REAL-TIME STREAMING VIDEO STYLIZATION VIA RECTIFIED STYLE FLOW

Anonymous authors
Paper under double-blind review

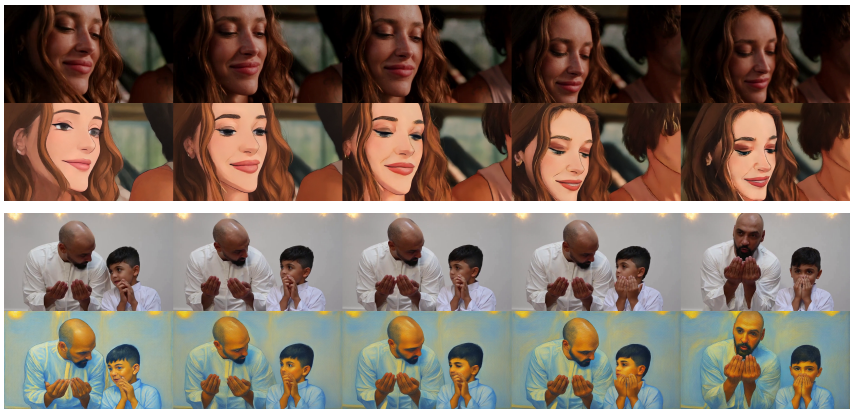

Figure 1: RSF transfer streaming video into different styles in real-time. Here the first row in each block exhibits the original video, while the second row stands for the stylized result of RSF.

ABSTRACT

Diffusion-based image stylization provides strong appearance control but is too slow and temporally unstable for streaming video. We introduce RSF, a staged framework for real-time streaming video stylization. Its core is *rectified style flow*: rather than generating from pure noise, the model transports a partially corrupted source distribution at noise level s to a target-style distribution. We first learn modular style LoRAs on paired source–target video clips. We then freeze a bank of these LoRAs and train the full video DiT with teacher ODE pairs and causal forcing, separating reusable autoregressive dynamics from style-specific parameters. The resulting causal teacher is distilled to a one-step student using distribution matching distillation, followed by generated-history rollout fine-tuning to reduce exposure bias. Finally, a lightweight recovery procedure adapts previously unseen style LoRAs without repeating the complete video training pipeline. At 832×480 resolution, optimized mixed-FP8 inference reaches 101.73 FPS, 78.76 ms median input-to-host-output latency, and 4.81 GiB steady GPU memory on one NVIDIA H100; it reaches 36.3 FPS and approximately 220 ms latency on an NVIDIA RTX 5090. Across the evaluated styles, RSF maintains competitive target-style alignment by CLIP-T while improving temporal consistency, with the lowest source-relative motion error among the compared methods. We will open-source all training code, model weights, and prepared datasets.

1 INTRODUCTION

Interactive content creation, game rendering, live communication, and video post-processing all benefit from a stylizer that transforms an incoming video stream with low latency. A useful system must satisfy three requirements at once: strong and controllable appearance transfer, temporal stability over an unbounded stream, and an inference cost compatible with real-time use. Existing solutions usually satisfy only a subset. Applying an image model to each frame preserves the capacity of

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.

large image generators but introduces flicker and repeated texture synthesis. Offline video diffusion improves coherence but processes fixed-length clips with many denoising evaluations.

Our starting observation is that stylization is substantially easier than unconditional generation when the source frame is still informative. We therefore define a rectified flow from a partially corrupted source distribution $\mathcal{P}_s$ to a target-style distribution $\mathcal{P}_y$. The intermediate noise level s exposes enough uncertainty for appearance to change while retaining the scene layout, identity, and motion needed for faithful video transfer. This local transport path also shortens the trajectory that must eventually be distilled.

The second observation is that style and streaming dynamics should be learned at different stages. RSF first learns a bank of style-specific LoRAs from aligned frame pairs. Keeping this bank frozen, it trains a shared full-parameter video DiT to follow teacher ODE pairs under causal histories. The causal model is then distilled to one denoising step and fine-tuned on its own short rollouts. This decomposition retains style modularity: a new image style LoRA can be made compatible with the distilled streaming backbone by a small recovery fine-tune rather than by retraining the entire pipeline.

Contributions.

- We formulate video stylization as rectified transport from an informative intermediate source distribution to a target-style distribution, explicitly separating source corruption from normalized style-flow time.
- We introduce a four-stage recipe combining a modular style LoRA bank, full-DiT causal forcing with teacher ODE pairs, one-step distribution matching distillation, and rollout fine-tuning.
- We demonstrate a low-latency, temporally stable streaming stylization system that runs in real time on a single consumer-level GPU, supporting interactive use cases such as game streaming and live communication.

2 Related Work

2.1 Image and video stylization

Image stylization methods learn appearance transfer from paired data, unpaired domains, textual instructions, or parameter-efficient adaptation of large diffusion models. Frame-wise deployment is attractive because each style can be represented by a compact LoRA, but independent stochastic generation causes temporal flicker. Video stylization methods add optical flow, temporal attention, or explicit correspondence constraints, usually at the cost of clip-wise processing. Our method preserves a modular image-style interface while learning shared causal dynamics at video level.

2.2 Video diffusion and streaming generation

Video diffusion models extend image generators with spatial–temporal attention and are commonly evaluated on fixed clips. Causal generation and rolling key–value caches instead permit bounded-memory streaming. However, teacher-forced causal training conditions on clean histories, whereas inference conditions on generated histories. Self-Forcing explicitly closes this train–test gap through autoregressive rollout training (Huang et al., 2025). We adopt the same general motivation, but study conditional style transport and retain a separately loadable style bank.

Streaming V2V systems provide the closest deployment comparisons. StreamV2V uses feature banks to stabilize an image-diffusion translation pipeline (Liang et al., 2025). StreamDiffusionV2 combines a causal video diffusion model with rolling KV-cache and scheduling optimizations (Feng et al., 2026). SANA-Streaming uses a hybrid linear/softmax attention video transformer for long-form editing (Zhao et al., 2026). JoyAI-Video-Edit uses autoregressive diffusion and a multimodal condition encoder for open-ended streaming edits (Xiao et al., 2026). LiveEdit distills a source-conditioned Wan editor through bidirectional editing, causal adaptation, and few-step distribution matching, and adds an autoregressive mask cache for unchanged regions (Wang et al., 2026). Unlike our intermediate-start style flow with separately loadable style LoRAs, its released V2V interface is controlled by an editing instruction. These systems differ in backbone, training data, inference resolution, and style-adapter

support. Our comparison therefore uses shared source content and style semantics while reporting native per-method settings, rather than attributing every difference to temporal modeling alone.

2.3 Diffusion and flow distillation

Rectified flow learns a velocity field along straight probability paths, supporting efficient ODE sampling (Liu et al., 2022; Lipman et al., 2022). Distribution Matching Distillation (DMD) trains a one-step generator by matching generated and real distributions through diffusion score differences (Yin et al., 2024b;a). We combine an intermediate-start rectified path with DMD and causal rollout training; the central question is not only reducing denoising steps, but retaining style compatibility under autoregressive deployment.

3 Preliminaries

3.1 Flow matching

Let $\mathbf{x}_0 \sim \mathcal{P}_0$ and $\mathbf{x}_1 \sim \mathcal{P}_1$ denote samples from source and target distributions. A linear probability path is

$$\mathbf{x}_t = (1-t)\mathbf{x}_0 + t\mathbf{x}_1, \qquad \mathbf{v}^*(\mathbf{x}_t, t) = \mathbf{x}_1 - \mathbf{x}_0, \tag{1}$$

and a velocity network $\mathbf{v}_\theta$ is trained with

$$\mathcal{L}_{\mathrm{FM}} = \mathbb{E}_{t,\mathbf{x}_0,\mathbf{x}_1}\left[\left\|\mathbf{v}_\theta(\mathbf{x}_t, t, c) - (\mathbf{x}_1 - \mathbf{x}_0)\right\|_2^2\right], \tag{2}$$

where c is a text or style condition. At inference, integrating the learned ODE transports samples from $\mathcal{P}_0$ to $\mathcal{P}_1$.

3.2 Parameter-efficient style adaptation

For a frozen linear projection W, LoRA represents a style-specific update as $W_y = W + B_y A_y$, where A_y and B_y have low rank (Hu et al., 2022). We denote the collection of style adapters by $\mathcal{B} = \{\phi_y\}_{y\in\mathcal{Y}}$. Only one ϕ_y is active for each sample. This interface is central to our design: the shared causal backbone models streaming dynamics, while ϕ_y selects the desired appearance.

3.3 Autoregressive video generation

A causal video model factorizes a latent stream into $p(\mathbf{z}_{1:T}) = \prod_i p(\mathbf{z}_i \mid \mathbf{z}_{<i}, c)$. During streaming, the keys and values from a bounded history are cached, and only the newest latent is generated. Teacher forcing uses clean historical latents during training and therefore differs from inference, where the cache contains model outputs. This exposure bias motivates the rollout stage in Section 4.6.

4 Method

4.1 Problem formulation

Given a source stream $\mathbf{x}^{\mathrm{src}}_{1:T}$ and style instruction y, our goal is to generate $\mathbf{x}^{y}_{1:T}$ online. The output should follow the content and motion of the source, match the target appearance, and remain stable for a stream longer than the training clips. Let F_θ denote a video DiT and ϕ_y a style LoRA. At deployment, the system keeps at most H historical latent frames and emits the next frame with one denoising evaluation after distillation.

4.2 Rectified style flow

Unconditional generation transports Gaussian noise to the data distribution. For stylization, this discards information already present in the source. Inspired by source-preserving editing methods (Meng et al., 2022), we construct an intermediate distribution $\mathcal{P}_s$ by partially corrupting source latents and learn transport from $\mathcal{P}_s$ to the target-style distribution $\mathcal{P}_y$:

$$\mathbf{x}_s = (1-s)\mathbf{x}_{\mathrm{src}} + s\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, I), \qquad \mathbf{x}_u = (1-u)\mathbf{x}_{\mathrm{tgt}} + u\mathbf{x}_s, \quad u \in [0,1], \qquad t = su, \tag{3}$$

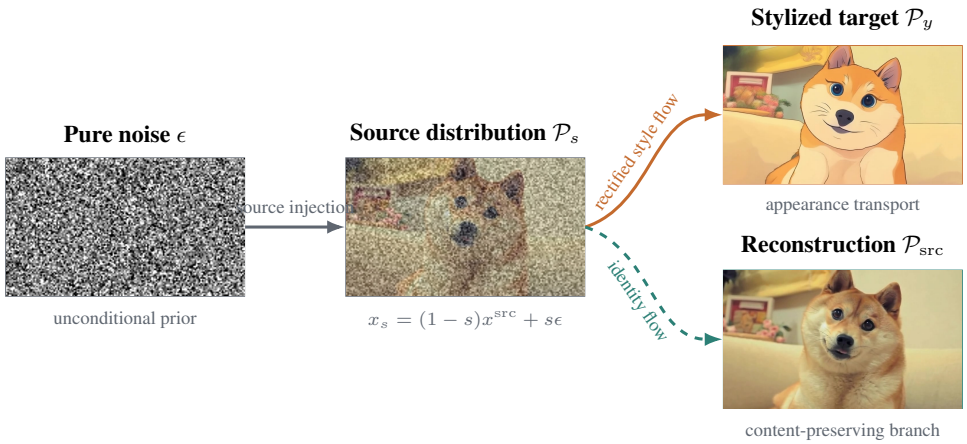

Figure 2: **Rectified Style Flow.** A partially corrupted source initializes the style and reconstruction branches. The original and stylized dog images are matched frame 16 from the same video.

Here $s \in (0, 1)$ is the source corruption level, u is a locally normalized coordinate on the shortened path, and $t = su$ is the native model time. The implementation follows the pretrained Wan (Wan Team et al., 2025) velocity convention, $\mathbf{v}^* = \partial \mathbf{x}_t / \partial t = (\mathbf{x}_s - \mathbf{x}_{\text{tgt}})/s$. We optimize

$$\mathcal{L}_{\text{RSF}} = \mathbb{E}\left[\left\| F_{\theta,\phi_y}(\mathbf{x}_u, t, c_y) - \mathbf{v}^* \right\|_2^2\right]. \tag{4}$$

We sample u uniformly and use unweighted mean squared error over latent elements. The denoiser therefore receives the native normalized timestep $t = su$ rather than u. We keep the variables distinct because confusing the pretrained noise schedule with the shortened style path changes the velocity by the factor $1/s$ and can reverse the integration convention. A smaller s preserves more geometry but limits stylistic change, while a larger s increases style freedom at the cost of correspondence. For a paired source–target sample, the local endpoint displacement is $(1-s)\mathbf{x}_{\text{src}} + s\boldsymbol{\epsilon} - \mathbf{x}_{\text{tgt}}$. With independent zero-mean unit-variance noise in d latent dimensions, its expected squared length is

$$\mathbb{E}\|\mathbf{x}_s - \mathbf{x}_{\text{tgt}}\|_2^2 = \mathbb{E}\|(1-s)\mathbf{x}_{\text{src}} - \mathbf{x}_{\text{tgt}}\|_2^2 + s^2 d. \tag{5}$$

When source and target are geometrically aligned and their residual is smaller than the variation of an independent noise draw, this displacement and the conditional ambiguity of the transport are reduced relative to generation from pure noise. The one-step student therefore learns a shorter, source-conditioned map rather than the full noise-to-video map. This is a motivation for one-step distillation, not a guarantee: a large appearance or geometry change can invalidate the proximity assumption, and s must still provide enough freedom to reach the target style. The method does not add a conditioning encoder, cross-attention branch, or extra video tokens. The Stage-1 LoRA update can be fused into each existing projection at deployment; Stages 2–4 update the original DiT pathway. The short history still incurs cache storage and attention work, which we include in measured end-to-end latency.

4.3 Stage 1: Modular Style-Flow LoRAs

For every style y, we train a rank-r LoRA ϕ_y on aligned source and stylized video clips while freezing the base DiT. Each target prompt names the style and describes the desired rendered domain. Training with Equation (4) makes the adapter specialize in appearance transport along the same s-to-target path used by the video system. The result is a bank $\mathcal{B}$ whose members share one backbone and can be swapped without changing the architecture. We did not directly train a full DiT since we found the results of different styles slowly converge to a specific style.

4.4 Stage 2: LoRA-aware causal forcing

Stage 1 uses the pretrained video DiT on 81-frame clips, so it already has bidirectional temporal modeling; it does not by itself provide bounded-history causal streaming. We next freeze $\mathcal{B}$ and train the full parameters of F_θ to reproduce bidirectional teacher outputs causally. For each clip we

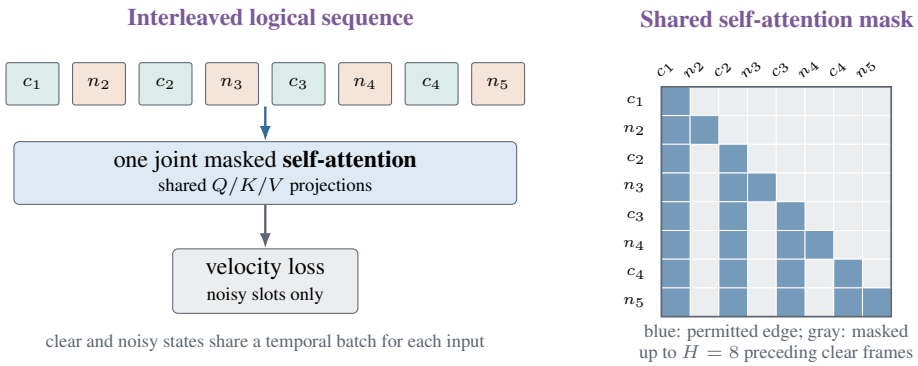

Figure 3: **LoRA-aware interleaved AR teacher forcing.** Clear and noisy latents form one temporal token sequence and pass through the same masked self-attention.

sample a style y, activate ϕ_y, and use the bidirectional teacher to generate a stylized endpoint $\mathbf{z}^{\text{teach}}$ from the source video. Fresh noise constructs $\mathbf{z}_s = (1-s)\mathbf{z}_{\text{src}} + s\epsilon$ and $\mathbf{z}_u = (1-u)\mathbf{z}^{\text{teach}} + u\mathbf{z}_s$; intermediate teacher ODE states are not cached. The student predicts the corresponding velocity while attending only to bounded, lightly perturbed teacher-output history:

$$\mathcal{L}_{\text{CF}} = \mathbb{E}\left[\left\| F^{\text{causal}}_{\theta,\phi_y}(\mathbf{z}_u, su, \widetilde{\mathbf{z}}^{\text{teach}}_{i-H:i-1}, c_{\text{neutral}}) - (\mathbf{z}_s - \mathbf{z}^{\text{teach}})/s \right\|_2^2\right]. \tag{6}$$

We use an interleaved clear/noisy sequence during training so that multiple causal prefix lengths are supervised in one forward pass. This is one masked self-attention computation: clear and noisy tokens share the same query/key/value projections, while a block mask removes disallowed temporal edges. The current logical order is $c_1, n_2, c_2, n_3, \ldots, c_{10}, n_{11}$ (stored as contiguous clear and noisy blocks for the real implementation); each token retains its own within-frame attention and can access at most the preceding $H = 8$ near-clean frames. The shared text prompt does not name a style; the frozen LoRA selects it. Context tokens are excluded from the prediction loss. Updating the full DiT gives the shared temporal model sufficient capacity, whereas freezing each LoRA encourages style to remain an external control rather than being absorbed by the backbone. This stage yields a multi-step autoregressive teacher.

4.5 Stage 3: One-step distribution matching

Running the causal ODE teacher for many evaluations per frame is not suitable for real-time inference. We initialize a full-DiT student from Stage 2 and distill its per-frame transport to one evaluation while keeping the routed style LoRA bank frozen. Following DMD, frozen real-score and trainable fake-score networks estimate the distribution-matching direction. The student update can be summarized as

$$\nabla_\theta \mathcal{L}_{\text{DMD}} \propto \mathbb{E}\left[(s_{\text{fake}}(\hat{\mathbf{z}}) - s_{\text{real}}(\hat{\mathbf{z}}))\frac{\partial \hat{\mathbf{z}}}{\partial \theta}\right], \tag{7}$$

where $\hat{\mathbf{z}}$ is the one-step causal output. The active style LoRA is kept frozen and sampled dynamically, so a single distilled backbone remains compatible with the bank.

4.6 Stage 4: Rollout fine-tuning

Stage 3 already samples student-generated histories, but its distribution matching objective does not directly supervise trajectory-level deviations from the teacher. These deviations can accumulate into brightness oscillation, blur, or drift. We fine-tune the student on short autoregressive rollouts against fixed-length multi-step teacher targets, which can also supply an optional clean prefix; the one-step student then generates every subsequent latent from its own history. Training replays each generated-history state with gradients and uses endpoint, pooled low-frequency, DC, and temporal-delta supervision. Each component below uses mean smooth-L_1 loss, a scaled Huber penalty (Huber, 1964), SL1_β with $\beta = 0.05$ on its respective residual:

$$\mathcal{L}_{\text{roll}} = \mathcal{L}_{\text{end}} + 0.2\mathcal{L}_{\text{low}} + 0.1\mathcal{L}_{\text{DC}} + 0.15\mathcal{L}_{\Delta}, \tag{8}$$

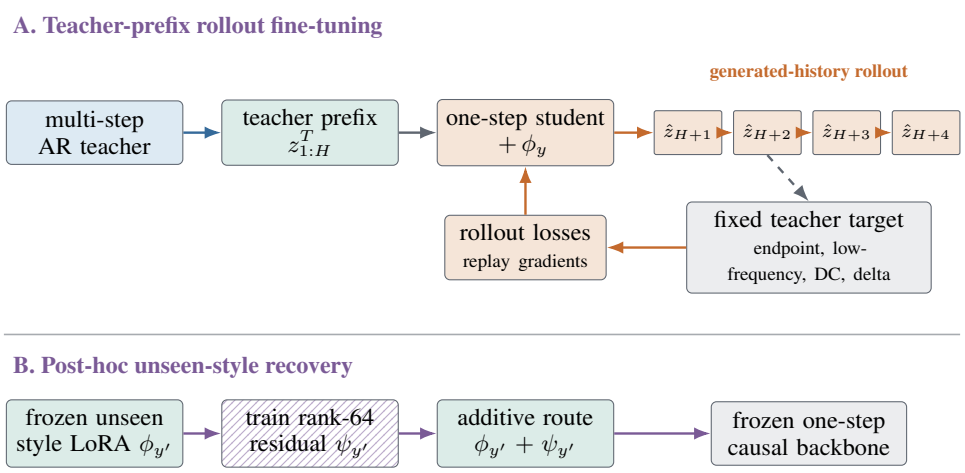

Figure 4: **Teacher-prefix rollout and unseen-style recovery.** An optional teacher prefix initializes the bounded cache; the one-step student then rolls out on its own generated states. The no-gradient rollout is replayed frame by frame and supervised by fixed multi-step teacher targets. For a style unseen during distillation, recovery freezes both its existing style LoRA and the causal backbone, and updates only an additive residual LoRA.

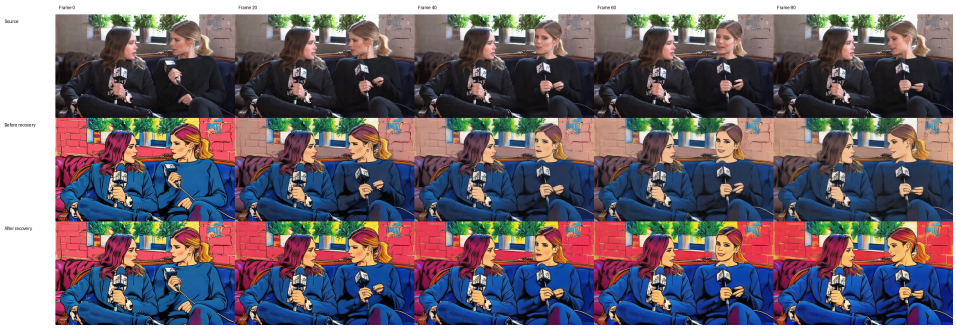

Figure 5: Recovery for an unseen Pop Art style. Matched frames compare the same frozen style LoRA before and after 100 recovery updates. The adapted rollout shows more consistent stylization.

where the first three terms compare the predicted and teacher latents at each supervised position ($\mathcal{L}_{\text{end}}$), respectively before pooling, after 8×8 spatial average pooling ($\mathcal{L}_{\text{low}}$), and after spatial averaging to per-channel means ($\mathcal{L}_{\text{DC}}$). For $\mathcal{H} = \{1, 2, 4, 8, 16\}$, $\mathcal{L}_\Delta$ averages SL1_β between the predicted displacement from a generated latent $\hat{z}_i - \text{sg}(\hat{z}_{i-h})$ and the corresponding teacher displacement $z_i^T - z_{i-h}^T$ over valid $h \in \mathcal{H}$. The generated history is detached in this comparison; the loss is averaged over positions after the optional clean prefix. The rollout itself is generated without gradients, then replayed one frame at a time for optimization; history states are detached. This exposes the student to its own errors without backpropagating through an entire long stream.

4.7 Recovery for unseen style LoRAs

A LoRA trained after Stages 2–3 may be individually valid yet incompatible with the distilled causal backbone, because the latter has never observed its activation statistics. Given a new style LoRA $\phi_{y'}$, we generate a small set of bidirectional teacher trajectories, freeze both the causal backbone and $\phi_{y'}$, and train an additive rank-64 recovery LoRA $\psi_{y'}$. The combined route $\phi_{y'} + \psi_{y'}$ is evaluated on one-step autoregressive rollouts. The implemented recovery loss reuses the four-term latent supervision from Equation (8):

$$\mathcal{L}_{\text{rec}} = \mathcal{L}_{\text{end}} + 0.2\mathcal{L}_{\text{low}} + 0.1\mathcal{L}_{\text{DC}} + 0.15\mathcal{L}_\Delta. \tag{9}$$

This preserves the low storage and deployment cost of LoRAs while calibrating the new route to one-step autoregressive inference.

Table 1: Quality and runtime comparison. Quality uses 47 complete videos across five styles. Runtime uses exclusive-H100, host-preloaded 401-frame V2V profiles at each method's native codec and resolution; latency is median per indicated output chunk. SANA's whole-clip throughput and memory are not bounded-stream measurements. Full protocols are in Appendix B.1 and Appendix Table 8.

Method	DINOv2 ↑	CLIP-T ↑	CLIP-I ↑	tLPIPS ↓	Flow EPE ↓
StreamV2V (1 step)	0.773	0.202	0.878	0.0219	0.274
StreamDiffusionV2 (2 steps)	0.470	0.219	0.703	0.0153	0.435
SANA-Streaming (4 steps)	0.426	0.276	0.704	0.0275	0.319
JoyAI-Video-Edit (1 step)	0.590	0.273	0.743	0.0283	0.252
RSF H1 (1 step)	0.688	0.256	0.792	0.0179	0.210

Method	FPS ↑	Latency (ms)↓	Peak GiB ↓
StreamV2V (1 step)	7.80	125	4.67
StreamDiffusionV2 (2 steps)	22.49	175	23.23
SANA-Streaming (4 steps)	20.08	—	29.14
JoyAI-Video-Edit (1 step)	23.44	1,046	25.13
RSF H1 (1 step)	101.73	79	4.81

4.8 Streaming inference

At runtime the source encoder constructs $\mathbf{x}_s$ for the newest frame, the one-step student reads a rolling cache of at most H clear generated latents, and the decoder immediately emits the styled frame. Text embeddings and style LoRA weights are cached. We decompose total latency into source encoding, DiT inference, cache update, and decoding in Section 5.4.

5 Experiments

5.1 Setup

Implementation. We train style LoRAs on aligned ReCo-Data pairs (Zhang et al., 2025) and evaluate one-step causal Wan2.1 (Wan Team et al., 2025) transfer on Pexels videos (Pexels, 2026). Training and inference configurations are given in Appendix A.

Baselines. We compare with four released video stylization systems on the same source videos and requested styles. In particular, SANA-Streaming is purpose-built for real-time V2V editing (Zhao et al., 2026), while JoyAI-Video-Edit is a recent state-of-the-art real-time streaming editor (Xiao et al., 2026). The paired results use only complete outputs; native resolutions and style conditioning differ across methods (Appendix B.1).

Metrics. We assess target-style alignment (CLIP-T), source-content preservation (CLIP-I and DINOv2), and temporal behavior (flow-warped tLPIPS and motion EPE). Definitions and aggregation are in Appendix B.1.

5.2 Comparison with prior work

RSF provides stronger style-text alignment than the image-oriented baselines, while the more strongly stylized video baselines retain less source content. It has the best source-relative motion agreement and low warped perceptual error, although StreamDiffusionV2 scores slightly better on the latter. No method dominates every style, content, and continuity metric; per-style scores and paired intervals appear in the appendix. Figure 6 illustrates the methods across four styles.

5.3 Human evaluation

We invited three independent human raters for human preference evaluation. We constructed a web-based comparison platform (see Appendix Figure 9). The raters were asked to decide their preference for two different stylized videos which are transferred from one original video clip with

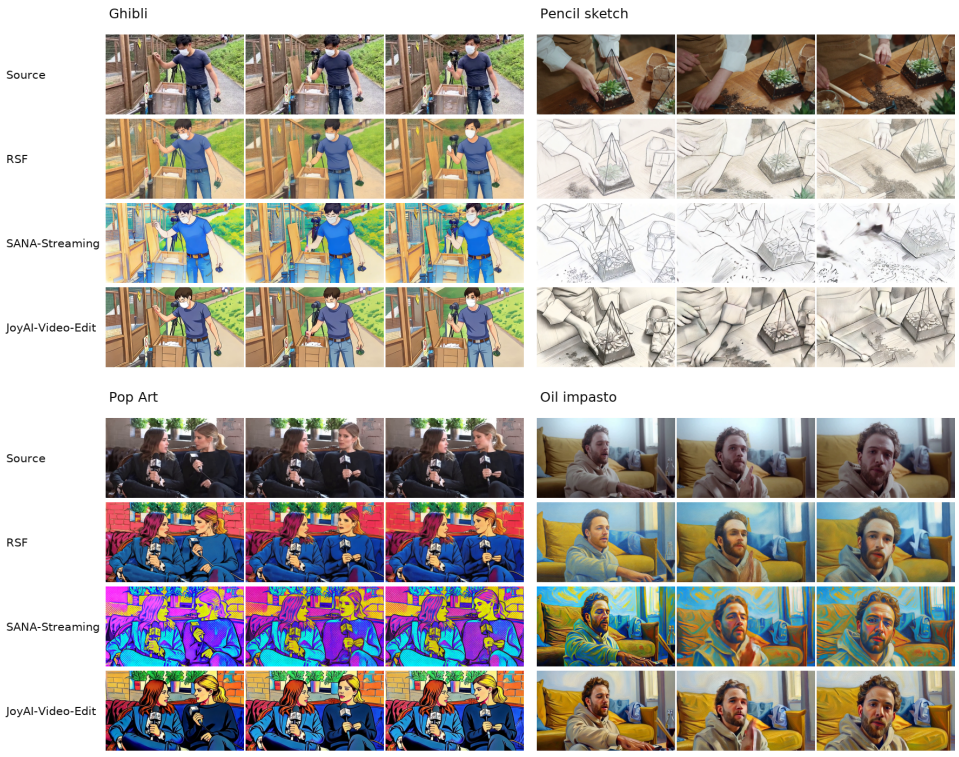

Figure 6: Four styles and three streaming editors on matched source frames. The upper panels show Ghibli and pencil sketch, and the lower panels show Pop Art and oil impasto. Within each panel, rows show the source, RSF, SANA-Streaming, and JoyAI-Video-Edit, and columns show the first, middle, and last frame of the same clip.

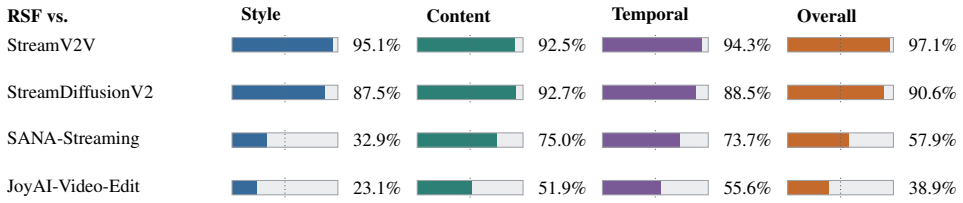

Figure 7: RSF win share against each baseline (wins plus half ties). Dotted marks indicate 50%;

two randomly picked methods. The raters need to make decision for three different domains (Style Strength, Content Preservation and Temporal Smoothness) first and then decide which one is better overall. Figure 7 visualizes the observed win shares. It can be noted that RSF beats StreamV2V, StreamDiffusionV2 and SANA-Streaming. Against JoyAI-Video-Edit, RSF is preferred slightly more often on content preservation and temporal smoothness, but less often on style strength and overall preference.

5.4 Streaming efficiency

On one H100, the mixed-FP8 H1 profile has the highest measured throughput among the tested systems while retaining a low GPU footprint (Table 1). On an RTX 5090, it reaches 36.3 FPS with approximately 220 ms latency. Native baseline profiles are in Appendix Table 8. Figure 8 shows Ghibli-style output on a continuous 2,000-frame person-centered video. The stylized appearance persists through the subject's hand and head movements.

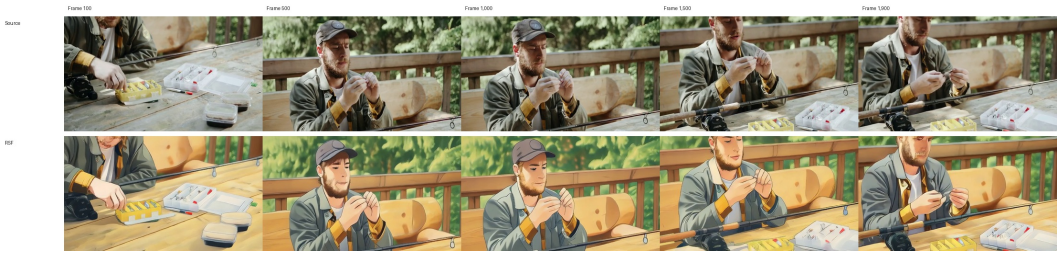

Figure 8: Ghibli-style stylization across a continuous 2,000-frame close-up of a person preparing fishing tackle. Corresponding frames show the source and the streaming RSF result.

Table 2: Ablations grouped by matched evaluation set.

Variant	CLIP-T ↑	DINOv2-I ↑	tLPIPS ↓	Flow EPE ↓
History length				
H0	0.2516	0.5621	0.0581	0.3434
H1	0.2557	0.6879	0.0179	0.2093
H2	0.2587	0.6863	0.0178	0.1910
H4	0.2566	0.6957	0.0188	0.1953
H1 rollout				
Before rollout	0.2561	0.6560	0.02776	0.2396
After rollout	0.2557	0.6879	0.0179	0.2093
Unseen Pop Art style				
Without recovery	0.2164	0.7664	0.0206	0.1869
With recovery	0.2556	0.6760	0.0221	0.2131

5.5 Ablations

The reconstruction-only source-corruption sweep shows a trade-off: stronger corruption makes the velocity easier to fit but worsens held-out source reconstruction (Appendix Table 4). We use $s = 0.4$ as a working compromise.

Table 2 evaluates the effect of causal history length, where H0 denotes inference without generated-history conditioning. A single latent of history (H1) already provides a large improvement in source correspondence and temporal consistency over H0, while longer histories provide smaller additional gains. Appendix Table. 9 reports the associated runtime cost. Rollout fine-tuning further improves temporal consistency: removing Stage 4 increases tLPIPS from 0.0179 to 0.0278 and Flow EPE from 0.2093 to 0.2396. Figure 10 provides a qualitative example. For the previously unseen Pop Art style, recovery increases CLIP-T from 0.2164 to 0.2556, indicating substantially stronger style alignment. This improvement comes with reduced source correspondence and moderately worse source-relative temporal metrics, illustrating a style–fidelity trade-off.

6 Conclusion

We presented RSF, a staged approach to streaming video stylization that transports an informative intermediate source distribution to a target style. Its design separates modular style learning, reusable causal dynamics, one-step distillation, and generated-history adaptation. Recovery fine-tuning extends the distilled backbone to new LoRAs without repeating full video training.

Limitations. The short style-flow path favors appearance changes that preserve source geometry; styles requiring substantial shape changes can remain too close to the input. Each new style still requires a trained LoRA and aligned video pairs. Our smallest represented style has 656 pairs, so roughly 600 pairs is a practical demonstrated data scale. Rank-128, 81-frame LoRA training also needs substantial GPU memory (about 15 GiB peak PyTorch reservation in a matched run; Appendix A).

AI USE STATEMENT

In this work, we used generative AI tools to assist with research-methodology and experiment-design discussions, software implementation and debugging, data-processing workflows, exploratory interpretation of experimental results, literature search and summarization, and the organization and editing of the manuscript. The authors reviewed the resulting research decisions, verified AI-assisted code and experimental outputs, and take responsibility for the final text, claims, code, and results.

REPRODUCIBILITY STATEMENT

Implementation and training configurations are specified in Appendix A; metric definitions and the matched evaluation protocol are given in Appendix B.1. Additional quantitative results appear in Appendix B. We intend to release the code, model weights, and evaluation manifests.

REFERENCES

Tianrui Feng, Zhi Li, Shuo Yang, Haocheng Xi, Muyang Li, Xiuyu Li, Lvmin Zhang, Keting Yang, Kelly Peng, Song Han, Maneesh Agrawala, Kurt Keutzer, Akio Kodaira, and Chenfeng Xu. Streamdiffusionv2: A streaming system for dynamic and interactive video generation. *Proceedings of Machine Learning and Systems*, 8:1656–1669, 2026. URL https://github.com/chenfengxu714/StreamDiffusionV2.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2106.09685.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2506.08009. Spotlight.

Peter J. Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964. doi: 10.1214/aoms/1177703732. URL https://doi.org/10.1214/aoms/1177703732.

Feng Liang, Akio Kodaira, Chenfeng Xu, Masayoshi Tomizuka, Kurt Keutzer, and Diana Marculescu. Looking backward: Streaming video-to-video translation with feature banks. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2405.15757.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *ArXiv*, abs/2210.02747, 2022. URL https://api.semanticscholar.org/CorpusID:252734897.

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.

Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2022.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. URL https://arxiv.org/abs/2304.07193.

Pexels. Pexels videos. Online media platform, 2026. URL https://www.pexels.com/videos/. Accessed 2026-09-23; usage governed by the Pexels license at https://www.pexels.com/license/.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763, 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Zachary Teed and Jia Deng. RAFT: Recurrent all-pairs field transforms for optical flow. In *European Conference on Computer Vision*, pp. 402–419, 2020. URL https://www.ecva.net/papers/eccv_2020/papers_ECCV/html/3526_ECCV_2020_paper.php.

Wan Team, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Chen, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. URL https://arxiv.org/abs/2503.20314.

Xinyu Wang, Chongbo Zhao, Fangneng Zhan, and Yue Ma. Liveedit: Towards real-time diffusion-based streaming video editing. *arXiv preprint arXiv:2606.26740*, 2026. URL https://arxiv.org/abs/2606.26740.

Yicheng Xiao, Wenxun Dai, Xinran Qin, Lin Song, Maoquan Zhang, Hang Xu, Yukang Chen, Yitong Li, Guohui Zhang, Yuan Zhang, Xuying Zhang, Tommy Zhang, Jianlong Yuan, Peihao Li, Shuai Lu, Siming Fu, Chuyang Zhao, Xin Han, Jie Huang, Wenbo Li, Guoqing Ma, Wei Huang, Xiaojuan Qi, Haoyang Huang, and Nan Duan. Joyai-video-edit: Real-time open-ended video editing with autoregressive diffusion. *arXiv preprint arXiv:2608.03974*, 2026. URL https://arxiv.org/abs/2608.03974.

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

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Frédo Durand, William T. Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024b. URL https://arxiv.org/abs/2311.18828.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 2018. URL https://openaccess.thecvf.com/content_cvpr_2018/html/Zhang_The_Unreasonable_Effectiveness_CVPR_2018_paper.html.

Zhongwei Zhang, Fuchen Long, Wei Li, Zhaofan Qiu, Wu Liu, Ting Yao, and Tao Mei. Region-constraint in-context generation for instructional video editing, 2025. URL https://arxiv.org/abs/2512.17650.

Yuyang Zhao, Yicheng Pan, Qiyuan He, Jincheng Yu, Junsong Chen, Tian Ye, Haozhe Liu, Enze Xie, and Song Han. Sana-streaming: Real-time streaming video editing with hybrid diffusion transformer. *arXiv preprint arXiv:2605.30409*, 2026. URL https://arxiv.org/abs/2605.30409.

A Implementation Details

In Stage 4 and new-style recovery, every endpoint, low-frequency, DC, and temporal-delta term is an elementwise smooth-L_1 loss averaged over its tensor. For a scalar residual r, $\mathrm{SL1}_\beta(r) = r^2/(2\beta)$ when $|r| < \beta$ and $|r| - \beta/2$ otherwise, with $\beta = 0.05$. Thus β is the transition threshold between quadratic and linear behavior, not an extra loss weight. We use the 1.3B-parameter Wan2.1 video DiT and its native causal video VAE. The evaluated H1 quality run uses source corruption $s = 0.4$, frozen rank-128 style LoRAs, one denoising step, and CFG 1.0. Stage 1 targets up to 5,000 single-GPU updates per style LoRA on 81-frame clips; the retained checkpoints are listed in Table 3. Stage 2 uses 1,767 recorded bidirectional teacher trajectories for 12,000 full-DiT updates on single-GPU, with clear/noisy latents interleaved under a causal mask. The one-step DMD stage and the subsequent rollout stage update the shared backbone while keeping the style bank frozen. The reported Stage-3 run uses a non-GAN DMD surrogate with a fake-score flow-matching update; its Stage-2 causal teacher initializes the student, but the generator loss has no separate regression term against teacher outputs. A prespecified matched rollout comparison uses 160 fixed teacher trajectories and 500 updates. New-style recovery adds a rank-64 residual LoRA. Non-distilled comparison runs use 25 denoising steps and CFG 1.0 unless a baseline's released configuration requires otherwise.

Our evaluated H1 checkpoint uses 832×480 RGB input. The Wan2.1 video VAE represents 81 RGB frames as 21 latent time positions of shape $16 \times 60 \times 104$. Stage-4 training uses a one-latent rollout chunk and one generated latent of causal history. For 70% of optimizer steps, one clean teacher latent initializes the stream and the student rolls out the remaining 20 positions; the other 30% use no teacher prefix and roll out all 21. The fixed multi-step teacher uses 30 denoising evaluations for this stage. The five style LoRAs remain frozen, while the shared DiT is updated. The Stage-4 learning rate is 10^{-6} with 200 warm-up steps and global gradient clipping at 1.0; this run samples source videos from the ReCo domain. The evaluated H1 inference setting uses one denoising step, CFG 1.0, $s = 0.4$.

Stage-1 style bank. The five LoRAs frozen in the H1 model are rank 128. Each is trained on 81-frame clips at $s = 0.4$. It uses learning rate 2×10^{-5}, 100 warm-up steps, gradient clipping at 1.0, 10% text-prompt dropout, and the native Wan VAE. It samples $u \sim \mathcal{U}(0, 1)$, applies unweighted mean squared velocity error, and passes $1000su$ to the Wan timestep interface; $t = su$ in the main text is the normalized time. The single-GPU rank-128, 81-frame trainer uses gradient checkpointing. An 832×480 run logged a 13.85 GiB peak PyTorch allocation and 14.85 GiB peak reservation over 5,000 updates. These are framework measurements on an H100: 16 GiB leaves little room for non-PyTorch allocations, while 24 GiB is a prudent training budget. The smallest style in the frozen bank contains 656 paired videos from 608 distinct sources; this supports an approximately 600-pair demonstrated scale. The fixed-160 Stage-4 cache likewise has 160 readable records from 45 distinct ReCo source paths.

Table 3: ReCo manifest coverage for the five active style LoRAs. The second column counts paired videos; the third counts distinct source-video keys within that style.

Style	Paired videos	Distinct sources
Ghibli	1,586	1,338
Oil impasto	3,266	2,767
Pencil sketch	17,744	10,648
Watercolor	9,532	7,479
Cel-shaded anime	656	608

Stage-2 causal teacher. Each cached teacher sample stores only the original source latent video and the final bidirectional stylized latent video, not the intermediate ODE states. At each optimizer update, the trainer selects an 11-latent window, draws independent Gaussian noise and a uniform local time for each of the 10 predicted latents, and reconstructs the $s = 0.4$ bridge and its velocity target. Each context latent is independently blended with Gaussian noise at a sampled corruption level in $[0, 0.05]$; its own timestep is passed to the model. A single style-neutral instruction is shared

across routes, while the selected style LoRA stays frozen. Only the 10 noisy query positions enter the unweighted velocity MSE.

B Additional Quantitative Results

B.1 Evaluation protocol and metric definitions

The frozen Pexels test manifest contains 50 clips from 50 distinct original video IDs. All methods receive the same source clips and target-style semantics, but retain their released checkpoints and native inference resolutions. Some public runners produce more than the requested 81 frames; we score the first 81. The strictly paired set excludes an entire source video if any method has a missing or shorter output in any style.

Generation prompts are not text-identical. H1 uses a fixed preservation instruction and selects the style through its frozen LoRA. StreamV2V, StreamDiffusionV2, and SANA-Streaming share a detailed per-style instruction; JoyAI-Video-Edit requires a shorter one.

CLIP-T (Radford et al., 2021) is the mean cosine similarity between OpenAI CLIP ViT-B/32 embeddings of generated frames and the same style-only text for every method. CLIP-I is cosine similarity between corresponding source/output CLIP image embeddings. Both use the first 81 frames and the standard 224-pixel CLIP crop. DINOv2-small content similarity (Oquab et al., 2024) compares normalized final-layer class tokens after the official 256-pixel resize and 224×224 center crop. We average frames within each video, then videos within each style, then macro-average styles. CLIP-T measures text alignment, not content; CLIP-I and DINOv2 measure feature correspondence, not geometric or facial identity. An identity mapping can score well on content and motion while providing no stylization, so these axes must be interpreted jointly.

Our primary temporal diagnostic estimates forward and backward RAFT-small flow (Teed & Deng, 2020) on adjacent *source* frames. At a 256×144 common grid, we backward-warp the preceding output frame and compare it to the current output with LPIPS-Alex (Zhang et al., 2018). Out-of-bounds and forward–backward-inconsistent pixels are rejected and filled from the current output before LPIPS; the same valid pixels give a masked RGB warp-L_1 diagnostic. Warped tLPIPS averages all 80 adjacent pairs within each video. An independent RAFT-small flow estimate on output pairs yields source/output endpoint error (EPE) on the valid source pixels. Pair means are averaged within videos and then across videos and styles. We also report output/source flow magnitude and stratify by source motion: frozen output can attain low warp error yet has flow EPE equal to source-flow magnitude. Large geometric edits can invalidate source-flow correspondence, so no single automatic metric replaces human assessment.

Human study protocol . Raters view synchronized videos with balanced left/right placement (two randomly picked methods for blinded comparison) and judge style match, content preservation, temporal stability, and overall preference, allowing ties. Among the 10 possible unordered method pairs, four contain RSF H1. For each source video, one of the five styles are sampled for each rater. Each rater therefore evaluates $10 \times 47 = 470$ paired comparisons, of which $4 \times 47 = 188$ involve RSF H1 and contribute to the reported RSF win-plus-half-tie rates.

Figure 9 shows the blinded annotation interface. The source is visible alongside anonymized outputs A and B; raters can choose either output or a tie for each of the four questions.

DINOv2 content correspondence. We use the frozen DINOv2-small checkpoint. Its standard processor resizes the shorter side to 256 pixels and center-crops 224×224. We compare L_2-normalized final class-token features at corresponding frame indices over the first 81 frames.

Flow-warped temporal diagnostic. The RAFT-small source-flow protocol uses the first 81 frames, all 80 adjacent pairs, a 256×144 evaluation grid, forward–backward consistency and out-of-bounds masks, and LPIPS-Alex on the same strict 47-video set.

JoyAI step count. The released JoyAI-Video-Edit deployment defaults to two denoising steps, whereas the realtime table uses one. On the same 47-video subset, changing only the step count gives macro CLIP-T 0.273 at one step versus 0.269 at two steps. The paired two-minus-one difference

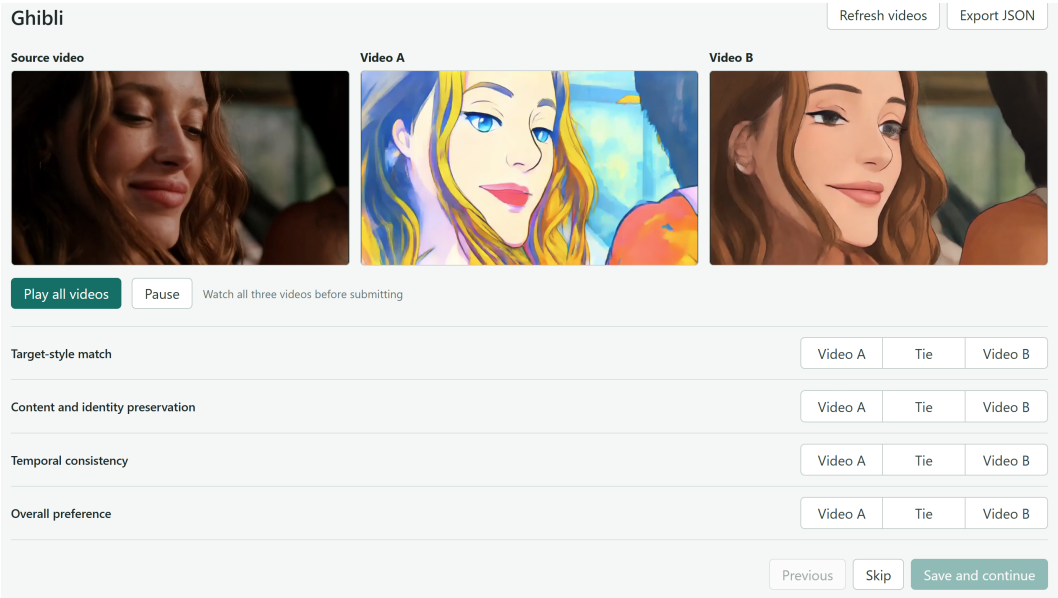

Figure 9: Rater interface for paired video preference judgments. The screen shows the source, anonymized alternatives, synchronized playback controls, and separate style, content, temporal, and overall choices.

Table 4: Source corruption ablation for reconstruction. Six independent rank-128, 17-frame LoRAs were trained at 832×480 for 5,000 updates on the same ReCo source domain. RGB MSE and PSNR use the same eight 17-frame ReCo clips, with 25 denoising steps, CFG 1.0, and matched seeds. Lower MSE is better.

s	Train velocity MSE ↓	RGB MSE ↓	PSNR ↑ (dB)
0.1	0.619	0.000513	33.48
0.2	0.447	0.000919	31.04
0.3	0.357	0.001429	29.16
0.4	0.302	0.002067	27.54
0.5	0.264	0.002914	25.98
0.6	0.236	0.004136	24.39

is -0.0035 with a video-level bootstrap 95% interval $[-0.0053, -0.0019]$. This small decrease concerns text-style alignment; we retain the two-step run as an official-quality control rather than silently replacing the realtime run. On 235 matched video-style cases, the median client-reported end-to-end throughput changes from 19.46 to 16.65 FPS and median first-output latency from 1.406 to 1.481 seconds. These are within-JoyAI comparisons only; its chunked client/server timing is not directly comparable with other methods' DiT-only or subprocess timing.

StreamV2V step count. The realtime StreamV2V row uses one denoising step; we also ran a four-step quality control on all five styles and 50 source videos, holding source noise strength at 0.4 and feature injection at 0.8. On the strict 47-video paired subset (235 video–style cells), macro CLIP-T rises from 0.2015 to 0.2110. The video-bootstrap four-minus-one difference is $+0.0094$ (95% interval $[+0.0065, +0.0124]$). However, source/output CLIP-I falls from 0.8783 to 0.8260, a paired difference of -0.0522 (95% interval $[-0.0607, -0.0441]$). Thus extra denoising increases measured style-text alignment at a clear content-preservation cost.

Native-runtime diagnostic profiles. Table 8 records available GPU-exclusive H100 measurements, not a cross-method latency ranking. All peak-memory entries are device-wide 250-ms `nvidia-smi` samples, but SANA's whole-clip pipeline retains the full video and is not a bounded online-memory

Table 5: CLIP-T by requested style. Each entry averages the first 81 output frames over the same 47 complete videos. We use one frozen style-only text per row and OpenAI CLIP ViT-B/32 for every method. Higher is better for this text-alignment measure.

Style	StreamV2V	StreamDiffusionV2	SANA-Streaming	JoyAI	RSF H1
Ghibli	0.224	0.241	0.285	0.278	0.271
Oil impasto	0.205	0.212	0.262	0.250	0.255
Pencil sketch	0.228	0.233	0.307	0.305	0.303
Watercolor	0.190	0.201	0.259	0.247	0.233
Cel-shaded anime	0.161	0.208	0.265	0.282	0.217
Macro average	0.202	0.219	0.276	0.273	0.256

Table 6: CLIP-I by requested style on the same 47 complete videos and first 81 frames as Table 5. Each score is the mean cosine similarity between corresponding source and output CLIP image embeddings. Higher indicates stronger coarse source correspondence.

Style	StreamV2V	StreamDiffusionV2	SANA-Streaming	JoyAI	RSF H1
Ghibli	0.866	0.693	0.719	0.737	0.809
Oil impasto	0.888	0.710	0.690	0.759	0.743
Pencil sketch	0.851	0.702	0.707	0.718	0.767
Watercolor	0.895	0.724	0.687	0.760	0.811
Cel-shaded anime	0.891	0.687	0.720	0.738	0.831
Macro average	0.878	0.703	0.704	0.743	0.792

Table 7: DINOv2-small source–output frame-aligned cosine on the same strict 47-video, five-style set (higher is better).

Style	StreamV2V	StreamDiffusionV2	SANA-Streaming	JoyAI	RSF H1
Ghibli	0.747	0.459	0.486	0.579	0.741
Oil impasto	0.801	0.473	0.441	0.660	0.699
Pencil sketch	0.706	0.454	0.340	0.515	0.509
Watercolor	0.811	0.503	0.355	0.627	0.729
Cel-shaded anime	0.801	0.461	0.510	0.568	0.760
Macro average	0.773	0.470	0.426	0.590	0.688

profile. StreamV2V executes frame by frame with the native SD image VAE. StreamDiffusionV2 uses TAEHV and its bounded session API; the 396 delivered frames reflect startup delay. The two-step profile uses four-frame chunks. JoyAI uses an eight-frame-chunk WebSocket service with at most 24 outstanding source frames, FP8 DiT/text weights, and CPU-offloaded text encoding; its xVAE stays on GPU. For these chunked methods, the latency cell measures input submission to *chunk completion*, not individual-frame display. SANA uses an LTX2VAE with eightfold temporal compression and its native 1280×704 V2V path; the source VAE encodes all 401 frames before sampling, then decodes the whole output. Its 20.08 FPS (mean of two warmed 401-frame runs, including RGB conversion but not MP4 encoding) is offline whole-clip throughput, so an online latency is not available. We rebuilt the mixed-FP8 H1 backend and measured 101.73 RGB FPS, 78.76-ms median input-to-host-output latency, and 4.81 GiB steady device memory. Its 7.52-GiB startup peak includes a first-frame engine that is released before steady streaming; the steady 4.81-GiB plateau is also visible in 250-ms `nvidia-smi` samples.

C Additional Qualitative Results

Table 8: H100 V2V runtime profiles on the same 401-frame source. c_t is codec temporal compression. All rows exclude model loading, prompt encoding, and video-file I/O; they include source transfer, model execution, target decoding, and host output materialization. Online FPS excludes initial warm-up; SANA's whole-clip FPS includes its first chunk and has no online frame latency. The RSF H1 row uses the retained step-4000 checkpoint; its GPU memory is the post-startup steady peak, excluding a one-time first-frame engine that is released before streaming.

Method	Codec	c_t	Resolution	FP8	Steps	GPU modules	Latency	FPS	Peak GiB
StreamV2V (speed)	SD VAE	1	832×480	No	1	UNet, VAE, CLIP	125 ms/frame	7.80	4.67
StreamV2V (quality)	SD VAE	1	832×480	No	4	UNet, VAE, CLIP	146 ms/frame	6.84	6.50
StreamDiffusionV2	TAEHV	4	832×480	No	2	DiT, TAE, T5	175 ms/4f	22.49	23.23
JoyAI-Video-Edit	xVAE	8	840×480	Yes	1	DiT, xVAE	1046 ms/8f	23.44	25.13
SANA-Streaming	LTX2VAE	8	1280×704	No	4	DiT, VAE, Gemma	N/A	20.08	29.14
RSF H1	TAE-Wan	4	832×480	Mixed	1	DiT, TAE	79 ms/4f	101.73	4.81

Table 9: Streaming efficiency versus history length on one H100 at 832×480, using one-step CFG1 inference, TensorRT FFN-only FP8, and FP16 TAE. FPS includes host transfers, encoding, DiT/cache updates, and decoding. Chunk service is the mean time per four output frames; frame latency is the median inferred from chunk timings and TAE temporal alignment under saturated input. Memory is steady resident usage, not model-loading peak.

History	FPS ↑	Chunk service (ms / 4 frames)	Frame latency (ms)	Resident memory (GiB)
H0	161.70	24.74	49.44	4.72
H1	101.94	39.24	78.41	4.79
H2	83.10	48.14	96.20	6.19
H4	64.77	61.76	123.45	8.26



Figure 10: Effect of Stage-4 rollout finetuning on facial appearance. Five corresponding frames show the source (top), the one-step model without Stage 4 (middle), and the rollout-finetuned model (bottom). Without rollout finetuning, facial shape and appearance vary more across this clip.



Figure 11: Person-centric Ghibli example: source and five stylization methods at corresponding frame indices.

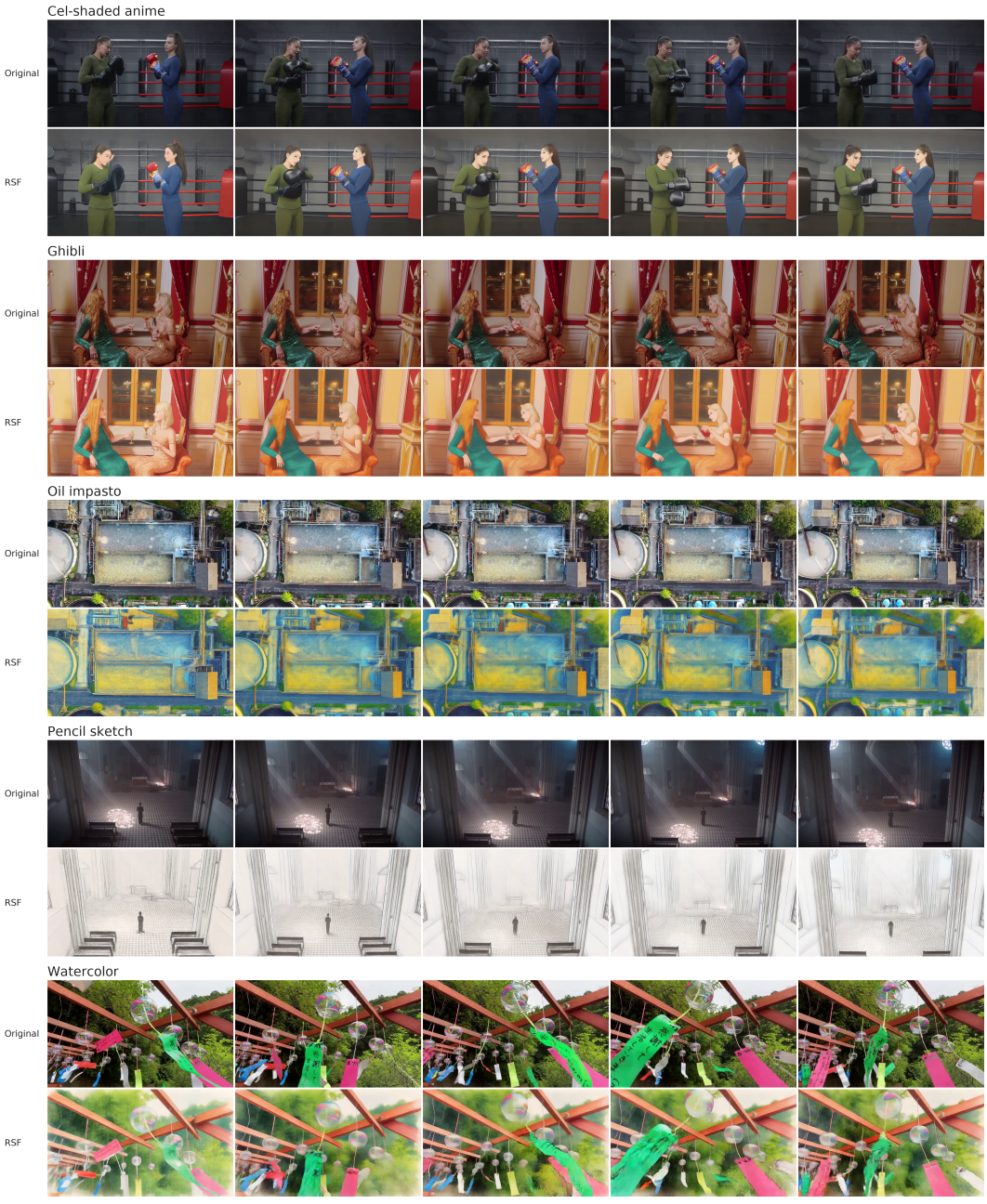

Figure 12: Additional random RSF examples across five styles. Each pair of rows shows five corresponding original frames (top) and stylized frames (bottom), uniformly spaced across the clip.