remove.
Decompose the baseline residual into a trainable part and an orthogonal leftover:

$$e^0 = P_{\mathcal{U}} e^0 + P_{\mathcal{U}^\perp} e^0.$$

Define the leftover residual

$$\ell := P_{\mathcal{U}^\perp} e^0, \qquad \beta := \|\ell\|_h. \tag{15}$$

If $\beta > 0$, write $\hat{\ell} := \ell / \beta$.

Lemma C.4 (Step-local optimum leaves the orthogonal leftover)**.** *There exists a parameter a_{sp} such that*

$$U a_{\mathrm{sp}} = -P_{\mathcal{U}} e^0.$$

For every such a_{sp},

$$e(a_{\mathrm{sp}}) = \ell.$$

Moreover, for any a, writing $g := U(a - a_{\mathrm{sp}}) \in \mathcal{U}$, one has

$$\mathcal{L}(a) = \frac{1}{2}\|\ell\|_h^2 + \frac{1}{2}\|g\|_h^2. \tag{16}$$

Hence every step-local minimizer has the same residual $e(a) = \ell$, and the minimum step-local loss is

$$\min_a \mathcal{L}(a) = \frac{1}{2}\|\ell\|_h^2.$$

Proof. Since $P_{\mathcal{U}} e^0 \in \mathcal{U} = \mathrm{range}(U)$, there exists a_{sp} with $U a_{\mathrm{sp}} = -P_{\mathcal{U}} e^0$. Then

$$e(a_{\mathrm{sp}}) = e^0 + U a_{\mathrm{sp}} = e^0 - P_{\mathcal{U}} e^0 = P_{\mathcal{U}^\perp} e^0 = \ell.$$

For any a, let $g = U(a - a_{\mathrm{sp}})$. Then

$$e(a) = e(a_{\mathrm{sp}}) + g = \ell + g,$$

with $\ell \in \mathcal{U}^\perp$ and $g \in \mathcal{U}$. By orthogonality,

$$\|e(a)\|_h^2 = \|\ell\|_h^2 + \|g\|_h^2,$$

which proves (16). Thus $\mathcal{L}(a)$ is minimized exactly when $g = 0$, i.e. when $e(a) = \ell$. □

Intuition. Step-local training removes exactly the part of the residual that lies in the trainable subspace $\mathcal{U}$. It cannot remove the orthogonal leftover ℓ. At the step-local optimum, the loss gradient vanishes: every feasible parameter direction only increases the loss. But this does not imply that the terminal error vanishes.
We now state the two conditions needed for the leftover to matter.

Assumption C.5 (Visibility and controllability)**.** We assume:
(A) **Visibility.** The leftover is terminally visible:

$$\beta > 0, \qquad \langle \hat{\ell}, s \rangle_h \geq c_0 > 0.$$

If $\langle \hat{\ell}, s \rangle_h < 0$, replace R by $-R$ once; the case $\langle \hat{\ell}, s \rangle_h = 0$ is excluded.
(B) **Controllability.** The terminal sensitivity has a reachable component: $s_{\mathcal{U}} = P_{\mathcal{U}} s \neq 0$.

Visibility says that the leftover contributes to the terminal readout rather than canceling inside it. Controllability says that the trainable directions can influence the terminal readout at all. If $s_{\mathcal{U}} = 0$, then no parameter update in $\mathcal{U}$ can change the terminal functional.

Proposition C.6 (The terminal error survives the step-local optimum)**.** *Under Assumption C.5, at any step-local minimizer a_{sp},*

$$\Psi_{\mathrm{sp}} := \Psi(a_{\mathrm{sp}}) = \langle \ell, s \rangle_h = \langle \ell, s_{\mathcal{U}^\perp} \rangle_h = \beta \, \langle \hat{\ell}, s \rangle_h \geq c_0 \beta > 0. \tag{17}$$

Moreover:

- *the step-local loss has zero first-order variation at a_{sp};*
- *the terminal functional has nonzero first-order variation at a_{sp} through the same trainable subspace $\mathcal{U}$.*

Equivalently, the step-local gradient vanishes, but the gradient of $\frac{1}{2}\Psi^2$ need not vanish.

Proof. Since $e(a_{\mathrm{sp}}) = \ell$, Lemma C.1 gives $\Psi_{\mathrm{sp}} = \langle \ell, s \rangle_h$. Because $\ell \in \mathcal{U}^\perp$ and $s_{\mathcal{U}} \in \mathcal{U}$, the split (6) gives $\langle \ell, s_{\mathcal{U}} \rangle_h = 0$, hence $\Psi_{\mathrm{sp}} = \langle \ell, s_{\mathcal{U}^\perp} \rangle_h$. Using $\ell = \beta \hat{\ell}$ gives $\Psi_{\mathrm{sp}} = \beta \langle \hat{\ell}, s \rangle_h \geq c_0 \beta > 0$.
For the loss, take any coordinate perturbation δa and let $g = U \delta a \in \mathcal{U}$. By (9), the first-order change of $\mathcal{L}$ at a_{sp} is $\langle \ell, g \rangle_h = 0$, because $\ell \perp \mathcal{U}$. Thus the step-local loss is stationary.
For the terminal functional, by (10),

$$\Psi(a_{\mathrm{sp}} + \delta a) - \Psi(a_{\mathrm{sp}}) = \langle U \delta a, s \rangle_h.$$

Since $s_{\mathcal{U}} \neq 0$ and $s_{\mathcal{U}} \in \mathcal{U}$, there exists a perturbation δa with $U \delta a = s_{\mathcal{U}}$. For this perturbation,

$$\langle U \delta a, s \rangle_h = \langle s_{\mathcal{U}}, s \rangle_h = \| s_{\mathcal{U}} \|_h^2 > 0.$$

Hence the terminal functional has a nonzero first-order direction, while the step-local loss does not. □

Intuition. At the step-local optimum, the model has already removed every error component that the trainable subspace can express. The remaining component ℓ is invisible to further step-local training: the loss gradient is zero. But if ℓ is aligned with the terminal sensitivity, it still produces a nonzero terminal error. At the same time, the terminal functional can still be changed by moving in the same trainable subspace. This mismatch—zero step-local gradient but nonzero terminal gradient—is the core of the high-sparsity trap in the surrogate.

C.6 Step 4: Terminal-aligned correction and its surrogate price

We now ask: if the step-local optimum leaves a terminal-visible residual, how much step-local sub-optimality must be paid to cancel it?
Let a_{sp} be a step-local minimizer and define the step-local sub-optimality

$$\Delta(a) := \mathcal{L}(a) - \mathcal{L}(a_{\mathrm{sp}}). \tag{18}$$

By Lemma C.4, if $g := U(a - a_{\mathrm{sp}})$, then

$$\Delta(a) = \frac{1}{2} \| g \|_h^2, \tag{19}$$

and as a ranges over all parameters, g ranges over all of $\mathcal{U}$. Also, since $g \in \mathcal{U}$,

$$\Psi(a) = \Psi_{\mathrm{sp}} + \langle g, s \rangle_h = \Psi_{\mathrm{sp}} + \langle g, s_{\mathcal{U}} \rangle_h. \tag{20}$$

Corollary C.7 (Surrogate price of terminal cancellation)**.** *Under Assumption C.5, the minimum step-local sub-optimality required to make the terminal functional vanish is*

$$\mathcal{C} := \min\{\Delta(a) : \Psi(a) = 0\} = \frac{\Psi_{\mathrm{sp}}^2}{2\,\|s_{\mathcal{U}}\|_h^2} > 0. \tag{21}$$

The minimum is attained exactly at those a with $U(a - a_{\mathrm{sp}}) = g^\star$, where

$$g^\star = -\frac{\Psi_{\mathrm{sp}}}{\|s_{\mathcal{U}}\|_h^2}\, s_{\mathcal{U}}. \tag{22}$$

Moreover, for every a,

$$|\Psi(a) - \Psi_{\mathrm{sp}}| \le \|s_{\mathcal{U}}\|_h \sqrt{2\Delta(a)}. \tag{23}$$

Proof. By (20), the constraint $\Psi(a) = 0$ is

$$\langle g, s_{\mathcal{U}}\rangle_h = -\Psi_{\mathrm{sp}}.$$

By Cauchy–Schwarz,

$$|\Psi_{\mathrm{sp}}| = |\langle g, s_{\mathcal{U}}\rangle_h| \le \|g\|_h\, \|s_{\mathcal{U}}\|_h,$$

so any feasible g satisfies $\|g\|_h^2 \ge \Psi_{\mathrm{sp}}^2/\|s_{\mathcal{U}}\|_h^2$. Using $\Delta = \frac{1}{2}\|g\|_h^2$ gives

$$\Delta \ge \frac{\Psi_{\mathrm{sp}}^2}{2\,\|s_{\mathcal{U}}\|_h^2}.$$

Equality holds if and only if g is parallel to $s_{\mathcal{U}}$ and satisfies the constraint, i.e. $g = g^\star$; such g is feasible because $s_{\mathcal{U}} \in \mathcal{U}$. The corresponding step-local sub-optimality is

$$\mathcal{C} = \frac{1}{2}\|g^\star\|_h^2 = \frac{\Psi_{\mathrm{sp}}^2}{2\,\|s_{\mathcal{U}}\|_h^2}.$$

Finally, for any a, using (20) and Cauchy–Schwarz,

$$|\Psi(a) - \Psi_{\mathrm{sp}}| = |\langle g, s_{\mathcal{U}}\rangle_h| \le \|g\|_h\, \|s_{\mathcal{U}}\|_h = \|s_{\mathcal{U}}\|_h \sqrt{2\Delta(a)},$$

which proves (23). □

Intuition. The terminal constraint $\Psi(a) = 0$ is a single linear constraint on the reachable residual correction $g \in \mathcal{U}$. Among all corrections satisfying this constraint, the cheapest one in step-local loss is the one aligned with the reachable terminal sensitivity $s_{\mathcal{U}}$. Its cost is exactly the squared ratio of the surviving terminal error to the amount of terminal sensitivity that training can reach.

Proposition C.8 (Terminal descent and the trade-off frontier)**.** *Under Assumption C.5, fix a step-local minimizer a_{sp} and let $\mathcal{C}$ and $g^\star$ be as in Corollary C.7.*

(1) ***Terminal descent exists where the step-local loss is flat.*** *At a_{sp}, the step-local loss has zero first-order variation. However, the terminal squared error $J(a) := \frac{1}{2}\Psi(a)^2$ has a nonzero first-order descent direction through the same trainable subspace $\mathcal{U}$.*

(2) ***Trade-off frontier.*** *For every step-local budget $B \ge 0$,*

$$\min\{|\Psi(a)| : \Delta(a) \le B\} = \left(\Psi_{\mathrm{sp}} - \|s_{\mathcal{U}}\|_h \sqrt{2B}\right)_+. \tag{24}$$

In particular, the terminal error decreases at first order in the parameter displacement, while the step-local loss increases only at second order.

(3) ***Optimal path.*** *For $t \in [0, 1]$, let $a(t)$ be any parameter satisfying $U(a(t) - a_{\mathrm{sp}}) = t\, g^\star$. Then*

$$\Psi(a(t)) = (1 - t)\Psi_{\mathrm{sp}}, \qquad \Delta(a(t)) = t^2\mathcal{C}, \qquad \|e(a(t))\|_h^2 = \beta^2 + 2t^2\mathcal{C}. \tag{25}$$

Thus the terminal error decreases linearly along this path, while the step-local cost increases quadratically.

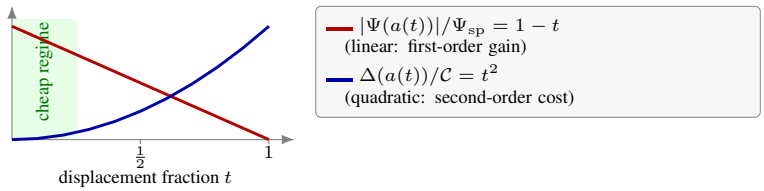

Figure 19: **The exchange rate inside the surrogate** (Corollary C.7, Proposition C.8). Along the optimal path the terminal error falls *linearly* in the displacement fraction t while the step-local loss rises only *quadratically*; the residual norm $\|e(a(t))\|_h^2 = \beta^2 + 2t^2\mathcal{C}$ follows the same quadratic. This is the formal reason for a *staged* recipe rather than a single joint objective. Both axes are surrogate quantities: neither is FID, sliced-W_2, or wall-clock training cost.

Proof. (1) By Lemma C.4, the first-order variation of $\mathcal{L}$ at a_{sp} vanishes because the residual ℓ is orthogonal to every reachable correction $g \in \mathcal{U}$. For the terminal functional, choose a perturbation δa such that $U\delta a = -s_{\mathcal{U}}$; such a perturbation exists because $s_{\mathcal{U}} \in \mathcal{U}$. Then, using (20),

$$\frac{d}{dt}\Big|_{t=0} \Psi(a_{\mathrm{sp}} + t\,\delta a) = \langle -s_{\mathcal{U}}, s\rangle_h = -\|s_{\mathcal{U}}\|_h^2 < 0.$$

Since $\Psi_{\mathrm{sp}} > 0$, the derivative of $J = \frac{1}{2}\Psi^2$ along δa is $\Psi_{\mathrm{sp}} \cdot (-\|s_{\mathcal{U}}\|_h^2) < 0$. Hence J has a first-order descent direction, while $\mathcal{L}$ does not.
(2) For any a with $\Delta(a) \le B$, write $g = U(a - a_{\mathrm{sp}})$. Then $\frac{1}{2}\|g\|_h^2 \le B$, so $\|g\|_h \le \sqrt{2B}$. By (20) and Cauchy–Schwarz,

$$|\Psi(a) - \Psi_{\mathrm{sp}}| = |\langle g, s_{\mathcal{U}}\rangle_h| \le \|g\|_h\,\|s_{\mathcal{U}}\|_h \le \|s_{\mathcal{U}}\|_h\sqrt{2B},$$

so $|\Psi(a)| \ge \Psi_{\mathrm{sp}} - \|s_{\mathcal{U}}\|_h\sqrt{2B}$, and since $|\Psi(a)| \ge 0$,

$$|\Psi(a)| \ge \Big(\Psi_{\mathrm{sp}} - \|s_{\mathcal{U}}\|_h\sqrt{2B}\Big)_+.$$

For achievability, set

$$g_B = -\min\left\{\sqrt{2B},\ \frac{\Psi_{\mathrm{sp}}}{\|s_{\mathcal{U}}\|_h}\right\}\frac{s_{\mathcal{U}}}{\|s_{\mathcal{U}}\|_h} \in \mathcal{U},$$

and let a_B be any parameter with $U(a_B - a_{\mathrm{sp}}) = g_B$. Then $\Delta(a_B) = \frac{1}{2}\|g_B\|_h^2 \le B$ and, by (20),

$$\Psi(a_B) = \Psi_{\mathrm{sp}} - \min\big\{\|s_{\mathcal{U}}\|_h\sqrt{2B},\ \Psi_{\mathrm{sp}}\big\},$$

which attains (24).
(3) Let $g = tg^\star$. From (20) and (22),

$$\Psi(a(t)) = \Psi_{\mathrm{sp}} + t\langle g^\star, s_{\mathcal{U}}\rangle_h = \Psi_{\mathrm{sp}} - t\Psi_{\mathrm{sp}} = (1-t)\Psi_{\mathrm{sp}}.$$

Also $\Delta(a(t)) = \frac{1}{2}\|tg^\star\|_h^2 = t^2\mathcal{C}$. Finally, since $e(a(t)) = \ell + tg^\star$ with $\ell \in \mathcal{U}^\perp$ and $g^\star \in \mathcal{U}$,

$$\|e(a(t))\|_h^2 = \|\ell\|_h^2 + t^2\|g^\star\|_h^2 = \beta^2 + 2t^2\mathcal{C}.$$

□

Intuition. At the step-local optimum, the step-local loss is locally flat: moving a small amount costs only quadratically in $\mathcal{L}$. But the terminal error changes linearly with the same move. Therefore a terminal-aligned signal can obtain a first-order reduction in terminal error while paying only a second-order step-local price. This is the formal reason for the staged recipe: first fit the step-local objective, then spend a small amount of step-local sub-optimality on terminal alignment.

C.7 Step 5: Cancellation is not removal

The previous step shows that terminal alignment can reduce the terminal error. However, it does not repair the underlying velocity error.

Corollary C.9 (Residual inflation)**.** *Under Assumption C.5, let a satisfy $\Psi(a) = 0$. Writing $g = U(a - a_{\mathrm{sp}})$, one has*

$$\|e(a)\|_h^2 = \beta^2 + \|g\|_h^2 \ge \beta^2 + 2\mathcal{C}, \tag{26}$$

with equality if and only if $g = g^\star$. Along the optimal path $g = tg^\star$, the terminal error decreases monotonically, $|\Psi(a(t))| = (1-t)\Psi_{\mathrm{sp}}$, while the residual norm increases monotonically, $\|e(a(t))\|_h^2 = \beta^2 + 2t^2\mathcal{C}$.

Proof. Since $e(a) = \ell + g$ with $\ell \in \mathcal{U}^\perp$ and $g \in \mathcal{U}$,

$$\|e(a)\|_h^2 = \|\ell\|_h^2 + \|g\|_h^2 = \beta^2 + \|g\|_h^2.$$

Among all $g \in \mathcal{U}$ with $\Psi(a) = 0$, Corollary C.7 shows that the minimum possible $\frac{1}{2}\|g\|_h^2$ is $\mathcal{C}$, attained uniquely by $g = g^\star$. Thus $\|e(a)\|_h^2 \geq \beta^2 + 2\mathcal{C}$. The statements along the optimal path follow directly from (25). □

Intuition. Terminal alignment does not remove the velocity error. It adds a compensating component whose terminal effect cancels the terminal-visible projection of the leftover error. As a result, the per-step residual becomes larger, even though the terminal readout becomes smaller. The correct wording is therefore: *terminal alignment cancels the terminal-visible projection of the error*, not *terminal alignment removes the error*.

C.8 Scope and limitations

Remark C.10 (What is and is not established). The preceding argument is a theorem about the surrogate model. It establishes three mechanism-level statements.

(a) Coherent per-step velocity errors can accumulate into a terminal error, while oscillatory errors of the same magnitude can cancel. A magnitude-based step-local loss cannot distinguish these cases.

(b) At a step-local optimum, the remaining terminal error can be strictly positive even though the step-local gradient vanishes. If the leftover is terminally visible and the terminal sensitivity has a reachable component, then a terminal-aligned signal can reduce this error through the same trainable subspace.

(c) Cancelling the terminal-visible projection of the error costs a computable surrogate step-local price and enlarges the residual norm. It is cancellation, not repair.

The surrogate is deliberately idealized. It is not a theorem about nonlinear sparse video diffusion transformers, optimizer trajectories, or concrete distillation losses. The terminal quantity Ψ is a single scalar functional of the terminal state; it is not FID, sliced-W_2, VBench, or any perceptual metric. No specific distribution-matching or consistency loss is verified to descend Ψ. Therefore the main text should read this appendix as motivation for a terminal-aligned stage, not as a guarantee about any particular generation metric.

C.9 A closed-form illustration

We give a small exact example to make the geometry concrete. Let

$$d = 1, \qquad N = 3, \qquad h = \frac{1}{3}, \qquad T = 1,$$

and take

$$v_\star(x, t) = 3x, \qquad R(x) = x.$$

Then each one-step Jacobian is $A_k = 1 + 3h = 2$, the terminal sensitivities are

$$s = (4, 2, 1),$$

and with the weighted inner product $\langle a, b\rangle_h = \frac{1}{3}\sum_k a_k b_k$,

$$\|s\|_h^2 = \frac{1}{3}(16 + 4 + 1) = 7.$$

One trainable direction. Let the trainable subspace be spanned by $\phi = (1, 1, -2)$, so that

$$\|\phi\|_h^2 = \frac{1}{3}(1 + 1 + 4) = 2.$$

Let the baseline residual be

$$e^0 = \beta\hat{\ell} + \gamma\phi, \qquad \hat{\ell} = (1, 1, 1), \qquad \beta > 0.$$

Since $\|\hat{\ell}\|_h = 1$ and $\langle\hat{\ell}, \phi\rangle_h = 0$, the step-local leftover is $\ell = \beta\hat{\ell}$. The surviving terminal error is

$$\Psi_{\text{sp}} = \langle \ell, s\rangle_h = \frac{\beta}{3}(4 + 2 + 1) = \frac{7}{3}\beta,$$

so the visibility constant is $c_0 = \langle \hat{\ell}, s \rangle_h = \frac{7}{3}$. The reachable part of the terminal sensitivity is

$$s_{\mathcal{U}} = \frac{\langle s, \phi \rangle_h}{\|\phi\|_h^2} \phi = \frac{4/3}{2} \phi = \frac{2}{3} \phi, \qquad \|s_{\mathcal{U}}\|_h^2 = \frac{4}{9} \cdot 2 = \frac{8}{9}.$$

The surrogate price is therefore

$$\mathcal{C} = \frac{\Psi_{\mathrm{sp}}^2}{2\|s_{\mathcal{U}}\|_h^2} = \frac{(7/3)^2 \beta^2}{2 \cdot 8/9} = \frac{49}{16} \beta^2,$$

and the optimal correction is

$$g^\star = -\frac{\Psi_{\mathrm{sp}}}{\|s_{\mathcal{U}}\|_h^2} s_{\mathcal{U}} = -\frac{(7/3)\beta}{8/9} \cdot \frac{2}{3} \phi = -\frac{7}{4} \beta \, \phi.$$

Thus the corrected residual is

$$e^\star = \beta(1,1,1) - \frac{7}{4}\beta(1,1,-2) = \beta \left(-\frac{3}{4}, -\frac{3}{4}, \frac{9}{2} \right),$$

and one checks directly that

$$\langle e^\star, s \rangle_h = \frac{\beta}{3} \left(4 \cdot \left(-\frac{3}{4} \right) + 2 \cdot \left(-\frac{3}{4} \right) + 1 \cdot \frac{9}{2} \right) = 0, \qquad \|e^\star\|_h^2 = \beta^2 + 2\mathcal{C} = \frac{57}{8} \beta^2.$$

The terminal error has been cancelled, but the residual norm has increased. Along the path of Proposition C.8(3), at $t = \frac{1}{2}$ the terminal error is halved for one quarter of the price, $\Delta = \frac{49}{64} \beta^2$.

Two trainable directions. Now add a second trainable direction $\psi = (1, -1, 0)$, so that

$$\mathcal{U} = \mathrm{span}\{\phi, \psi\}, \qquad \mathcal{U}^\perp = \mathrm{span}\{(1,1,1)\},$$

and the leftover is still $\ell = \beta(1,1,1)$, hence $\Psi_{\mathrm{sp}} = \frac{7}{3}\beta$ as before. The reachable part of s is now

$$s_{\mathcal{U}} = s - \langle s, \hat{\ell} \rangle_h \, \hat{\ell} = \left(\frac{5}{3}, -\frac{1}{3}, -\frac{4}{3} \right), \qquad \|s_{\mathcal{U}}\|_h^2 = \frac{14}{9},$$

so the price becomes

$$\mathcal{C} = \frac{(7/3)^2 \beta^2}{2 \cdot 14/9} = \frac{7}{4} \beta^2,$$

and the optimal correction is

$$g^\star = -\frac{\Psi_{\mathrm{sp}}}{\|s_{\mathcal{U}}\|_h^2} s_{\mathcal{U}} = -\frac{(7/3)\beta}{14/9} \, s_{\mathcal{U}} = -\frac{3}{2} \beta \, s_{\mathcal{U}} = \beta \left(-\frac{5}{2}, \frac{1}{2}, 2 \right).$$

The corrected residual is therefore

$$e^\star = \beta(1,1,1) + g^\star = \beta \left(-\frac{3}{2}, \frac{3}{2}, 3 \right), \qquad \|e^\star\|_h^2 = \beta^2 + 2\mathcal{C} = \frac{9}{2} \beta^2.$$

In this two-dimensional case, the constraint $\Psi(a) = 0$ is still one linear equation, but the trainable subspace has dimension two. Therefore there are infinitely many corrections $g \in \mathcal{U}$ that cancel the terminal error. Among them, $g^\star$ is the unique correction with minimum step-local cost. Any additional component in $\mathcal{U}$ orthogonal to $s_{\mathcal{U}}$ leaves the terminal error unchanged but increases the residual norm and the step-local sub-optimality.

Summary of the illustration. This example shows the three core mechanisms in exact arithmetic:

- the step-local optimum leaves a nonzero terminal error Ψ_{sp};
- a terminal-aligned correction through the same trainable subspace can cancel Ψ_{sp};
- the cancellation costs a finite surrogate step-local price $\mathcal{C}$ and enlarges the residual norm.

Thus the appendix gives a closed-form picture of why a terminal-aligned stage can help after step-local sparse training, while also making clear that it cancels the terminal-visible projection of the error rather than removing the underlying velocity error.

D GENERALITY ACROSS TRAINABLE SPARSE DESIGNS

We verify that the high-sparsity trap is not specific to the RoLA selector, and that strong baseline recipes do not survive 97% sparsity. All experiments use the Wan2.1-T2V-14B-480P testbed of Sec. 3: the same dense checkpoint, the same enlarged training set, and the same evaluation protocol (terminal error and VBench at 480×832).

Step-local training across selectors. We instantiate two representative trainable sparse branches—VSA-style block selection (Zhang et al., 2026c) and SLA-style sparse–linear attention (Zhang et al., 2026a)—and train them with the step-local flow-matching loss (1) at 90%/95%/97% sparsity under the extended budget of Sec. 3 (10,000 steps).

Baseline recipe at 97%. We retrain the full FastWan (VSA) recipe (Zhang et al., 2026c)—joint sparse-attention training with DMD-style distillation—at 97% sparsity from the same dense checkpoint, changing only the sparsity level in its official training configuration.

Staged recipe across selectors. We apply the Stage-2 trajectory-mixed distillation of Sec. 4.2 to the step-local VSA- and SLA-style 97% checkpoints without modifying their sparse branches.

Table 6 shows the same trap signatures as Sec. 3 on both selectors: the step-local validation loss plateaus at a low value at every sparsity level, while the terminal error and VBench degrade sharply at 95%–97%. SLA attains a consistently lower validation loss than VSA at matched sparsity, yet its terminal error is only marginally smaller—a lower step-local loss does not translate into better terminal quality. Retraining the full FastWan recipe at 97% degrades sharply relative to its official 90% operating point, while SPARKDIFFUSION at the same 97% sparsity remains close to the dense model. The terminal-aligned Stage-2 stage restores terminal quality on both selectors, confirming that the trap is induced by step-local supervision rather than by any specific sparse design, and that the staged recipe is selector-agnostic.

Table 6: The high-sparsity trap across trainable sparse designs (Wan2.1-T2V-14B-480P). Step-local-only rows are trained from the same dense checkpoint with the flow-matching loss (1) under the extended 10,000-step budget of Sec. 3; terminal error is the paired latent MSE of Sec. 3. "FastWan recipe" retrains the official VSA training pipeline (Zhang et al., 2026c) with only the sparsity level changed to 97%, and is evaluated under its official inference settings; "+ Stage 2" applies our trajectory-mixed distillation to the corresponding 97% checkpoint.

Sparse branch / recipe	Sp.	Val loss ↓	Term. error ↓	VBench ↑
VSA-style, step-local	90%	0.0892	0.1205	81.75
VSA-style, step-local	95%	0.0947	0.1232	80.89
VSA-style, step-local	97%	0.0998	0.1255	79.72
SLA-style, step-local	90%	0.0871	0.1203	81.87
SLA-style, step-local	95%	0.0932	0.1227	80.94
SLA-style, step-local	97%	0.0979	0.1251	79.84
FastWan (VSA) recipe, retrained	97%	0.0996	0.1198	80.23
VSA-style + Stage 2	97%	0.0995	0.1001	82.45
SLA-style + Stage 2	97%	0.0977	0.0987	82.51
RoLA + Stage 2 (SPARKDIFFUSION)	97%	0.0953	0.0921	82.99

E TRAINING DETAILS

This section reports the training configurations of SPARKDIFFUSION for the primary backbone Wan2.1-T2V-14B (stages defined in Sec.4; Stage 2 initializes from the Stage 1 checkpoint), together with hardware, wall-clock time, and GPU hours.

E.1 STAGE 1: SPARSE WARM-UP

Stage 1 starts from the pretrained dense Wan2.1-T2V-14B checkpoint and replaces dense self-attention with the compensated sparse attention of Sec. 4.1, optimized with the native flow-matching objective alone (no auxiliary or layerwise losses).

Table 7 summarizes the data configuration, sparse-attention architecture, optimization hyperparameters, and training cost used in this stage.

Table 7: Training configuration of Stage 1 sparse warm-up on Wan2.1-T2V-14B.

Hyperparameter	Setting
Model & Data	
Backbone	Wan2.1-T2V-14B
Training dataset	OpenVid subset (Nan et al., 2025)
Number of training videos	2,000
Video setting	480×832 (480P), 720×1280 (720P), 81 frames
Sparse Architecture	
Attention sparsity	97%
Sparse block size	64×64
Compensation rank	64
Trainable modules	Full DiT backbone (including our low-rank compensation modules and gating parameters)
Frozen modules	VAE and text encoder
Optimization	
Training objective	Flow matching
Optimizer	AdamW
Learning rate	2×10^{-6} (backbone); 5×10^{-5} (low-rank projections / gate bias); 3×10^{-5} (gate projection)
Weight decay	1×10^{-5} (backbone); 0 (others)
LR schedule	Cosine decay
Training epochs	4 ($\approx$250 steps)
Global batch size	32
Gradient clipping	1.0
Training precision	BF16
Gradient checkpointing	Block-wise activation checkpointing
Random seed	0
Compute	
Hardware	8$\times$ NVIDIA H100 80GB SXM
Parallelism	FSDP full-shard
Wall-clock training time	$\approx$2.9 hours
GPU Hours	$\approx$23 GPU hours

E.2 Stage 2: Trajectory-Mixed Distillation

Stage 2 initializes the student from the Stage 1 checkpoint and freezes the dense Wan2.1-T2V-14B model as the teacher, following the CrossDistill schedule of Sec. 4.2: high-noise PCM-style consistency, low-noise DMD-style distribution matching, and alternating student/critic updates.

Table 8 reports the complete distillation schedule, distribution-matching configuration, optimization hyperparameters, and computational cost.

Table 8: Training configuration of Stage 2 trajectory-mixed distillation on Wan2.1-T2V-14B.

Hyperparameter	Setting
Model setting	
Student backbone	Wan2.1-T2V-14B
Student initialization	Stage 1 sparse warm-up checkpoint
Teacher model	Frozen dense Wan2.1-T2V-14B
Teacher CFG scale	5.0
Training dataset	Teacher-synthesized T2V dataset Zheng et al. (2026)
Video setting	480×832 (480P), 720×1280 (720P), 81 frames
Attention sparsity	97%
Trajectory-Mixed Distillation	
Noise split	CrossDistill crosspoint (0.934)
High-noise objective	PCM-style consistency on the high-noise segment
Low-noise objective	DMD-style distribution matching on the low-noise segment
Student sampling steps	3 (one high-noise PCM step and two low-noise DMD steps; Sec. 4.2)
Fake model	Dense Wan2.1-T2V-14B
Fake update ratio	$10:1$
Student trainable modules	Full DiT backbone (including our low-rank compensation modules and gating parameters)
Optimization	
Optimizer	AdamW
Learning rate	2×10^{-6} (student); 4×10^{-7} (fake)
Weight decay	0.01
Training steps	8,000
Global batch size	16
Training precision	BF16
EMA	Disabled
Compute	
Hardware	$8\times$ NVIDIA H100 80GB
Wall-clock training time	$\approx$130 hours
GPU Hours	$\approx$1 K GPU hours