SWIFTVR: REAL-TIME ONE-STEP GENERATIVE VIDEO RESTORATION

Anonymous authors
Paper under double-blind review

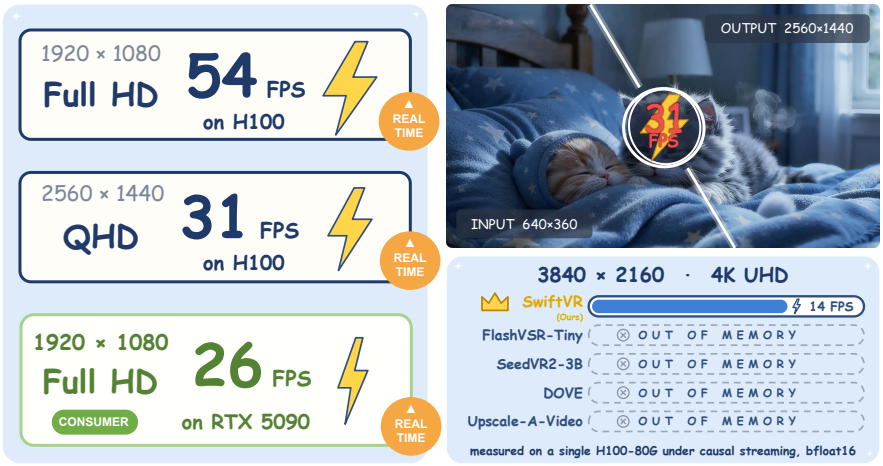


Figure 1: SwiftVR enables real-time streaming video restoration on a single GPU. **Left:** throughput at different resolutions on an H100 and a consumer-grade RTX 5090. **Top right:** a 640×360 input restored to 2560×1440 at 31 FPS. **Bottom right:** at 3840×2160, SwiftVR runs at 14 FPS on a single H100-80G, whereas all compared diffusion-based baselines run out of memory.

ABSTRACT

Real-time video restoration (VR) for live streams requires high-resolution outputs under strict per-frame latency constraints. Existing one-step diffusion-based VR models remain difficult to deploy on consumer-grade GPUs due to two main bottlenecks: quadratic spatial attention at high resolutions and the latency-memory overhead of large video autoencoders. We present SwiftVR, a streaming one-step generative VR framework that reduces both bottlenecks under a causal chunk-wise protocol. For attention, mask-free shifted-window self-attention uses deterministic indexing to gather spatial windows into dense tensors. This design keeps all attention operations on the dense scaled dot-product attention path, eliminating masks, cyclic shifts, padding, and hardware-specific sparse kernels. Consequently, the trained model transfers directly to consumer-grade GPUs without retraining or custom kernels. For autoencoding, a lightweight Restoration-aware Autoencoder enables fast chunk-wise decoding while preserving reconstruction quality. On a single H100, SwiftVR sustains 31 FPS at 2560×1440 and 14 FPS at 3840×2160 (4K), whereas all compared diffusion-based VR baselines exceed the memory limit at 4K. On a consumer-grade RTX 5090 GPU, SwiftVR reaches 26 FPS at 1920×1080 (1080p). To our knowledge, SwiftVR is the first generative VR model to achieve real-time 1080p streaming on a consumer-grade GPU, while attaining strong no-reference perceptual quality with lower inference cost.

1 INTRODUCTION

Live video applications such as video conferencing and game streaming demand ever higher resolution. In practical deployments, however, limited bandwidth (An et al., 2026; Shao & Li, 2025) forces aggressive compression (Chen et al., 2026b;a), which leaves streams with artifacts, noise, and loss of detail. Receiver-side video restoration (VR) can mitigate these degradations, but its use

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.

in live streams calls for real-time processing and sustained display-resolution throughput on a single consumer-grade GPU. In this setting, VR also faces unpredictable, time-varying degradations without full-clip temporal context or multi-pass refinement.

Diffusion-based generative models have become increasingly popular in real-world VR because of their perceptual realism. They have evolved from multi-step samplers (Zhou et al., 2024; He et al., 2024; Xie et al., 2025) to one-step models (Chen et al., 2025b; Wang et al., 2026; Zhuang et al., 2025), which reduce iterative sampling to a single network evaluation. Even one-step models, however, cannot restore high-resolution video in real time on consumer-grade GPUs. Once sampling is removed, the remaining forward pass is dominated by two costs: quadratic self-attention in the DiT, and encoding and decoding in the 3D VAE. FlashVSR (Zhuang et al., 2025) reduces these costs with block-sparse attention and a tiny conditional decoder. Still, it reaches only 17 FPS at 768×1408 on a server-class A100, and its speedup depends on a dedicated sparse attention kernel. Its Wan2.1-1.3B backbone (Wan et al., 2025) also limits scalability. The VAE compresses video by only $(4, 8, 8)$ in time, height, and width. As a result, the token count remains large at high resolutions.

We therefore adopt Wan2.2-TI2V-5B (Wan et al., 2025), whose $(4, 16, 16)$ VAE yields $4\times$ fewer tokens per latent frame. Even so, a single 3840×2160 (4K) forward pass on one H100 takes 6.3 s for VAE encoding, 60.9 s for the DiT, and 25.8 s for VAE decoding, with VAE tiling enabled (Figure 2). The DiT cost comes mainly from self-attention. Let T denote the temporal length and $H \times W$ the spatial size of the latent grid. Self-attention over the resulting $N = THW$ tokens scales as $\mathcal{O}(N^2)$, so for a fixed aspect ratio it grows with the fourth power of the output width. The VAE accounts for most of the remaining time, as encoding and decoding together take more than one-third of the total latency. Consequently, methods that fine-tune Wan2.2-TI2V-5B without changing its architecture, such as LiteVSR (Cao et al., 2026), remain limited in inference efficiency.

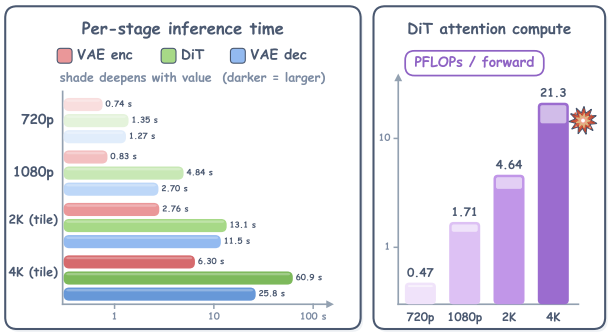


Figure 2: Latency and attention cost of one Wan2.2-TI2V-5B forward pass on an H100 (bfloat16, 25-frame chunk). Left: per-stage inference time, with VAE tiling at 2K and 4K. Right: DiT self-attention compute in PFLOPs.

To make real-time generative restoration practical, we present SwiftVR, a streaming one-step generative VR framework. On a single H100, SwiftVR sustains 31 FPS at 2560×1440 (2K) and 14 FPS at 4K. At 4K, all compared diffusion-based VR baselines exceed the memory limit. On a consumer-grade RTX 5090, SwiftVR reaches 26 FPS at 1920×1080 (1080p; Figure 1). These results come from two designs, each targeting one of the bottlenecks above. For attention, SwiftVR processes streams in causal chunks. Each chunk bounds the temporal length T of the DiT input, so attention cost grows only with the spatial size. This motivates partitioning windows along the spatial axes only. Existing shifted-window designs are ill-suited to this goal. Swin attention (Liu et al., 2021) relies on cyclic shifts and attention masks, and 3D Swin backbones (Wang et al., 2025; 2026) introduce variable-sized boundary windows. Both prevent attention from running as uniformly shaped calls on fused dense kernels. Our mask-free shifted-window self-attention (MFSWA) instead uses deterministic indexing to gather each spatial window into a dense tensor. Every attention call thus stays on the standard scaled dot-product attention (SDPA) fast path. MFSWA improves throughput by $1.62\times$ over the full-attention teacher. Because it needs no custom kernels, the trained model runs on the RTX 5090 without retraining or kernel rewriting. For autoencoding, we replace the heavy 3D VAE with a lightweight Restoration-aware Autoencoder (ReAE), which is jointly fine-tuned with the DiT in pixel space. ReAE decodes $27\times$ faster than Wan2.2-VAE while retaining competitive reconstruction quality.

In summary, our contributions are threefold. **(i)** We present SwiftVR, a streaming one-step VR model that, to our knowledge, is the first generative VR model to achieve real-time 1080p streaming on a consumer-grade GPU. **(ii)** We propose MFSWA and ReAE, which reduce attention and autoencoder costs under a causal chunk-wise streaming protocol. Because MFSWA relies only on standard dense SDPA operations, the trained model is compatible with widely used fused SDPA backends

and requires no hardware-specific kernels. **(iii)** Experiments show that SwiftVR achieves leading no-reference perceptual quality among recent one-step VR methods with lower inference cost. It is also the only evaluated diffusion-based method that scales to 4K on a single GPU, whereas all compared diffusion-based VR baselines exceed the memory limit.

2 Related Work

2.1 Real-world Video Restoration

Video restoration extends image restoration architectures (Wang et al., 2021; Liang et al., 2021; Chen et al., 2023; 2025a) with inter-frame alignment, using motion compensation, deformable convolutions, recurrent propagation, or transformer-based aggregation to exploit temporal redundancy (Chan et al., 2021; 2022a; Wang et al., 2019; Liang et al., 2024; 2022). Early video methods typically assume fixed, known degradations, such as bicubic downsampling, and generalize poorly to real-world videos with spatiotemporally varying compression, noise, and blur. Real-world variants therefore use richer synthetic degradation pipelines and introduce cleaning modules to suppress input artifacts before upsampling (Chan et al., 2022b; Zhang & Yao, 2024; Yang et al., 2021). This prevents residual noise amplification. These methods are efficient and temporally stable, but their regression-oriented objectives optimize pixel-wise errors and bias outputs toward averaged solutions, limiting perceptual realism in heavily degraded regions.

2.2 One-step Diffusion Video Restoration

Despite improving perceptual realism in restoration (Wang et al., 2024a; Lin et al., 2024; Wu et al., 2024b; Yu et al., 2024; Wang et al., 2025), diffusion priors rely on iterative denoising, resulting in substantial computational overhead for high-resolution video streams. Distillation and rectified-flow-based techniques (Luo et al., 2023; Liu et al., 2022; Yin et al., 2024b;a) reduce this overhead by enabling single-step inference and have recently been extended to VR. DOVE (Chen et al., 2025b) fine-tunes a pretrained video diffusion model into a one-step student using a two-stage latent-to-pixel scheme for offline video super-resolution (VSR). SeedVR2 (Wang et al., 2026) performs one-step VR via diffusion adversarial post-training and adopts adaptive window attention, where the window size is resized according to output resolution. FlashVSR (Zhuang et al., 2025) formulates streaming VSR as a sparse-attention problem, combining locality-constrained block-sparse attention with a compact decoder. One-step image restoration (Wang et al., 2024b; Wu et al., 2024a; Tai et al., 2026; Liu et al., 2025) is computationally efficient per frame but lacks temporal compression and modeling, limiting its extension to efficient and consistent VR.

Although these one-step methods substantially reduce sampling cost, they retain bottlenecks that hinder real-time streaming on consumer hardware. These include offline-oriented designs overlooking streaming and autoencoder costs, heavy attention backbones that bottleneck consumer-grade 4K inference, and speedups tied to hardware-specific sparse kernels. FlashVSR (Zhuang et al., 2025) reaches $\sim$17 FPS at 768×1408 on a server-class A100, remaining below real-time speed and 1080p resolution. Real-time 1080p generative VR on consumer-grade GPUs remains unresolved.

2.3 Efficient Attention in Diffusion

Once sampling is reduced to a single step, attention computation in the diffusion transformer becomes dominant, motivating three lines of work. The first is trainable sparse attention (Zhuang et al., 2025; Zhang et al., 2025b; 2026), which achieves high sparsity but relies on dedicated fused sparse kernels for wall-clock speedup. On consumer-grade GPUs without such kernels, sparse arithmetic may not yield measurable acceleration. The second is training-free feature reuse across denoising iterations through caching or forecasting (Ma et al., 2024; Lv et al., 2025; Zou et al., 2025; Li et al., 2024). These methods reduce computation along the sampling axis and therefore provide limited benefit in single-step diffusion inference. Rolling KV caches in causal streaming generators (Huang et al., 2025; Yin et al., 2025) also improve efficiency, but along the temporal axis. By reusing keys and values from prior frames, they avoid recomputing these representations across chunks. However, they do not reduce the per-step spatial attention cost within the current chunk. The third line exploits window-based attention to impose architectural locality, as in SwinIR (Liang et al., 2021)

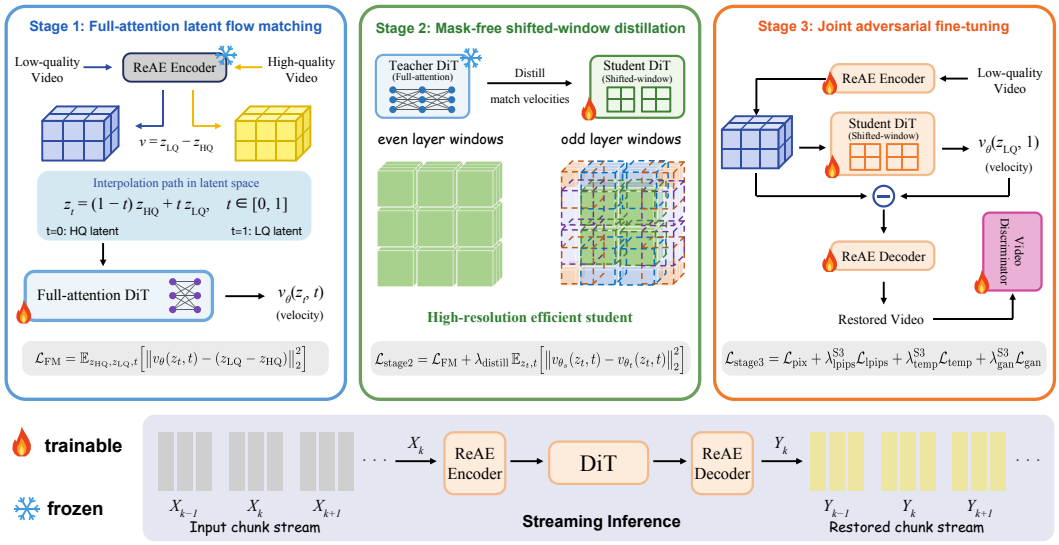


Figure 3: **Overview of SwiftVR.** *Stage 1:* in the frozen ReAE latent space, a full-attention DiT learns the constant velocity $v = z_{\mathrm{LQ}} - z_{\mathrm{HQ}}$ along $z_t = (1-t)z_{\mathrm{HQ}} + tz_{\mathrm{LQ}}$. *Stage 2:* the full-attention teacher is distilled into an MFSWA student. *Stage 3:* the student DiT and ReAE are jointly fine-tuned with a video discriminator under one-step inference. *Bottom:* each input chunk X_k is restored to Y_k with a single DiT pass.

and Uformer (Wang et al., 2022). SeedVR (Wang et al., 2025) and SeedVR2 (Wang et al., 2026) extend this idea to diffusion transformers with 3D shifted windows. To process full sequences at arbitrary resolutions, they use cyclic shifts, attention masks, and variable-sized boundary windows.

Among these directions, window-based locality requires no specialized kernels and remains effective under single-step inference. SwiftVR follows this direction but restricts partitioning to the spatial axes and realizes shifted windows without masks or cyclic shifts (Sec. 3.2).

3 Method

SwiftVR is a streaming, one-step generative VR framework built on a Restoration-aware Autoencoder (ReAE) and a DiT with window-based self-attention (Figure 3). Videos are processed causally in fixed-length chunks, which bounds the temporal length T of each DiT input. Since self-attention scales quadratically with $N = THW$, the attention cost is then governed by the latent spatial size $H \times W$, which motivates spatial-only window partitioning. The DiT is optimized in three stages (Sec. 3.2) and restores each chunk with a single forward pass at inference (Sec. 3.3).

3.1 Restoration-aware Autoencoder

Once sampling is reduced to a single step, the autoencoder becomes a major source of end-to-end latency. The 3D VAE of large video generation backbones (Wan et al., 2025) is too slow for real-time high-resolution decoding and costly to optimize jointly with the DiT. We therefore introduce ReAE, a compact Restoration-aware Autoencoder with encoder E_ϕ and decoder D_ϕ, such that $z = E_\phi(x)$ and $\hat{x} = D_\phi(z)$. ReAE is initialized from a lightweight autoencoder (Boer Bohan, 2025) and fine-tuned on video data.

ReAE is trained independently on clean videos in two phases. The first phase optimizes pixel fidelity, perceptual similarity, and temporal consistency:

$$\mathcal{L}^{(1)}_{\mathrm{ReAE}} = \mathcal{L}_{\mathrm{pix}} + \lambda^{\mathrm{ReAE}}_{\mathrm{lpips}}\mathcal{L}_{\mathrm{lpips}} + \lambda^{\mathrm{ReAE}}_{\mathrm{temp}}\mathcal{L}_{\mathrm{temp}}, \tag{1}$$

where $\mathcal{L}_{\mathrm{pix}} = \|\hat{x} - x\|_1$, $\mathcal{L}_{\mathrm{lpips}}$ measures the perceptual distance between the reconstructed and target frames in the LPIPS feature space (Zhang et al., 2018), and $\mathcal{L}_{\mathrm{temp}}$ denotes the MSE between consecutive frame differences. The second phase adds adversarial supervision after reconstruction

training converges:

$$\mathcal{L}_{\text{ReAE}}^{(2)} = \mathcal{L}_{\text{ReAE}}^{(1)} + \lambda_{\text{gan}}^{\text{ReAE}} \mathcal{L}_{\text{gan}}. \tag{2}$$

ReAE is then frozen in Stages 1 and 2 and fine-tuned jointly with the DiT in Stage 3.

3.2 Progressive DiT Optimization

Stage 1: Full-attention latent flow matching. Following flow matching (Lipman et al., 2023), we formulate restoration as learning a velocity field along a straight path between paired latents. Given a high-quality (HQ) and low-quality (LQ) paired clip $(x_{\text{HQ}}, x_{\text{LQ}})$, let $z_{\text{HQ}} = E_\phi(x_{\text{HQ}})$ and $z_{\text{LQ}} = E_\phi(x_{\text{LQ}})$ denote its latents under the frozen ReAE encoder. The two latents are connected by the linear interpolation path

$$z_t = (1-t)\, z_{\text{HQ}} + t\, z_{\text{LQ}}, \quad t \in [0,1], \tag{3}$$

whose velocity $\mathrm{d}z_t/\mathrm{d}t = z_{\text{LQ}} - z_{\text{HQ}}$ is independent of t. A full-attention DiT v_θ is trained to regress this target velocity:

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{z_{\text{HQ}}, z_{\text{LQ}}, t}\left[\left\| v_\theta(z_t, t) - (z_{\text{LQ}} - z_{\text{HQ}}) \right\|_2^2 \right]. \tag{4}$$

Stage 2: Mask-free shifted-window distillation. Although full attention provides a global receptive field, its cost grows quadratically with the number of latent tokens and dominates the Stage-1 DiT latency at high resolutions. Restricting attention to local windows reduces this cost in wall-clock time only if attention still runs on fused dense SDPA kernels (Dao, 2024; Zhang et al., 2025a); windows realized through attention masks or padding fall back to slower execution paths. We therefore propose mask-free shifted-window self-attention (MFSWA). Its key idea is to specify the shifted-window partition entirely through deterministic gather and scatter indices, so that the attention kernel only receives dense, uniformly shaped blocks. Figure 4 illustrates its three components.

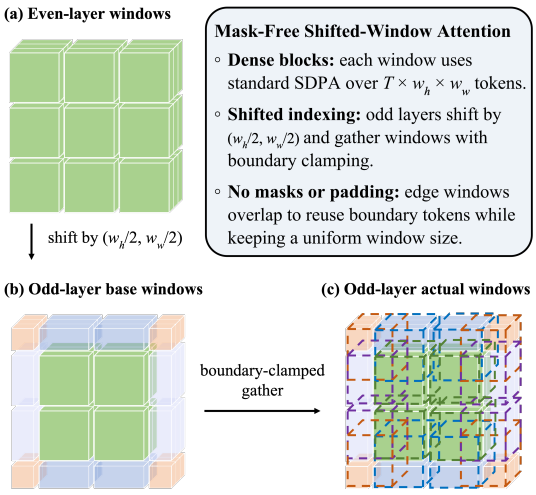


Figure 4: Illustration of MFSWA. Each cube denotes a spatial window spanning all T frames of a chunk. In (b), faded cubes mark partial border windows; in (c), dashed cubes mark the full-size windows actually gathered in odd layers.

Dense blocks. MFSWA partitions only the spatial axes of each latent chunk into windows of size $w_h \times w_w$, with each window extending across all T frames. In every layer, the tokens of each window, either a regular window in even layers (Figure 4(a)) or a boundary-clamped window in odd layers (Figure 4(c)), are gathered into a dense block of Tw_hw_w tokens. Since all blocks share the same shape, they are batched and processed by standard SDPA with `attn_mask=None`, and the outputs are scattered back to their original token positions. Attention is thus restricted to local spatial neighborhoods, while each token retains full temporal visibility within its chunk.

Shifted indexing. To enable information exchange across window boundaries, MFSWA alternates between two spatial partitions. Even layers adopt the regular partition in Figure 4(a), whereas odd layers displace the window anchors by $(w_h/2, w_w/2)$, such that tokens separated by a window boundary in one layer fall within a common window in the subsequent layer. However, this displacement produces incomplete windows along the borders of the latent grid (Figure 4(b)). Rather than completing these windows with padding tokens or excluding invalid positions through attention masks, MFSWA applies boundary clamping during the gather operation. Specifically, a window anchored at row i is assigned the row interval $[i', i' + w_h)$, where $i' = \min(\max(i, 0), H - w_h)$, and the column interval is determined analogously. Each incomplete window is thereby translated inward until it covers a full $w_h \times w_w$ region (Figure 4(c)). Because the same rule applies to any latent size, MFSWA accommodates resolutions that are not divisible by the window size.

No masks or padding. Clamped edge windows overlap their neighbors and reuse boundary tokens, so every gathered block keeps a uniform size at any resolution, and border tokens attend to a complete neighborhood of valid tokens. For tokens covered by multiple windows, the scatter step follows a fixed window priority applied consistently across each overlapped region, so every output token is written exactly once. Shifted-window attention therefore requires no masks, padding, or cyclic shifts. Further implementation details are provided in the supplementary material.

Distillation. The MFSWA student is initialized from the Stage-1 teacher and trained with flow matching and a teacher-distillation term:

$$\mathcal{L}_{\text{stage2}} = \mathcal{L}_{\text{FM}} + \lambda_{\text{distill}}\, \mathbb{E}_{z_t,t}\left[\left\| v_{\theta_s}(z_t,t) - v_{\theta_t}(z_t,t)\right\|_2^2\right], \tag{5}$$

where θ_s and θ_t denote the student and frozen teacher parameters. This term transfers the restoration behavior of the full-attention teacher to the window-restricted student.

Stage 3: Joint adversarial fine-tuning. Stages 1 and 2 supervise only latent velocities, which constrain the decoded video indirectly. We therefore fine-tune the student DiT, denoted v_θ hereafter, jointly with ReAE in pixel space under the one-step inference protocol used at deployment. A single Euler step from $t=1$ recovers the high-quality latent, which is then decoded:

$$\hat{z}_{\text{HQ}} = E_\phi(x_{\text{LQ}}) - v_\theta\big(E_\phi(x_{\text{LQ}}),\, 1\big), \quad \hat{x} = D_\phi(\hat{z}_{\text{HQ}}). \tag{6}$$

The output is supervised by $\mathcal{L}_{\text{stage3}} = \mathcal{L}_{\text{pix}} + \lambda^{\text{S3}}_{\text{lpips}}\mathcal{L}_{\text{lpips}} + \lambda^{\text{S3}}_{\text{temp}}\mathcal{L}_{\text{temp}} + \lambda^{\text{S3}}_{\text{gan}}\mathcal{L}_{\text{gan}}$, where the adversarial term is computed by a video discriminator that pairs a frozen, pretrained VGG-19 feature extractor (Simonyan & Zisserman, 2015) with lightweight trainable heads.

3.3 Streaming Inference

We adopt a causal chunk-wise streaming protocol. The input stream is split into non-overlapping L-frame chunks aligned with the temporal stride of ReAE. Each DiT forward pass processes only the current latent chunk, without access to frames from future chunks, or a rolling KV cache. Cross-chunk continuity is instead maintained by the streaming ReAE, whose encoder and decoder carry boundary states s_E^k and s_D^k across chunks. For the k-th input chunk X_k,

$$z^k_{\text{LQ}},\, s^k_E = E^{\text{str}}_\phi(X_k,\, s^{k-1}_E), \quad \hat{z}^k_{\text{HQ}} = z^k_{\text{LQ}} - v_\theta(z^k_{\text{LQ}},\, 1), \quad Y_k,\, s^k_D = D^{\text{str}}_\phi(\hat{z}^k_{\text{HQ}},\, s^{k-1}_D). \tag{7}$$

The decoder discards causal-padding frames in the first chunk, and the last chunk is padded to the ReAE stride with only valid frames retained.

4 Experiments

4.1 Experimental Setup

Implementation. SwiftVR builds on Wan2.2-TI2V-5B (Wan et al., 2025) and is trained with AdamW (Loshchilov & Hutter, 2019) and DeepSpeed ZeRO-2 (Rajbhandari et al., 2020) on 8× H100-80G GPUs. We use 33-frame 768×1280 clips for ReAE fine-tuning, latent flow matching, and window-attention distillation, and 13-frame multi-resolution clips for joint fine-tuning. The learning rate is $2{\times}10^{-5}$ for Stage 1 and $1{\times}10^{-5}$ for Stages 2–3. The ReAE loss weights $(\lambda^{\text{ReAE}}_{\text{lpips}}, \lambda^{\text{ReAE}}_{\text{temp}}, \lambda^{\text{ReAE}}_{\text{gan}})$ are $(1.0, 1.0, 0.05)$, while those for joint fine-tuning are $(0.5, 1.0, 1.0)$, with $\lambda_{\text{distill}} = 1.0$. We train on curated UltraVideo clips (Xue et al., 2025), with LQ–HQ pairs synthesized using the RealBasicVSR degradation pipeline (Chan et al., 2022b).

Evaluation. We evaluate on three synthetic benchmarks: SPMCS (Tao et al., 2017), UDM10 (Yi et al., 2019), and YouHQ40 (Zhou et al., 2024). These benchmarks use the same degradation protocol as training. We also evaluate on the real-world VideoLQ benchmark (Chan et al., 2022b), which contains real-world videos without ground-truth references. All methods are evaluated under a unified chunk-based streaming protocol, with implementation details provided in the supplementary material. We compare SwiftVR with three categories of real-world video restoration baselines: non-diffusion methods, including Real-ESRGAN (Wang et al., 2021), RealBasicVSR (Chan et al., 2022b), and RealViFormer (Zhang & Yao, 2024); the multi-step diffusion method Upscale-A-Video (Zhou et al., 2024); and one-step diffusion methods, including DOVE (Chen et al., 2025b), SeedVR2-3B (Wang et al., 2026), and FlashVSR-Tiny (Zhuang et al., 2025).

Table 1: Quantitative comparison on synthetic and real-world video restoration benchmarks. Methods are grouped by method type for comparison. ↑ and ↓ indicate that higher and lower values are better, respectively. The best results are highlighted in red.

Dataset	Metric	Real-ESRGAN	RealBasicVSR	RealViFormer	Upscale-A-Video	DOVE	SeedVR2-3B	FlashVSR-Tiny	SwiftVR (Ours)
SPMCS	PSNR ↑	21.67	22.54	22.96	20.91	23.31	22.23	21.56	22.33
	SSIM ↑	0.5686	0.5743	0.5958	0.4795	0.6272	0.5984	0.5466	0.5742
	LPIPS ↓	0.3631	0.3572	0.3288	0.4097	0.2745	0.2619	0.2736	0.2837
	DISTS ↓	0.2204	0.2121	0.2083	0.2361	0.1662	0.1410	0.1631	0.1535
	CLIP-IQA ↑	0.5079	0.4354	0.3994	0.5712	0.5127	0.5441	0.5314	0.5011
	MUSIQ ↑	66.47	63.29	64.63	69.12	69.67	68.76	69.94	71.74
	MANIQA ↑	0.3938	0.3350	0.3116	0.4067	0.3871	0.4137	0.3961	0.3866
	NIQE ↓	3.3081	3.3123	3.5901	3.1558	4.6175	3.5371	3.6077	3.4166
UDM10	PSNR ↑	24.52	25.09	26.24	24.57	26.97	26.13	24.02	25.58
	SSIM ↑	0.7313	0.7331	0.7690	0.6883	0.7953	0.7712	0.7028	0.7659
	LPIPS ↓	0.3281	0.3291	0.2885	0.3249	0.2172	0.2192	0.2483	0.2508
	DISTS ↓	0.1901	0.1868	0.1752	0.1774	0.1291	0.1089	0.1334	0.1311
	CLIP-IQA ↑	0.4611	0.4482	0.4022	0.4355	0.4744	0.4032	0.4733	0.5241
	MUSIQ ↑	58.31	62.03	58.85	60.60	63.33	57.52	65.95	67.34
	MANIQA ↑	0.3398	0.3194	0.2929	0.2907	0.3423	0.2832	0.3588	0.3609
	NIQE ↓	4.0600	3.6600	4.0476	3.9552	5.0568	4.5434	3.7783	3.8798
YouHQ40	PSNR ↑	22.75	22.35	23.44	22.17	24.29	23.11	22.24	22.67
	SSIM ↑	0.6336	0.5873	0.6234	0.5651	0.6722	0.6427	0.5871	0.6048
	LPIPS ↓	0.3642	0.4137	0.3773	0.3781	0.3039	0.2924	0.2989	0.2947
	DISTS ↓	0.1844	0.1979	0.1982	0.1862	0.1485	0.1265	0.1373	0.1195
	CLIP-IQA ↑	0.4518	0.4903	0.4410	0.4876	0.4551	0.4845	0.5227	0.5950
	MUSIQ ↑	57.28	64.72	62.30	60.77	60.86	60.46	66.98	69.81
	MANIQA ↑	0.3078	0.3012	0.2841	0.3040	0.3061	0.3227	0.3659	0.3669
	NIQE ↓	3.8349	3.1222	3.3621	3.4593	4.8584	3.5318	3.4382	3.1634
VideoLQ	CLIP-IQA ↑	0.3619	0.3801	0.3465	0.2892	0.2880	0.2319	0.3616	0.3789
	MUSIQ ↑	49.86	55.05	52.12	45.23	44.59	40.42	50.93	55.43
	MANIQA ↑	0.2903	0.3006	0.2796	0.2416	0.2566	0.2179	0.2825	0.2767
	NIQE ↓	4.2017	3.7051	4.0577	4.6649	5.3566	5.3106	4.0159	4.0468

Metrics. For synthetic benchmarks, we report PSNR (Huynh-Thu & Ghanbari, 2008) and SSIM (Wang et al., 2004) for fidelity, LPIPS (Zhang et al., 2018) and DISTS (Ding et al., 2020) for perceptual similarity, and CLIP-IQA (Wang et al., 2023), MUSIQ (Ke et al., 2021), MANIQA (Yang et al., 2022), and NIQE (Mittal et al., 2012) for no-reference quality. For real-world benchmarks, only no-reference metrics are reported. Temporal quality is further assessed with the video-level DOVER (Wu et al., 2023) in the supplementary material. Streaming efficiency is evaluated by FPS and peak GPU memory.

4.2 Comparison with Existing Methods

Quantitative Comparisons. Table 1 summarizes the quantitative results. SwiftVR achieves the best MUSIQ on all four benchmarks, reaching 71.74, 67.34, 69.81, and 55.43 on SPMCS, UDM10, YouHQ40, and VideoLQ, respectively. It also ranks first in CLIP-IQA and MANIQA on UDM10 and YouHQ40, with the largest margin on YouHQ40 CLIP-IQA (0.5950 vs. 0.5227). For full-reference perceptual quality, SwiftVR obtains the best DISTS on YouHQ40 (0.1195 vs. 0.1265). Compared with FlashVSR-Tiny, the closest one-step streaming baseline, SwiftVR improves PSNR by 0.77, 1.56, and 0.43 dB on the three synthetic benchmarks, indicating that its perceptual gains do not come at the cost of fidelity. Fidelity-oriented methods such as DOVE obtain the highest PSNR and SSIM, but lag behind in perceptual quality (e.g., 60.86 vs. 69.81 MUSIQ on YouHQ40). Overall, SwiftVR delivers leading perceptual quality with competitive fidelity, and also ranks first in video-level DOVER on SPMCS and YouHQ40 (Table 7 in Appendix B.4).

Qualitative Comparisons. Figure 5 compares visual quality on real-world videos using enlarged local patches. The two examples cover complementary restoration challenges: fine feather textures on the falcon's head and beak, and repeated thin structures in the street scene, including branches, foliage, fences, and the car. Regression-based baselines, including Real-ESRGAN, RealBasicVSR, and RealViFormer, recover global silhouettes but oversmooth fine details and introduce color fringing along branches. Although DOVE achieves higher PSNR and SSIM, its outputs exhibit oversmoothed head feathers and foliage, reflecting its stronger emphasis on pixel fidelity. SeedVR2-3B and FlashVSR-Tiny recover more high-frequency content but introduce localized color shifts, halos, or over-sharpening near branches and car contours. In contrast, SwiftVR produces sharper and more natural reconstructions, with directional feather textures, cleaner beak details, clearer branch boundaries, better leaf separation, and sharper car contours. These observations are consistent with the



Figure 5: Qualitative comparison on real-world video clips. **Top:** a perched falcon, with crops showing the head, beak, and a bare branch against the sky. **Bottom:** a residential street with autumn foliage and a parked car, including crops of the dense leaf canopy and the car behind a chain-link fence. Columns from left to right show the low-quality input (LQ), Real-ESRGAN (Wang et al., 2021), RealBasicVSR (Chan et al., 2022b), RealViFormer (Zhang & Yao, 2024), DOVE (Chen et al., 2025b), SeedVR2-3B (Wang et al., 2026), FlashVSR-Tiny (Zhuang et al., 2025), and SwiftVR (Ours). Best viewed at high magnification.

improvements in perceptual metrics. Although RealBasicVSR achieves slightly better CLIP-IQA, MANIQA, and NIQE on VideoLQ, its visual results remain overly smooth. More visual comparisons are provided in the supplementary material.

4.3 Efficiency Analysis

At 2K, SwiftVR achieves the highest throughput among the evaluated one-step VR methods (Table 2). It processes a 24-frame chunk in 0.766 s, equivalent to 31.32 FPS: approximately 3.3× the throughput of FlashVSR-Tiny and over 20× that of DOVE and SeedVR2-3B. The latter two require VAE tiling to fit in memory at this resolution.

Table 2: Efficiency comparison of one-step video restoration methods at 2560×1440 on a single H100 under causal streaming, measured over 24 output frames. DOVE and SeedVR2-3B exceed the memory limit with their default VAEs; therefore, we enable `use_tile=True`.

Metric	DOVE	SeedVR2-3B	FlashVSR-Tiny	SwiftVR (Ours)
Avg. Time (s)	27.615	17.320	2.493	0.766
FPS	0.87	1.39	9.61	31.32
Peak Mem. (GB)	59.24	35.35	34.35	38.01

Table 3 extends the 2K comparison across three resolutions on a single H100. At 1080p, 2K, and 4K, SwiftVR achieves 54.42, 31.32, and 13.84 FPS, with peak memory of 29.26, 38.01, and 60.91 GB, respectively. The DiT remains the dominant runtime component, taking 327.72, 566.31, and 1270.10 ms per chunk at these resolutions, consistent with near-linear scaling in spatial tokens. At 4K, SwiftVR fits on the H100, whereas all evaluated diffusion-based VR baselines exceed its memory limit, even with VAE tiling enabled where supported.

The same streaming protocol also runs on consumer-grade gpus. At 1080p, SwiftVR sustains 26 FPS on a single NVIDIA RTX 5090 with $L = 24$, using less than 32 GB of GPU memory. This throughput exceeds the 24 FPS frame rate common in video content. To our knowledge, SwiftVR is the first generative VR

Table 3: Runtime breakdown of our SwiftVR per 24-frame chunk on a single H100 using bfloat16 and the default streaming protocol. FPS reports measured end-to-end streaming throughput at each resolution.

Resolution	Enc. (ms) ↓	DiT (ms) ↓	Dec. (ms) ↓	Mem. (GB) ↓	FPS ↑
1920×1080	25.67	327.72	85.37	29.26	54.42
2560×1440	45.10	566.31	151.74	38.01	31.32
3840×2160	111.06	1270.10	344.27	60.91	13.84

model to achieve real-time 1080p throughput on a consumer-grade GPU. The closely related one-step method FlashVSR (Zhuang et al., 2025) reports 17 FPS at 768×1408 on a server-class A100 using a dedicated block-sparse attention kernel. In contrast, MFSWA expresses window locality through tensor indexing and invokes only the standard dense SDPA interface. Deploying SwiftVR on the RTX 5090 therefore requires neither porting a custom kernel nor retraining the model.

4.4 Ablation Study

MFSWA. We compare the full-attention teacher with two shifted-window students on 24-frame chunks at 2560×1440: Swin-style SW-MSA with cyclic shifts and attention masks (Liu et al., 2021), and MFSWA (Table 4). Both use $(16, 16)$ windows with identical shift schedules and training settings. The teacher, SW-MSA, and MFSWA reach 19.36, 27.47, and 31.32 FPS. SW-MSA improves throughput by 41.9% over the teacher, while MFSWA adds 14.0% over SW-MSA ($1.62\times$ the teacher). MFSWA reduces DiT time by 16.0% (674.49 to 566.31 ms) and improves PSNR/LPIPS from 25.34 dB/0.2637 to 25.58 dB/0.2508, versus 25.86 dB/0.2417 for the teacher. MFSWA replaces cyclic shifts and masks with boundary-clamped gathering and deterministic output selection. The resulting uniform dense windows span all frames within each chunk and support fused SDPA. Because border-token neighborhoods differ, the gain cannot be attributed to mask removal alone. Peak memory is 38.01 GB, partly reflecting buffers for overlapping windows and precomputed indices: above 35.37 GB for the teacher but below 38.17 GB for SW-MSA.

Table 4: MFSWA ablation on 24-frame chunks at 2560×1440. SW-MSA uses Swin-style cyclic shifts and masks (Liu et al., 2021). DiT reports per-chunk runtime; Mem. reports peak GPU memory.

Variant	PSNR↑	LPIPS↓	DiT (ms)↓	Mem. (GB)↓	FPS↑
Full Attention (Teacher)	25.86	0.2417	1039.11	35.37	19.36
SW-MSA (Swin)	25.34	0.2637	674.49	38.17	27.47
MFSWA (Ours)	25.58	0.2508	566.31	38.01	31.32

ReAE. Table 5 compares ReAE with Wan2.2-VAE (Wan et al., 2025) and a tiny autoencoder (Boer Bohan, 2025) on $25 \times 1088 \times 1920$ clips. Wan2.2-VAE achieves the highest reconstruction quality (35.48 dB PSNR, 0.0513 LPIPS) but requires 704.69 M parameters and 2714 ms decoding, prohibitive for real-time streaming. The tiny autoencoder decodes in 40 ms but degrades reconstruction to 30.89 dB PSNR and 0.1183 LPIPS. With 40.95 M parameters, ReAE encodes each clip in 34 ms and decodes it in 99 ms, approximately $27\times$ faster in decoding than Wan2.2-VAE, while improving PSNR by 1.85 dB over the tiny autoencoder. ReAE thus alleviates the autoencoder bottleneck and makes joint fine-tuning with the DiT tractable.

Table 5: Ablation study of ReAE on $25 \times 1088 \times 1920$ clips. Enc. and Dec. report per-clip encoding and decoding times; Mem. reports peak GPU memory.

Variant	PSNR↑	LPIPS↓	Params (M) ↓	Mem. (GB) ↓	Enc. (ms) ↓	Dec. (ms) ↓
Wan2.2-VAE	35.48	0.0513	704.69	24.86	818	2714
Tiny autoencoder	30.89	0.1183	11.42	7.74	33	40
ReAE	32.74	0.0777	40.95	16.97	34	99

5 Conclusion

We present SwiftVR, a one-step generative framework for real-time streaming VR. SwiftVR processes low-quality streams with a causal chunk-wise protocol and addresses the two dominant costs of one-step diffusion VR through complementary designs. Mask-free shifted-window self-attention confines attention to local spatial windows while keeping every attention call on the dense SDPA path, yielding a substantial speedup over full attention without custom kernels. The restoration-aware autoencoder replaces the heavy VAE with a lightweight design that greatly reduces encoding and decoding cost while preserving reconstruction quality. Experiments show that SwiftVR achieves strong perceptual quality, particularly on no-reference metrics, at a much lower inference cost than existing one-step methods. To our knowledge, SwiftVR is the first generative method to achieve real-time 1080p streaming restoration on a consumer-grade GPU, and the only evaluated diffusion-based method capable of 4K inference on a single H100. Real-time generative 4K restoration on consumer-grade GPUs remains an open challenge, motivating future work on further inference acceleration and more compact backbones.

AI USE STATEMENT

Generative AI tools were used to improve the readability and presentation of this paper. For the text, they were used for language polishing, rewriting selected passages for clarity and concision, and assistance with LaTeX formatting. For the figures, they were used for visual refinement. They were also used to check the presentation of the coverage-factor derivation in Appendix A.1. Generative AI tools were not used to develop the proposed methods, formulate research hypotheses, design or run experiments, implement the methods, process datasets, or interpret results. All AI-assisted text, formulas, and figures were verified by the authors against the implementation and experimental records, and the authors take full responsibility for the content of this paper.

REPRODUCIBILITY STATEMENT

Architecture and training objectives are described in Sec. 3, and implementation details in Sec. 4. Appendix A specifies the MFSWA index construction, and Appendix B.1 details the unified streaming evaluation protocol for all methods.

REFERENCES

Hongjun An, Wenhan Hu, Sida Huang, Siqi Huang, Ruanjun Li, Yuanzhi Liang, Jiawei Shao, Yiliang Song, Zihan Wang, Cheng Yuan, et al. Ai flow: Perspectives, scenarios, and approaches. *Vicinagearth*, 3(1):1, 2026.

Ollin Boer Bohan. Taehv: Tiny autoencoder for hunyuan video. `https://github.com/madebyollin/taehv`, 2025.

Yu Cao, Ziquan Liu, Zhensong Zhang, Jiankang Deng, Shaogang Gong, and Jifei Song. LiteVSR: Lightweight adaptation of frozen diffusion transformers for video super-resolution. In *Forty-third International Conference on Machine Learning*, 2026.

Kelvin CK Chan, Xintao Wang, Ke Yu, Chao Dong, and Chen Change Loy. Basicvsr: The search for essential components in video super-resolution and beyond. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4947–4956, 2021.

Kelvin CK Chan, Shangchen Zhou, Xiangyu Xu, and Chen Change Loy. Basicvsr++: Improving video super-resolution with enhanced propagation and alignment. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5972–5981, 2022a.

Kelvin CK Chan, Shangchen Zhou, Xiangyu Xu, and Chen Change Loy. Investigating tradeoffs in real-world video super-resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5962–5971, 2022b.

Xiangyu Chen, Xintao Wang, Jiantao Zhou, Yu Qiao, and Chao Dong. Activating more pixels in image super-resolution transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 22367–22377, 2023.

Xiangyu Chen, Xintao Wang, Wenlong Zhang, Xiangtao Kong, Yu Qiao, Jiantao Zhou, and Chao Dong. Hat: Hybrid attention transformer for image restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025a.

Xiangyu Chen, Jixiang Luo, Yuankai Fan, Haibin Huang, Chi Zhang, and Xuelong Li. Generative transmission: Rethinking computation, bandwidth, and memory in communication. *arXiv preprint arXiv:2607.17482*, 2026a.

Xiangyu Chen, Jixiang Luo, Jingyu Xu, Fangqiu Yi, Chi Zhang, and Xuelong Li. Generative video compression: towards 0.01% compression rate for video transmission. *Vicinagearth*, 3 (1):7, 2026b.

Zheng Chen, Zichen Zou, Kewei Zhang, Xiongfei Su, Xin Yuan, Yong Guo, and Yulun Zhang. Dove: Efficient one-step diffusion model for real-world video super-resolution. *arXiv preprint arXiv:2505.16239*, 2025b.

Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=mZn2Xyh9Ec.

Keyan Ding, Kede Ma, Shiqi Wang, and Eero P Simoncelli. Image quality assessment: Unifying structure and texture similarity. *IEEE transactions on pattern analysis and machine intelligence*, 44(5):2567–2581, 2020.

Jingwen He, Tianfan Xue, Dongyang Liu, Xinqi Lin, Peng Gao, Dahua Lin, Yu Qiao, Wanli Ouyang, and Ziwei Liu. Venhancer: Generative space-time enhancement for video generation. *arXiv preprint arXiv:2407.07667*, 2024.

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

Quan Huynh-Thu and Mohammed Ghanbari. Scope of validity of psnr in image/video quality assessment. *Electronics Letters*, 44(13):800–801, 2008. doi: 10.1049/el:20080522.

Junjie Ke, Qifei Wang, Yilin Wang, Peyman Milanfar, and Feng Yang. Musiq: Multi-scale image quality transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 5148–5157, 2021.

Benjamin Lefaudeux, Francisco Massa, Diana Liskovich, Wenhan Xiong, Vittorio Caggiano, Sean Naren, Min Xu, Jieru Hu, Marta Tintore, Susan Zhang, Patrick Labatut, Daniel Haziza, Luca Wehrstedt, Jeremy Reizenstein, and Grigory Sizov. xformers: A modular and hackable transformer modelling library, 2022.

Muyang Li, Tianle Cai, Jiaxin Cao, Qinsheng Zhang, Han Cai, Junjie Bai, Yangqing Jia, Kai Li, and Song Han. Distrifusion: Distributed parallel inference for high-resolution diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7183–7193, June 2024.

Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1833–1844, 2021.

Jingyun Liang, Yuchen Fan, Xiaoyu Xiang, Rakesh Ranjan, Eddy Ilg, Simon Green, Jiezhang Cao, Kai Zhang, Radu Timofte, and Luc Van Gool. Recurrent video restoration transformer with guided deformable attention. In *Advances in Neural Information Processing Systems*, 2022.

Jingyun Liang, Jiezhang Cao, Yuchen Fan, Kai Zhang, Rakesh Ranjan, Yawei Li, Radu Timofte, and Luc Van Gool. Vrt: A video restoration transformer. *IEEE Transactions on Image Processing*, 33:2171–2182, 2024.

Xinqi Lin, Jingwen He, Ziyan Chen, Zhaoyang Lyu, Bo Dai, Fanghua Yu, Yu Qiao, Wanli Ouyang, and Chao Dong. Diffbir: Toward blind image restoration with generative diffusion prior. In *European Conference on Computer Vision*, pp. 430–448. Springer, 2024.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Jingren Liu, Shuning Xu, Qirui Yang, Yun Wang, Xiangyu Chen, and Zhong Ji. Fape-ir: Frequency-aware planning and execution framework for all-in-one image restoration. *arXiv preprint arXiv:2511.14099*, 2025.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference. *arXiv preprint arXiv:2310.04378*, 2023.

Zhengyao Lv, Chenyang Si, Junhao Song, Zhenyu Yang, Yu Qiao, Ziwei Liu, and Kwan-Yee K. Wong. Fastercache: Training-free video diffusion model acceleration with high quality. In *International Conference on Learning Representations*, 2025.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. Deepcache: Accelerating diffusion models for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15762–15772, June 2024.

Anish Mittal, Rajiv Soundararajan, and Alan C Bovik. Making a "completely blind" image quality analyzer. *IEEE Signal processing letters*, 20(3):209–212, 2012.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2020. doi: 10.1109/SC41405.2020.00024.

Jay Shah, Ganesh Bikshandi, Ying Zhang, Vijay Thakkar, Pradeep Ramani, and Tri Dao. Flashattention-3: Fast and accurate attention with asynchrony and low-precision. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-2193.

Jiawei Shao and Xuelong Li. Ai flow at the network edge. *IEEE Network*, 40(1):330–336, 2025.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

Ying Tai, Rui Xie, Chen Zhao, Kai Zhang, Zhenyu Zhang, Jun Zhou, and Jian Yang. Addsr: Accelerating diffusion-based blind super-resolution with adversarial diffusion distillation. *Pattern Recognition*, pp. 113012, 2026.

Xin Tao, Hongyun Gao, Renjie Liao, Jue Wang, and Jiaya Jia. Detail-revealing deep video super-resolution. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

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.

Jianyi Wang, Kelvin CK Chan, and Chen Change Loy. Exploring clip for assessing the look and feel of images. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 2555–2563, 2023.

Jianyi Wang, Zongsheng Yue, Shangchen Zhou, Kelvin CK Chan, and Chen Change Loy. Exploiting diffusion prior for real-world image super-resolution. *International Journal of Computer Vision*, 132(12):5929–5949, 2024a.

Jianyi Wang, Zhijie Lin, Meng Wei, Yang Zhao, Ceyuan Yang, Chen Change Loy, and Lu Jiang. Seedvr: Seeding infinity in diffusion transformer towards generic video restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2161–2172, 2025.

Jianyi Wang, Shanchuan Lin, Zhijie Lin, Yuxi Ren, Meng Wei, Zongsheng Yue, Shangchen Zhou, Hao Chen, Yang Zhao, Ceyuan Yang, Xuefeng Xiao, Chen Change Loy, and Lu Jiang. Seedvr2: One-step video restoration via diffusion adversarial post-training. In *ICLR*, 2026.

Xintao Wang, Kelvin CK Chan, Ke Yu, Chao Dong, and Chen Change Loy. Edvr: Video restoration with enhanced deformable convolutional networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1954–1963. IEEE, 2019.

Xintao Wang, Liangbin Xie, Chao Dong, and Ying Shan. Real-esrgan: Training real-world blind super-resolution with pure synthetic data. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1905–1914, 2021.

Yufei Wang, Wenhan Yang, Xinyuan Chen, Yaohui Wang, Lanqing Guo, Lap-Pui Chau, Ziwei Liu, Yu Qiao, Alex C Kot, and Bihan Wen. Sinsr: diffusion-based image super-resolution in a single step. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 25796–25805, 2024b.

Zhendong Wang, Xiaodong Cun, Jianmin Bao, Wengang Zhou, Jianzhuang Liu, and Houqiang Li. Uformer: A general u-shaped transformer for image restoration. In *CVPR*, pp. 17683–17693, 2022.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

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 *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 20087–20097. IEEE, 2023.

Rongyuan Wu, Lingchen Sun, Zhiyuan Ma, and Lei Zhang. One-step effective diffusion network for real-world image super-resolution. *Advances in Neural Information Processing Systems*, 37: 92529–92553, 2024a.

Rongyuan Wu, Tao Yang, Lingchen Sun, Zhengqiang Zhang, Shuai Li, and Lei Zhang. Seesr: Towards semantics-aware real-world image super-resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 25456–25467, 2024b.

Rui Xie, Yinhong Liu, Penghao Zhou, Chen Zhao, Jun Zhou, Kai Zhang, Zhenyu Zhang, Jian Yang, Zhenheng Yang, and Ying Tai. Star: Spatial-temporal augmentation with text-to-video models for real-world video super-resolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 17108–17118, 2025.

Zhucun Xue, Jiangning Zhang, Teng Hu, Haoyang He, Yinan Chen, Yuxuan Cai, Yabiao Wang, Chengjie Wang, Yong Liu, Xiangtai Li, and Dacheng Tao. Ultravideo: High-quality uhd video dataset with comprehensive captions. In *Advances in Neural Information Processing Systems*, 2025. Datasets and Benchmarks Track.

Sidi Yang, Tianhe Wu, Shuwei Shi, Shanshan Lao, Yuan Gong, Mingdeng Cao, Jiahao Wang, and Yujiu Yang. Maniqa: Multi-dimension attention network for no-reference image quality assessment. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1191–1200, 2022.

Xi Yang, Wangmeng Xiang, Hui Zeng, and Lei Zhang. Real-world video super-resolution: A benchmark dataset and a decomposition based learning scheme. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4781–4790, 2021.

Peng Yi, Zhongyuan Wang, Kui Jiang, Junjun Jiang, and Jiayi Ma. Progressive fusion video super-resolution network via exploiting non-local spatio-temporal correlations. In *IEEE International Conference on Computer Vision (ICCV)*, pp. 3106–3115, 2019.

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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6613–6623, 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 *CVPR*, 2025.

Fanghua Yu, Jinjin Gu, Zheyuan Li, Jinfan Hu, Xiangtao Kong, Xintao Wang, Jingwen He, Yu Qiao, and Chao Dong. Scaling up to excellence: Practicing model scaling for photo-realistic image restoration in the wild. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 25669–25680, 2024.

Jintao Zhang, Jia Wei, Pengle Zhang, Jun Zhu, and Jianfei Chen. Sageattention: Accurate 8-bit attention for plug-and-play inference acceleration. In *International Conference on Learning Representations (ICLR)*, 2025a.

Jintao Zhang, Kai Jiang, Chendong Xiang, Weiqi Feng, Yuezhou Hu, Haocheng Xi, Jianfei Chen, and Jun Zhu. Spargeattention2: Trainable sparse attention via hybrid top-k+ top-p masking and distillation fine-tuning. *arXiv preprint arXiv:2602.13515*, 2026.

Peiyuan Zhang, Yongqi Chen, Haofeng Huang, Will Lin, Zhengzhong Liu, Ion Stoica, Eric Xing, and Hao Zhang. Vsa: Faster video diffusion with trainable sparse attention. *arXiv preprint arXiv:2505.13389*, 2025b.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.

Yuehan Zhang and Angela Yao. Realviformer: Investigating attention for real-world video super-resolution. In *European conference on computer vision*, pp. 412–428. Springer, 2024.

Shangchen Zhou, Peiqing Yang, Jianyi Wang, Yihang Luo, and Chen Change Loy. Upscale-A-video: Temporal-consistent diffusion model for real-world video super-resolution. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2535–2545, 2024.

Junhao Zhuang, Shi Guo, Xin Cai, Xiaohui Li, Yihao Liu, Chun Yuan, and Tianfan Xue. Flashvsr: Towards real-time diffusion-based streaming video super-resolution. *arXiv preprint arXiv:2510.12747*, 2025.

Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, and Linfeng Zhang. Accelerating diffusion transformers with token-wise feature caching. In *International Conference on Learning Representations*, 2025.

A MFSWA Design and Analysis

The main text introduces MFSWA through three components: dense blocks, shifted indexing, and the mask-free treatment of boundary windows. This section details the boundary-clamped gather, analyzes its redundant attention cost, summarizes the dense SDPA implementation, and relates the theoretical attention saving to the measured speedup.

A.1 Boundary-clamped Gather Overhead

Construction. Window anchors are placed on a grid with stride (w_h, w_w), offset by $(w_h/2, w_w/2)$ in odd layers. Each anchor is clamped to $[0, H - w_h] \times [0, W - w_w]$, so every window covers a full block of $T \cdot w_h \cdot w_w$ valid tokens and no padding tokens are introduced. In even layers, clamping affects only the trailing windows when H or W is not divisible by the window size. In odd layers, the leading windows are always clamped, and the trailing windows are clamped whenever they extend beyond the grid. Tokens covered by multiple windows receive the output of a single window selected by a fixed priority rule. The gathered tensor has a regular shape of $(B \cdot N_w) \times \text{heads} \times (T \cdot w_h \cdot w_w) \times d$, avoiding ragged tensors, padding masks, and variable-size boundary windows.

Source of overhead. Boundary clamping can place a token in multiple windows, so a layer attends to more than $H \cdot W$ spatial tokens. Since every window has the same size, attention compute is proportional to the total number of gathered tokens. We define α as the ratio of gathered spatial tokens to $H \cdot W$. Thus, α measures compute relative to an ideal overlap-free partition into equal-size windows, rather than to a ragged boundary implementation whose cost depends on the squared sizes

of boundary windows. Because partitioning is spatial-only, the temporal factor T cancels. Half-window shifting creates additional boundary overlap, so odd layers never incur less overhead than even layers.

Coverage factor. Each window is the Cartesian product of a row interval and a column interval, so α factorizes over the two spatial axes. Consider one axis of length S with window size $w \le S$. Even layers use $n_{\text{even}} = \lceil S/w \rceil$ windows. Odd layers use one clamped window at the leading border, followed by windows anchored at $w/2, 3w/2, \ldots$ that cover the remaining $S - w/2$ positions, giving $n_{\text{odd}} = 1 + \lceil (S - w/2)/w \rceil$. The per-axis coverage is $\rho = n\,w/S$, and $\alpha = \rho(H, w_h)\,\rho(W, w_w)$. Using $x \le \lceil x \rceil < x + 1$,

$$1 \;\le\; \rho_{\text{even}} \;<\; 1 + \frac{w}{S}, \qquad 1 + \frac{w}{2S} \;\le\; \rho_{\text{odd}} \;<\; 1 + \frac{3w}{2S}.$$

The even-layer factor equals 1 exactly when S is divisible by w, whereas $\rho_{\text{odd}} > 1$ for every S. Moreover, $\lceil S/w - 1/2 \rceil \ge \lceil S/w \rceil - 1$ implies $n_{\text{odd}} \ge n_{\text{even}}$, with equality possible. Multiplying across both axes,

$$\left(1 + \tfrac{w_h}{2H}\right)\left(1 + \tfrac{w_w}{2W}\right) \;\le\; \alpha_{\text{odd}} \;<\; \left(1 + \tfrac{3w_h}{2H}\right)\left(1 + \tfrac{3w_w}{2W}\right).$$

The bounds depend only on w_h/H and w_w/W, so the overhead is content-independent and approaches 1 as resolution increases.

Example. At 2560×1440, the latent size is $(H, W) = (45, 80)$ and $(w_h, w_w) = (16, 16)$. Even layers use 3×5 windows, giving $\alpha_{\text{even}} = \frac{48}{45} \cdot \frac{80}{80} \approx 1.07$, and odd layers use 4×6 windows, giving $\alpha_{\text{odd}} = \frac{64}{45} \cdot \frac{96}{80} \approx 1.71$, for an average of about 1.39 over alternating layers. At 3840×2160, the latent size is $(68, 120)$, and both layer types use 5×8 windows, giving $\alpha_{\text{even}} = \alpha_{\text{odd}} = \frac{80}{68} \cdot \frac{128}{120} \approx 1.25$.

Relation to measured memory. The coverage factor α describes redundant attention compute, not peak memory. The gathered Q, K, and V windows are transient SDPA inputs rather than persistent activations, so the odd-layer overhead of $\alpha_{\text{odd}} \approx 1.71$ at 2560×1440 does not imply a comparable memory increase. In practice, peak memory is dominated by resident activations and workspace, and increases only modestly from 35.37 to 38.01 GB relative to the full-attention teacher (Table 4). The redundant attention is bounded by the resolution and window size, and decreases at higher resolutions.

A.2 Dense SDPA Implementation

In each layer, all gathered windows share one shape and are batched into dense SDPA calls. The window layout is encoded by two index tensors, one for gathering and one for scattering, which are precomputed and cached per resolution. The training graph contains no attention mask, padding token, block-sparse descriptor, or cyclic shift. MFSWA thus obtains locality from the window partition while keeping every attention call dense.

A.3 Attention Complexity and Attained Speedup

Window attention complexity. Ignoring the head count and feature dimension, which are shared by both variants, full attention over a latent chunk costs $C_{\text{full}} \propto (THW)^2$. Under MFSWA, each window contains $M = T w_h w_w$ tokens, and a layer gathers $N_w = \alpha HW/(w_h w_w)$ windows, where α is the coverage factor of Sec. A.1. The dominant attention term is therefore

$$C_{\text{window}} \;\propto\; N_w M^2 \;=\; \alpha T^2 HW\, w_h w_w, \qquad \frac{C_{\text{window}}}{C_{\text{full}}} \;\approx\; \frac{\alpha\, w_h w_w}{HW}. \tag{8}$$

For a fixed chunk length and window size, the spatial term thus grows linearly rather than quadratically in HW, while the dependence on T remains quadratic, consistent with restricting the partition to the spatial axes. With $(w_h, w_w) = (16, 16)$ and the grids of Sec. A.1, the layer-averaged ratio is $1.3867 \times 256/3600 \approx 0.099$ at 2560×1440 and $1.2549 \times 256/8160 \approx 0.039$ at 3840×2160.

Gap between estimated and measured speedup. Eq. (8) counts only the two batched matrix products inside attention. It excludes the QKV and output projections, the MLP blocks, normalization, and the gather and scatter copies, none of which are reduced by windowing. Since self-attention accounts for roughly 60% of the Stage-1 DiT latency, even a cost-free attention operator would cap the DiT speedup at about $2.5\times$. The measured DiT time falls from 1039.11 to 566.31 ms (Table 4), a $1.84\times$ gain, indicating that the residual attention cost exceeds the arithmetic estimate above: the gathered windows are numerous and individually small, so the kernels are more memory-bound and less arithmetically intensive than a single large attention call, and the index copies add bandwidth that does not appear in the FLOP count.

End-to-end speedup. The $1.62\times$ end-to-end gain in Table 4 is smaller still, because MFSWA leaves the autoencoder untouched. At 2560×1440, ReAE encoding and decoding take 45.10 and 151.74 ms per chunk (Table 3), about a quarter of the 0.766 s chunk time, and are identical for the teacher and the student. Ratios computed from Eq. (8) should therefore be read as an estimate of the attention arithmetic alone, not as a predicted throughput gain.

B More Evaluation and Deployment

This section specifies the unified streaming evaluation protocol, an extended efficiency comparison at 2560×1440, additional qualitative results, video-level perceptual quality, and cross-backend deployment results.

B.1 Unified Streaming Evaluation Protocol

A like-for-like comparison in Table 1 requires a common streaming setting. Because the baselines use different temporal strides and overlap conventions, we adopt a unified protocol. RealBasicVSR and RealViFormer process 24-frame chunks with a 4-frame overlap, and metrics are computed only on non-overlapped outputs. Upscale-A-Video, DOVE, and SeedVR2-3B process 25-frame chunks ($=4k+1$) with a 4-frame overlap. Real-ESRGAN and FlashVSR-Tiny use their official evaluation scripts, as they already operate per frame or per causal block. SwiftVR uses its native causal chunk protocol without overlap, with ReAE carrying boundary states across chunks. All methods use the same input resolution and test clips, and metrics are computed only on emitted frames. We use official implementations with their released default precision: `float32` for non-diffusion baselines and `bfloat16` for diffusion-based methods.

This protocol underlies both the quality results in Table 1 and the efficiency results in Table 2. Because all methods are evaluated under the same streaming constraint, the numbers may differ from those reported offline in the original papers. Chunking also affects efficiency, since attention cost scales quadratically with temporal length.

B.2 Extended Per-method Efficiency Comparison

Table 6: Extended efficiency comparison at 2560×1440 on one H100 under causal streaming, measured over 24 output frames. DOVE and SeedVR2-3B use `use_tile=True`.

Metric	Real-ESRGAN	RealBasicVSR	RealViFormer	DOVE	SeedVR2-3B	FlashVSR-Tiny	SwiftVR (Ours)
Params (M)	16.70	6.29	5.82	10,548	3,642	1,752	5,081
Avg. Time (s)	5.770	1.324	0.940	27.615	17.320	2.493	0.766
FPS	4.16	18.12	25.53	0.87	1.39	9.61	31.32
Peak Mem. (GB)	3.15	7.12	5.82	59.24	35.35	34.35	38.01

Table 6 extends the 2560×1440 efficiency comparison in Table 2 with non-generative baselines. Upscale-A-Video is excluded from this timing comparison because it requires 30 sampling steps, but it is included in the quality evaluation and the 4K memory check.

At 3840×2160, all compared diffusion-based VR methods, including the multi-step Upscale-A-Video, run out of memory on a single H100-80G under the same streaming protocol, even with VAE tiling enabled. SwiftVR sustains 13.84 FPS at this resolution with a peak memory of 60.91 GB (Table 3). The non-generative baselines (Real-ESRGAN, RealBasicVSR, and RealViFormer) fit at 3840×2160 but yield substantially lower perceptual quality, as shown in Table 1.

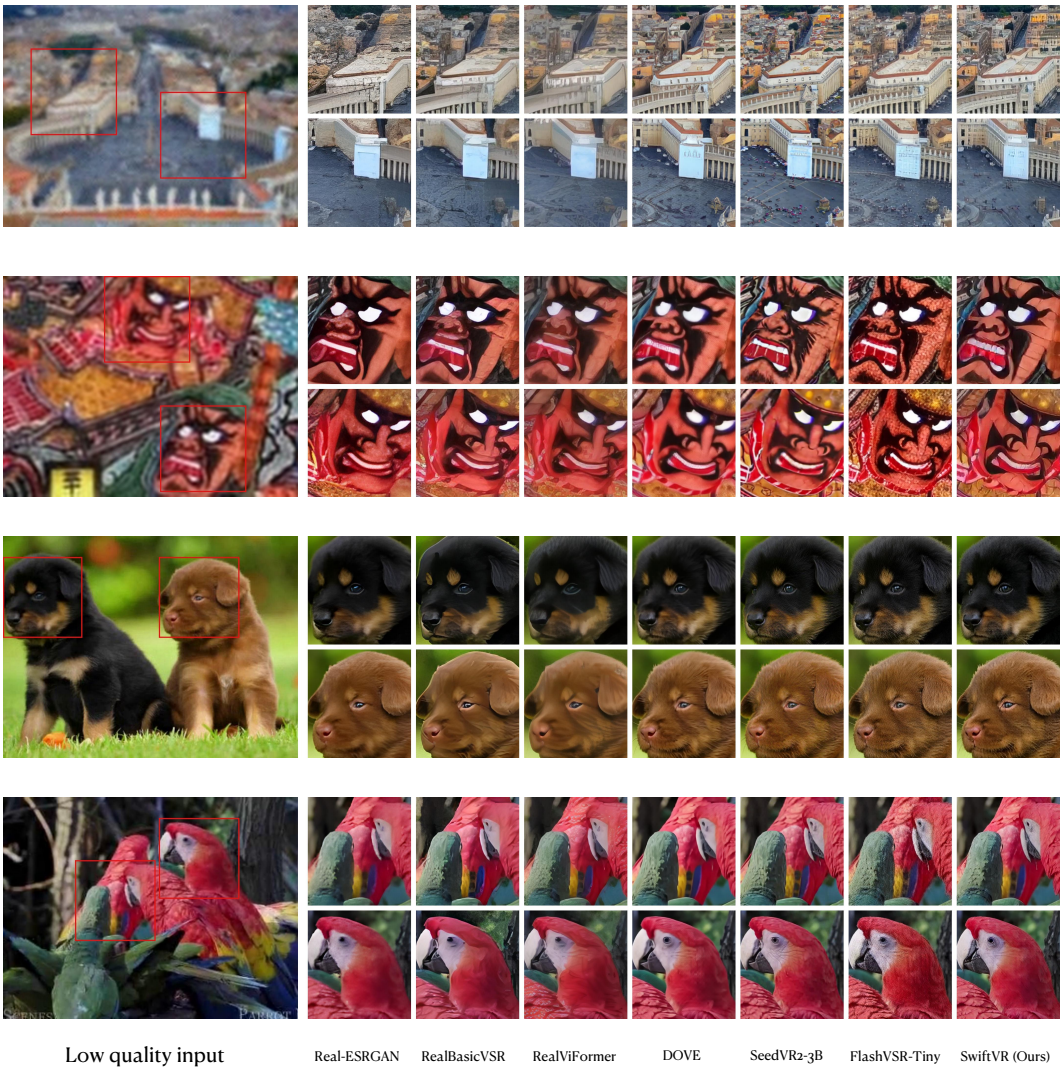


Figure 6: Additional qualitative comparisons on real-world videos. Columns show the low-quality input, Real-ESRGAN, RealBasicVSR, RealViFormer, DOVE, SeedVR2-3B, FlashVSR-Tiny, and SwiftVR (Ours).

B.3 Additional Visualization Results

Figure 6 presents additional qualitative comparisons on real-world videos, covering distant buildings, mural patterns, animal fur, and bird plumage. Regression-based methods recover coarse structures but smooth fine details and reduce local contrast. DOVE produces stable outputs but preserves less high-frequency detail. SeedVR2-3B and FlashVSR-Tiny recover sharper patterns but may introduce color shifts, halos, or excessive sharpening. SwiftVR restores clearer boundaries and more natural details, including roof edges, fur, and feather structures, with stable color and fewer local artifacts. These results are consistent with the perceptual gains in the main comparison.

B.4 Video-level Perceptual Quality

We additionally report DOVER (Wu et al., 2023), a no-reference video quality metric that takes temporal clips as input and is thus sensitive to flicker and texture instability. As shown in Table 7, SwiftVR ranks first on SPMCS and YouHQ40 and second on UDM10; on the real-world VideoLQ benchmark it stays within 0.031 of the top score. Because chunks are processed without overlap

and the DiT carries no cross-chunk state, these video-level scores indicate that the boundary states propagated by ReAE suffice to maintain temporal continuity across chunk boundaries.

Table 7: DOVER scores on synthetic and real-world benchmarks, higher is better.

Method	SPMCS	UDM10	YouHQ40	VideoLQ
Real-ESRGAN	0.4900	0.5175	0.6734	0.5142
RealBasicVSR	0.3770	0.4490	0.6623	0.5323
RealViFormer	0.3594	0.4134	0.6401	0.4953
Upscale-A-Video	0.4336	0.4778	0.6050	0.4292
DOVE	0.4980	0.5566	0.6897	0.5312
SeedVR2-3B	0.4722	0.4732	0.6692	0.4749
FlashVSR-Tiny	0.5045	0.5592	0.7077	0.5610
SwiftVR (Ours)	0.5087	0.5589	0.7190	0.5305

B.5 Cross-backend Deployment

Because MFSWA keeps every attention call on the standard dense SDPA interface, SwiftVR runs on different fused-attention backends without weight conversion. Table 8 reports throughput for five backends at 2560×1440. Peak memory remains 38.01 GB and restoration metrics match to the reported precision, so both are omitted.

Table 8: Cross-backend deployment on one H100 at 2560×1440. Peak memory is constant at 38.01 GB, and restoration metrics match to the reported precision.

Attention backend	FPS↑
PyTorch SDPA (cuDNN/Flash auto)	31.32
FlashAttention-2 (Dao, 2024)	31.36
FlashAttention-3 (Shah et al., 2024)	32.31
SageAttention (Zhang et al., 2025a)	30.70
xFormers (Lefaudeux et al., 2022)	31.36

On one H100, PyTorch SDPA already dispatches to the cuDNN/Flash path and matches FlashAttention-2 and xFormers within about 0.1%. FlashAttention-3 is about 3% faster than SDPA, whereas SageAttention is slightly slower at this scale and precision, although it can outperform FlashAttention on Ada-class consumer GPUs (Zhang et al., 2025a). These results confirm that MFSWA preserves dense SDPA compatibility while introducing window locality.

C Limitations and Future Work

Limitations. SwiftVR does not yet deliver real-time generative 4K restoration on consumer-grade GPUs. At 4K, it reaches 13.84 FPS with 60.91 GB peak memory on an H100, which fits a server-class GPU but exceeds consumer-grade GPU memory and remains below 24 FPS. In addition, chunk-wise processing introduces an algorithmic delay of L input frames before each chunk can be restored, which corresponds to about one second for $L = 24$ at 24 FPS input. The reported FPS therefore measures sustained throughput rather than end-to-end latency; a smaller L shortens this delay. Finally, because chunks are processed without overlap, restoration at a chunk boundary relies solely on the boundary states propagated by ReAE rather than on direct attention to the preceding chunk. The video-level scores in Sec. B.4 indicate that this is sufficient under the degradations considered, but scenes with abrupt cuts or very fast motion at a boundary may still benefit from a short overlap, at a proportional throughput cost.

Future work. SwiftVR currently uses no inference-side acceleration. Future work targets two directions. The first is inference acceleration, including post-training quantization, KV-state caching

and compression, and learned token reduction, all orthogonal to the architecture. The second is a smaller, more compressed backbone. Wan2.2-TI2V-5B remains large, so higher latent compression and smaller base models are likely necessary for real-time 4K on consumer-grade GPUs.