RESILIENT STREAMING VIDEO EDITING

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing streaming video editors can exhibit **ruminative** behavior, over-relying on historical frames even as errors accumulate in the generation history. To address this limitation, we present JoyAI-Video-Edit, a **resilient** 16B-parameter autoregressive diffusion model for source-faithful, low-latency streaming editing. Specifically, we introduce Resilient-DMD, which selectively strengthens teacher-side source guidance during distillation when relative source attention falls below matched baselines, yielding a two-step generator. Long-Horizon Autoregressive Distillation (LHAD) extends this adaptive supervision to deeper rollout states while bounding activation memory. Our streaming engine combines bounded transformer state, single-frame VAE state handoff, and asynchronous execution to reduce end-to-end overhead. To support training and sustained-stream evaluation, we construct an editing data pipeline and introduce LongV2VBench, built on one-minute videos. Automatic and human evaluations show that JoyAI-Video-Edit outperforms the evaluated streaming baselines and remains competitive with strong offline editors. The system delivers end-to-end 720p editing at 34.87 FPS on a single B200 GPU, with support for RTX 5090 and RTX PRO 6000.

1 INTRODUCTION

Instruction-guided video editing changes objects, appearances, backgrounds, and styles while preserving source motion and scene structure (Qi et al., 2023; Jiang et al., 2025; Ye et al., 2025; DecartAI Team, 2025; Liao et al., 2025). Live applications require these edits as source frames arrive. Offline editors rely on future frames and iterative denoising, whereas recent streaming editors use causal generation and cached computation to produce each chunk from the current source and past outputs (Wang et al., 2026; Zhao et al., 2026). These advances make online editing possible, but sustaining it over streams of unknown duration requires source-faithful, temporally consistent outputs with low latency and bounded memory.

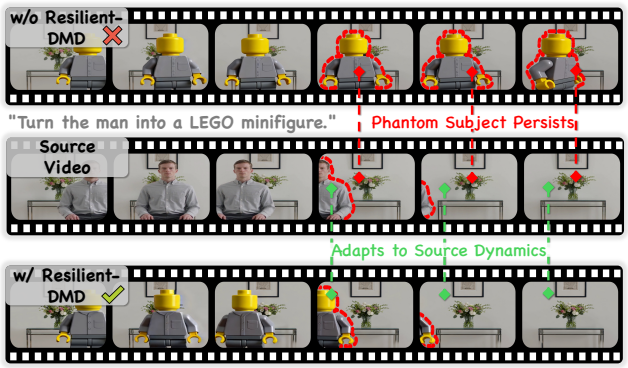

Figure 1: Without **Resilient-DMD**, the model returns **ruminative** predictions in which a phantom subject persists after the man exits the source video. In contrast, Resilient-DMD presents **resilient** results, remaining faithful to the current source despite errors in generated history.

Meeting these requirements poses two key design challenges. *First, the editor must use generated history for continuity while remaining responsive to the current source.* Supervised training can favor copying similar content from preceding target frames because it reduces prediction loss. At inference, however, overreliance on generated history can propagate errors and prevent the editor from following source changes (Huang et al., 2025; Guo et al., 2025). In Figure 1, an edited subject remains visible after leaving the source video. Stronger source guidance can counter this drift, but applying it uniformly may constrain edits that are already well grounded. *Second, a streaming editor must control latency across the entire pipeline.* Source encoding, target decoding, cache updates,

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.

and I/O all add per-chunk latency. Sustained editing requires coordinated execution and temporal state that remains bounded as the stream grows.

We present JoyAI-Video-Edit, a 16B-parameter autoregressive diffusion editor. To maintain source fidelity, we introduce Resilient-DMD, which adapts teacher-side source guidance during distillation. It compares the generator's attention to source and history tokens with moving baselines matched by layer, noise level, and rollout depth. When relative source attention falls, the teacher increases its source-conditioned correction for that chunk. The generator thus receives stronger source supervision where relative source reliance weakens, while retaining editing flexibility elsewhere. This control is used only during training, leaving a two-step generator without inference-time monitoring or extra guidance branches. We further introduce Long-Horizon Autoregressive Distillation (LHAD) to bring adaptive supervision to deeper generated states. LHAD divides extended rollouts into segments, reuses detached history, and accumulates gradients before an optimizer update. This enables longer training rollouts with bounded activation memory, exposing the editor to its own accumulated prediction errors and helping mitigate exposure bias.

For efficient deployment, we design a bounded-state asynchronous engine. A fixed transformer history window and static KV pool bound temporal memory; FP8 linear layers, fused operators, and CUDA graphs reduce execution overhead. A single boundary frame carries VAE context between chunks, enabling fixed-shape encoding and decoding. The engine overlaps VAE processing, DiT denoising, and I/O where dependencies allow, reducing exposed end-to-end latency. To support training and sustained-stream evaluation, we develop a paired-data pipeline and introduce LongV2VBench. The pipeline combines keyframe-guided edit propagation with latent-sharing video generation, followed by quality filtering and MLLM-based instruction refinement. LongV2VBench comprises 229 one-minute editing instances across five global and local editing categories, assessing whether edits remain consistent and aligned with the evolving source.

Quantitative evaluations on OpenVE-Bench and LongV2VBench demonstrate that JoyAI-Video-Edit consistently outperforms the evaluated streaming baselines across all five editing categories. It matches the best open-source offline overall score on OpenVE-Bench and achieves 81–90% win rates in human comparisons against streaming baselines. Long-video examples show sustained edits, and ablations demonstrate complementary gains from Resilient-DMD and LHAD. The system achieves end-to-end 720p editing at 34.87 FPS on a single NVIDIA B200 GPU. It also supports deployment on NVIDIA GeForce RTX 5090 and RTX PRO 6000 GPUs.

2 Related Work

2.1 Video Editing

Instruction-guided video editors support diverse transformations through text and visual conditioning (Chen et al., 2026; Lin et al., 2026; He et al., 2025), but fixed-clip processing with bidirectional temporal interactions limits online deployment. Streaming methods instead use causal or incremental computation: StreamDiffusionV2 enables interactive generation (Feng et al., 2025), SANA-Streaming combines streaming V2V modeling with long-video training (Zhao et al., 2026), and LiveEdit adapts offline editors through causal modeling and cached computation (Wang et al., 2026). Our focus is source-faithful, open-ended editing that combines few-step generation with bounded temporal state while maintaining editing consistency over sustained streams.

2.2 Long Video Generation

Diffusion Forcing assigns independent noise levels to temporal elements (Chen et al., 2024), while Self Forcing and Resampling Forcing expose training to model-generated histories to reduce train–inference mismatch (Huang et al., 2025; Guo et al., 2025). Distribution Matching Distillation accelerates diffusion sampling (Yin et al., 2024) and extends to causal video generation (Yin et al., 2025). LongLive and Helios further address long-duration stability and efficiency (Yang et al., 2025; Yuan et al., 2026). Beyond temporal coherence, streaming editing must preserve alignment with a changing source despite errors in generated history. We address this requirement through attention-driven source guidance during distillation and long-horizon optimization on model-generated trajectories.

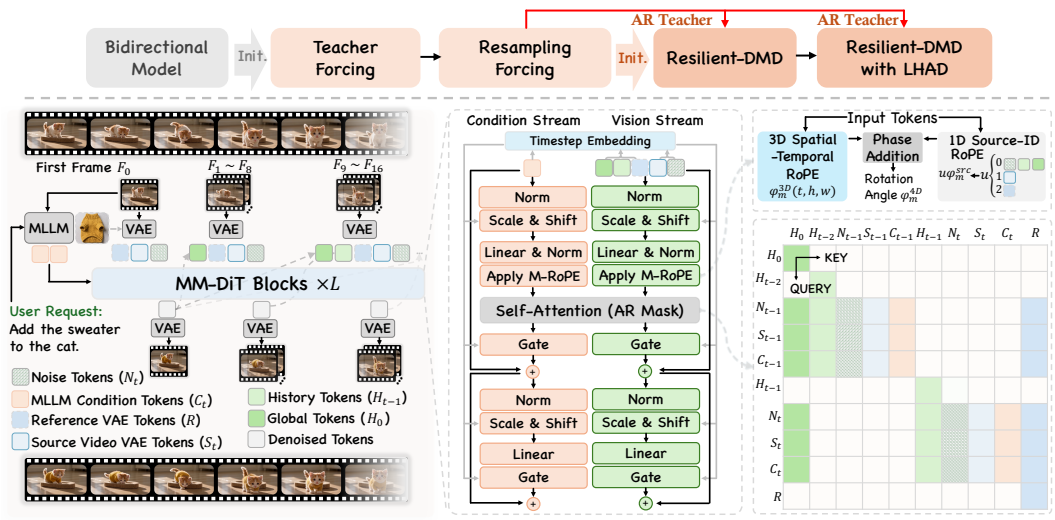

Figure 2: **Training pipeline and architecture of JoyAI-Video-Edit.** Causal adaptation, Resilient-DMD, and LHAD convert a bidirectional editor into a streaming model (top). Our model comprises an MLLM, a causal VAE, and an MM-DiT with 4D RoPE and a hybrid attention mask (bottom).

3 METHOD

In this section, we first describe chunk-wise autoregressive adaptation, then introduce Resilient-DMD and Long-Horizon Autoregressive Distillation (LHAD) to enable source-faithful, few-step editing over long horizons. We further present the real-time streaming engine, followed by the data engine for constructing training pairs and LongV2VBench for evaluating long-video editing.

3.1 CHUNK-WISE AUTOREGRESSIVE ADAPTATION

As shown in Figure 2, JoyAI-Video-Edit uses an MLLM to encode the instruction and first source frame into condition tokens c. An MM-DiT with L layers predicts target denoising velocities conditioned on these inputs. We first conduct foundation training for bidirectional text-to-video generation and video-to-video editing, as detailed in Appendix A. Then, we adapt the model to a chunk-wise autoregressive paradigm through supervised fine-tuning (SFT) with teacher forcing followed by resampling forcing (Guo et al., 2025). Specifically, we use aligned source target chunks with bidirectional attention within each chunk and causal access to a fixed recent history window plus the first chunk sink, as illustrated in Figure 2. We extend 3D spatio-temporal RoPE to four coordinates by composing it with a source-identity rotation on the same channels to distinguish input sources, as detailed in Appendix A.1. Teacher forcing conditions the denoising of the current chunk on the ground-truth target chunks corresponding to its visible history, together with the aligned source, instruction, and optional reference. To reduce the mismatch between training and inference, we then apply resampling forcing, replacing ground-truth history with chunks regenerated through single-step denoising and detached before reuse, thereby exposing the model to its own prediction errors.

3.2 RESILIENT-DMD

Copying temporally similar target history during SFT can reduce prediction loss but encourage over-reliance on generated history at inference, allowing errors to propagate and weakening responsiveness to source changes. To maintain source fidelity, we introduce Resilient-DMD, which adaptively adjusts teacher-side source guidance during distillation. Rather than strengthening source guidance uniformly, we selectively increase the teacher's source-conditioned correction when relative source attention falls below matched baselines, as detailed below.

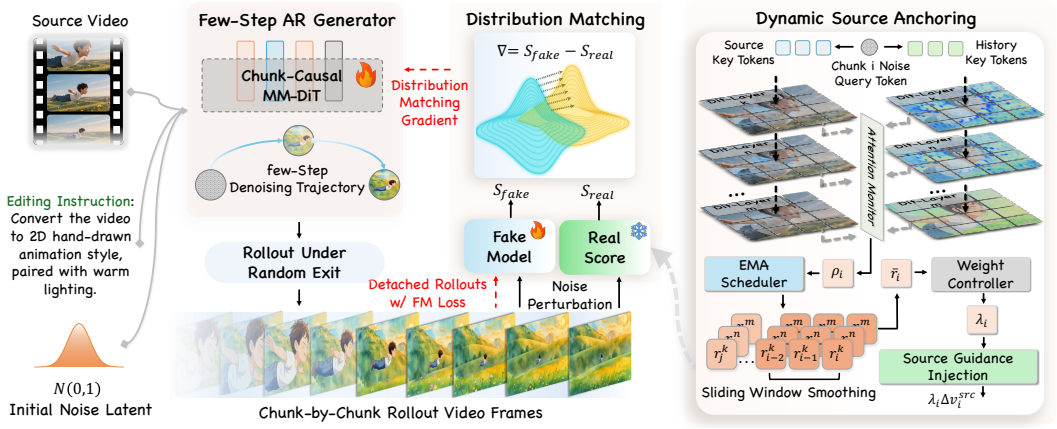

Figure 3: **Illustration of Resilient-DMD.** Relative source attention is normalized against matched EMA baselines and smoothed to selectively strengthen the real-score teacher's source guidance when grounding weakens. All monitoring and control operate only during distillation.

Grounding Monitor. We initialize the frozen real-score teacher, trainable generator and fake-score model from the SFT-trained editor. Given N source chunks and the corresponding editing instruction, the generator performs an autoregressive rollout using a predefined few-step denoising schedule for each chunk. Taking chunk i as an example, we terminate denoising at a randomly selected step and take the resulting prediction as the edited output. At this exit step, we measure relative source attention at selected generator layers $\ell \in \mathcal{L}$, where $\mathcal{L} \subseteq \{1, \dots, L\}$. Let $\mathcal{Q}_i$ denote noisy tokens and $\mathcal{S}_i, \mathcal{H}_i$ the visible source and generated-history key sets, with attention logits $a_{\ell,i}(q,k)$ from q to k. We compute the layer-wise relative source-attention ratios $\{\rho_i^\ell\}_{\ell \in \mathcal{L}}$ as follows:

$$\Delta_{\ell,i}(q) = \log \sum_{k \in \mathcal{S}_i} \exp\big(a_{\ell,i}(q,k)\big) - \log \sum_{k \in \mathcal{H}_i} \exp\big(a_{\ell,i}(q,k)\big),$$
$$\rho_i^\ell = \frac{1}{|\mathcal{Q}_i|} \sum_{q \in \mathcal{Q}_i} \frac{1}{1 + \exp\big(-\Delta_{\ell,i}(q)\big)}. \tag{1}$$

Here, ρ_i^ℓ averages the source-attention fraction within the combined source and history keys, measuring relative source reliance. Exact blockwise reductions reuse forward-pass queries and keys without additional model evaluations or materializing the attention matrix.

Weight Controller. We maintain exponential moving average (EMA) baselines $b_{\ell,s,d}$, indexed by layer ℓ, exit-step noise bin s, and rollout-depth bin d. Baselines accumulate valid observations across rollouts, remaining fixed within each rollout and updating only afterward to exclude the current trajectory from its own reference. The normalized statistic $r_i^\ell = b_{\ell,s_i,d_i}/\rho_i^\ell$ exceeds one when source attention falls below its matched baseline. We then apply sliding-window smoothing by averaging these statistics across monitored layers within a causal window of K chunks:

$$\bar{r}_i = \frac{1}{|\mathcal{V}_i|} \sum_{(\ell,j) \in \mathcal{V}_i} r_j^\ell, \qquad \mathcal{V}_i \subseteq \mathcal{L} \times \{\max(0, i-K+1), \dots, i\}, \tag{2}$$

where $\mathcal{V}_i$ contains valid layer–chunk pairs from the current rollout. The smoothed statistic drives a one-sided, bounded controller:

$$\lambda_i = \mathrm{clip}_{[\lambda_{\text{base}}, \lambda_{\max}]}\left[\lambda_{\text{base}} \left(\frac{\bar{r}_i}{1+\delta}\right)^{\kappa}\right], \tag{3}$$

where $\kappa > 0$ controls the response exponent and $\delta \geq 0$ defines a dead zone that retains λ_{base} for $\bar{r}_i \leq 1+\delta$. The dead zone mitigates positive anchoring bias from rectifying zero-mean fluctuations, while the cap limits source correction that could restrict editing flexibility.

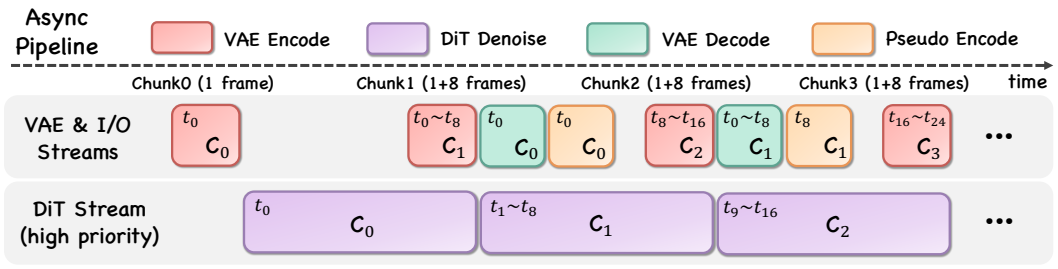

Figure 4: **Asynchronous streaming inference on a single GPU.** DiT denoising and cache storage overlap VAE processing of adjacent chunks under causal dependencies. Output packing is omitted.

Source Guidance Injection. For a generated chunk z_i, we form $\tilde{z}_{i,\sigma} = (1-\sigma)z_i + \sigma\epsilon$ with Gaussian noise ϵ. All teacher branches share this perturbed sample and the same generated history. Let v_i^{cond} denote the fully conditioned velocity, while $v_i^{-\text{txt}}$ and $v_i^{-\text{src}}$ omit the text condition and aligned source latent, respectively. We use $\bar{v}_i^{\text{txt}} = v_i^{\text{cond}} + \gamma(v_i^{\text{cond}} - v_i^{-\text{txt}})$ as the text-guided baseline, where γ is fixed and source conditioning is retained. With source-correction direction $\Delta v_i^{\text{src}} = v_i^{\text{cond}} - v_i^{-\text{src}}$, the guided teacher velocity is

$$v_i^{\text{SA}} = \bar{v}_i^{\text{txt}} + \lambda_i \, \Delta v_i^{\text{src}}. \tag{4}$$

The stop-gradient coefficient λ_i is computed and applied per sample and chunk in the same training step, scaling only the source correction. Source anchoring modifies only the real-score estimate, as detailed in Algorithm 1 of Appendix B. Inference uses two denoising steps with one conditional branch each, without score models, monitoring, or additional guidance branches.

3.3 Long-Horizon Autoregressive Distillation

Training on source-length rollouts does not supervise deeper states where errors in generated history can accumulate (Yang et al., 2025). We therefore introduce Long-Horizon Autoregressive Distillation (LHAD) to extend adaptive source supervision to these states. Specifically, we extend finite source videos through indexed forward–reverse traversal, avoiding end-to-start content jumps without materializing repeated sequences. We partition the resulting extended rollouts into segments, retaining a depth-calibrated λ_i for each chunk. Segment losses sum the per-chunk losses and are backpropagated separately. Generated chunks are detached before history reuse; each segment's graph is released before the next. Gradients accumulate across segments before a single optimizer update, as detailed in Algorithm 2 of Appendix B. As a result, bounding activation memory by segment length enables arbitrarily long training rollouts within a fixed memory budget. By exposing the editor to its own accumulated prediction errors during training, LHAD mitigates exposure bias and narrows the training–inference gap in long-horizon editing.

3.4 Real-Time Streaming Engine

To reduce end-to-end overhead, we develop a bounded-state asynchronous engine that integrates efficient transformer execution, single-frame VAE state handoff, and overlapped processing. Condition tokens and optional reference KV entries are computed once per session. The causal KV cache retains the first-chunk sink (Xiao et al., 2024) and a fixed recent history window, with rotary indices restricted to the trained range. Both denoising steps reuse this context, followed by a zero-noise *store pass* that updates the cache from the clean generated chunk. Fixed-shape buffers and a static KV pool enable joint CUDA-graph capture of denoising, the store pass, and cache commit, while FP8 linear layers and operator fusion provide further acceleration. For VAE processing, a single boundary frame carries context between chunks, enabling fixed-shape execution without internal feature caches. Encoding prepends the preceding boundary frame to new source frames, while decoding prepends a *pseudo latent* re-encoded from the last emitted frame and retains only outputs for the current chunk. As illustrated in Figure 4, bounded queues and CUDA events coordinate asynchronous execution, prioritizing DiT denoising for chunk i while overlapping it with decoding

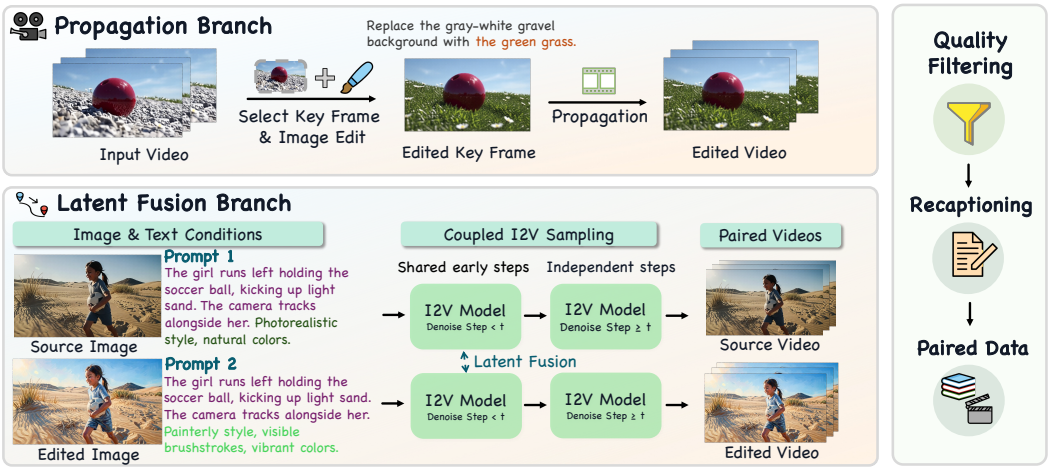

Figure 5: **Video editing data engine.** The propagation branch transfers keyframe edits to source videos, while the latent fusion branch couples image-to-video trajectories to generate aligned video pairs. Quality filtering and MLLM-based instruction refinement yield the final training data.

and re-encoding of chunk $i-1$ and source encoding of chunk $i+1$ when available. Prefetched source frames do not condition chunk i, preserving causality. Frame ingestion and output packing also overlap with model execution to reduce exposed I/O latency. For fixed resolution, chunk size, and condition length, temporal state and per-chunk work remain bounded independently of stream duration. Implementation details are provided in Appendix C.

3.5 Data Engine and LongV2VBench

To support training and sustained-stream evaluation, we construct paired editing data and introduce LongV2VBench. As shown in Figure 5, the data pipeline comprises two branches. In the propagation branch, an image-editing model (Song et al., 2026) edits a selected source-video keyframe, and an image-and-video-to-video model propagates the edit while preserving source motion and untargeted content. In the latent fusion branch, source–edited image pairs and their respective prompts condition two image-to-video trajectories. Early-step latent fusion encourages aligned motion and composition. Independent denoising then produces paired videos. Quality filtering and MLLM-based instruction refinement yield the final training pairs, with data distributions provided in Appendix A.2. For evaluation, LongV2VBench comprises 229 one-minute editing instances spanning background change, global style, local addition, local modification, and local removal. These long-video tasks mimic open-ended streaming scenarios to assess sustained editing quality, temporal consistency, and source fidelity, with full-pipeline throughput reported as a complementary metric. Task distributions and examples are provided in Appendix D.

4 Experiments

4.1 Implementation Details

A causal VAE encodes the source and optional reference inputs into latent representations with $8 \times 24 \times 24$ spatio-temporal compression. We use 1 latent frame per chunk and a 3-chunk attention window that includes the global sink chunk. Both SFT and Resilient-DMD use a learning rate of 5×10^{-5}, and Resilient-DMD employs LoRA with rank 64. We evaluate editing quality under BF16 inference at 720×1248 on a single NVIDIA B200 GPU, with each chunk generated in two denoising steps. Detailed settings are provided in Appendices B and C.

Table 1: Quantitative comparison of closed-source, open-source, and streaming video editing methods on OpenVE-Bench (He et al., 2025). Bold: best among streaming methods; underline: best among all methods.

Method	Params.	Resolution	Global Style	Local Change	Background Change	Local Remove	Local Add	Overall ↑
Closed-source Methods								
PixVerse V6 (PixVerse, 2026)	—	720×1280	3.02	4.10	2.23	2.82	3.09	3.05
Runway-Aleph (Runway, 2025)	—	720×1280	2.62	4.18	4.16	2.78	3.49	3.45
Kling-O1 (Kuaishou Technology, 2025)	—	1080×1920	3.38	4.44	3.23	3.32	3.74	3.62
Kling-3.0 Omni (Kuaishou Technology, 2026)	—	1080×1920	4.29	4.29	3.46	3.37	3.54	3.79
Open-source Methods								
VACE (Jiang et al., 2025)	14B	720×1280	1.49	1.55	2.07	1.46	1.26	1.57
DITTO (Bai et al., 2026)	14B	480×832	4.01	1.68	2.03	1.53	1.41	2.13
Kiwi-Edit (Lin et al., 2026)	5B	704×1280	3.64	2.64	3.83	2.63	2.36	3.02
Bernini-R (Bernini Team, 2026)	27B	480×848	4.16	4.47	3.25	3.88	2.89	3.72
Streaming Video Editing								
StreamDiffusionV2 (Feng et al., 2025)	1.3B	480×832	1.48	1.35	1.01	1.27	1.05	1.23
SANA-Streaming (Zhao et al., 2026)	2B	704×1280	3.48	2.29	3.20	2.27	1.88	2.62
LiveEdit (Wang et al., 2026)	1.3B	480×832	2.18	2.73	2.05	1.55	1.51	2.00
XMax-X2.0 (Xmax AI, 2026)	—	832×1440	2.47	2.09	1.63	1.73	1.41	1.87
JoyAI-Video-Edit (Ours)	16B	720×1248	**3.94**	**4.50**	**3.25**	**3.50**	**3.38**	**3.72**

Table 2: Quantitative comparison with streaming video editing methods on LongV2VBench. Overall is the mean of the five subtask scores. Full Throughput denotes the end-to-end processing speed measured in frames per second, where higher values indicate faster inference. The best result among streaming methods is shown in bold.

Method	Resolution	Throughput (FPS) ↑	Background Change	Global Style	Local Add	Local Change	Local Remove	Overall ↑
StreamDiffusionV2 (Feng et al., 2025)	480×832	18.07	1.08	1.71	1.18	1.11	1.03	1.22
SANA-Streaming (Zhao et al., 2026)	704×1280	14.51	1.19	2.02	1.50	1.72	1.85	1.66
LiveEdit (Wang et al., 2026)	480×832	15.45	1.10	1.34	1.33	1.34	1.04	1.23
XMax-X2.0 (Xmax AI, 2026)	832×1440	20.90	1.36	2.07	1.64	2.08	1.40	1.71
JoyAI-Video-Edit (Ours)	720×1248	**34.87**	**3.15**	**3.69**	**3.22**	**4.17**	**3.85**	**3.61**

4.2 Main Results

Benchmark Evaluation. We compare with streaming counterparts on OpenVE-Bench and our proposed LongV2VBench, additionally including open- and closed-source offline models on OpenVE-Bench. As shown in Table 1, our method achieves an overall score of 3.72, matching the best open-source offline result while outperforming streaming counterparts across all five categories. It also achieves the highest local-change score of 4.50 among all methods. Table 2 further shows that our method leads all five categories on LongV2VBench, achieving an overall score of 3.61 and an end-to-end throughput of 34.87 FPS. Human evaluation in Appendix E.3 similarly favors our method over streaming counterparts, with more balanced preferences against offline models.

Deployment Efficiency. As shown in Table 7, our pipeline processes 81 frames in 2.32 s, leading the evaluated streaming baselines in clip-level latency and steady-state throughput. On a single B200 at $Q = 1$, median chunk-processing latency is 261.06 ms, excluding frame accumulation, prompt encoding, and network transfer. In Table 3, two-GPU execution improves throughput by 1.16–1.28× and reduces median latency at $Q = 4$, with little latency change at $Q = 1$. On consumer-grade RTX 5090 at 480×840, throughput rises from 27.76 to 32.99 FPS. Detailed timing boundaries and profiling are provided in Appendix E.2.

4.3 Qualitative Results

All examples in Figure 6 span 60 seconds. Other streaming methods exhibit pronounced degradation in both instruction adherence and visual quality over long editing horizons. Requested edits become inconsistent or disappear, while blur and structural artifacts intensify in later frames. In contrast, our method sustains the specified background and appearance edits while preserving scene structure and visual clarity across the displayed sequences. Figure 7 further illustrates clothing replacement guided by reference images. The reference-conditioned edits remain temporally consistent even dur-



Figure 6: **Qualitative comparison with streaming video editing methods on one-minute videos.**

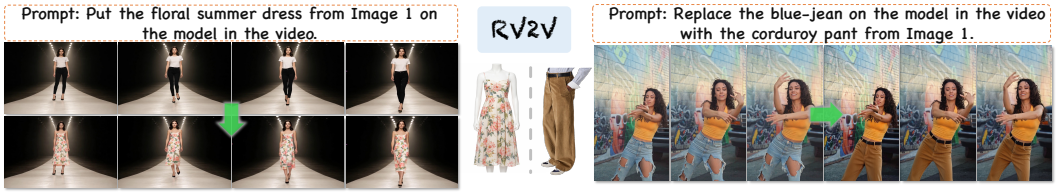

Figure 7: **Qualitative results of reference-conditioned video editing.**

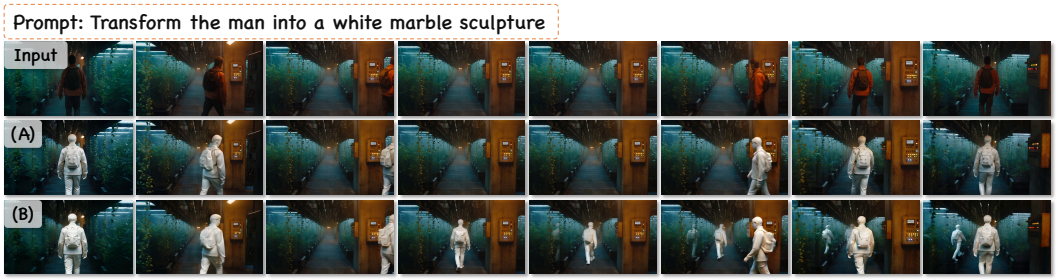

Figure 8: **Qualitative comparison of models trained (A) with and (B) without Resilient-DMD.**

ing large-amplitude subject motion (see Appendix E.4 for additional examples). Moreover, Figure 8 shows that the model trained with Resilient-DMD faithfully follows source changes. Without it, the model reproduces the subject from generated history after the subject disappears from the source, creating a phantom figure and leaving two figures in later frames. Finally, a real user's online editing experience, shown in Figure 14 of Appendix E.4, further illustrates the effectiveness of our model and asynchronous deployment engine in supporting responsive editing of live camera input.

4.4 Ablation Study

Distillation components. Table 4 isolates the contributions of Resilient-DMD and LHAD on LongV2VBench. Global averages global-style and background-change scores, and Local averages local-add, local-change, and local-removal scores. Overall averages all five categories. From a baseline score of 2.81, Resilient-DMD alone improves the overall score to 3.38, while LHAD alone reaches 3.19. Combining both yields the highest Local (3.75) and Overall (3.61) scores while matching Resilient-DMD alone on Global (3.42).

Hyperparameter sensitivity. As shown in Table 5, we vary the smoothing window K, dead-zone margin δ, and base anchoring strength λ_{base} one at a time on LongV2VBench. Overall scores range

Table 3: **Engine scaling across GPUs.** Q is the maximum number of chunks concurrently in flight within the engine.

GPU / Resolution ($H \times W$)	GPU Count	Throughput ($Q=4$) Rate (FPS)	Speedup ($\times$)	Latency ($Q=1$) P50 (ms)	Latency ($Q=1$) P95 (ms)	Latency ($Q=4$) P50 (ms)	Latency ($Q=4$) P95 (ms)
B200 720×1248	1	$34.87_{\pm 0.103}$	1.00	261.06	269.45	917.62	926.07
	2	$40.49_{\pm 0.046}$	1.16	260.22	269.51	789.50	794.73
PRO6000 720×1248	1	$15.20_{\pm 0.047}$	1.00	573.92	578.75	2102.19	2113.87
	2	$19.37_{\pm 0.024}$	1.28	578.10	581.59	1651.02	1661.02
RTX 5090 480×840	1	$27.76_{\pm 0.397}$	1.00	337.08	345.81	1159.81	1166.34
	2	$32.99_{\pm 0.315}$	1.19	335.40	343.40	966.66	1180.73

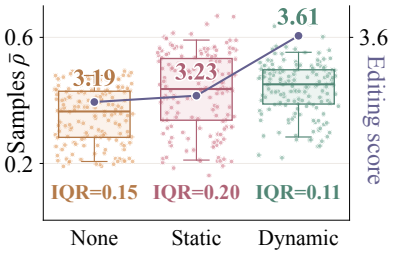

Figure 9: **Effect of source anchoring.** Source attention $\bar{\rho}$ and editing scores across anchoring strategies.

Table 4: **Ablation of distillation components on LongV2VBench.** Global and Local average two and three editing categories, respectively. Bold denotes column-wise maxima.

Resilient-DMD	LHAD	Global	Local	Overall
		3.03	2.66	2.81
✓		3.42	3.35	3.38
	✓	3.34	3.10	3.19
✓	✓	**3.42**	**3.75**	**3.61**

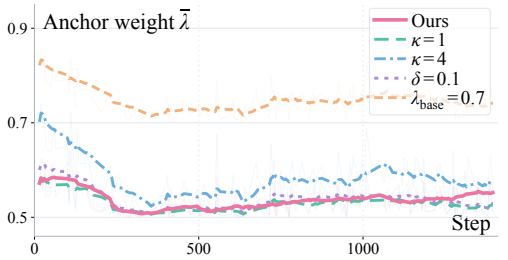

Figure 10: **Anchor-weight dynamics.** Mean anchoring weight $\bar{\lambda}$ over training under different controller settings in Resilient-DMD.

Table 5: **Analysis of Resilient-DMD hyperparameters on LongV2VBench.**

K	Global	Local	Overall	δ	Global	Local	Overall	λ_{base}	Global	Local	Overall
1	3.41	3.32	3.36	0.00	3.26	3.07	3.14	0.3	3.34	**3.37**	3.36
2	**3.42**	**3.35**	**3.38**	0.05	**3.42**	**3.35**	**3.38**	0.5	**3.42**	3.35	**3.38**
4	3.31	3.33	3.32	0.10	3.28	3.28	3.28	0.7	3.10	3.17	3.14

from 3.32 to 3.38 across the tested smoothing windows, indicating limited sensitivity to K. In contrast, removing the dead zone or increasing λ_{base} to 0.7 reduces the overall score from 3.38 to 3.14. The latter setting also maintains consistently higher mean anchoring weights during training, as shown in Figure 10. Together, these results favor moderate, selective source correction over uniformly stronger anchoring.

Effects of different source anchoring strategies. Figure 9 compares per-sample source-attention ratios $\bar{\rho}$ and editing scores under no, static, and dynamic anchoring, with static anchoring using the best fixed weight tested. Static anchoring raises source attention but widens its interquartile range (IQR) from 0.15 to 0.20, with only a modest score gain from 3.19 to 3.23. Dynamic anchoring reaches 3.61, exceeding the tested static baseline by 0.38, while reducing the IQR to 0.11. Thus, its advantage is reflected in editing quality, not merely in a more concentrated attention distribution. This is consistent with Resilient-DMD's selective reinforcement of weak source grounding, which adapts correction strength across chunks rather than applying a fixed source-guidance weight.

5 Conclusion

We present JoyAI-Video-Edit, a 16B-parameter autoregressive diffusion model for source-faithful, real-time streaming video editing. We introduce Resilient-DMD to counter overreliance on generated history through selective teacher-side source guidance, while LHAD extends this supervision to longer rollouts with bounded activation memory. Our bounded-state asynchronous engine reduces overhead through single-frame VAE state handoff and overlapped execution. Our data pipeline supports training, and LongV2VBench evaluates sustained editing on one-minute videos. Automatic and human evaluations show gains over the evaluated streaming baselines and competitive quality with offline editors. Our system achieves end-to-end 720p editing at 34.87 FPS on a single NVIDIA B200 GPU and supports deployment on RTX PRO 6000 and consumer-grade RTX 5090 GPUs.

AI USE STATEMENT

In this work, we used generative AI to enhance the grammar, clarity, and readability of the manuscript. The research ideas, theoretical developments, mathematical claims and proofs, method implementations, and experimental design and execution are the work of the authors. The authors reviewed and, where necessary, revised all AI-assisted edits and take full responsibility for the final manuscript, including its claims, results, and conclusions.

REFERENCES

Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, et al. Pytorch 2: Faster machine learning through dynamic python bytecode transformation and graph compilation. In *Proceedings of the ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, 2024.

Qingyan Bai, Qiuyu Wang, Hao Ouyang, Yue Yu, Hanlin Wang, Wen Wang, Ka Leong Cheng, Shuailei Ma, Yanhong Zeng, Zichen Liu, Yinghao Xu, Yujun Shen, and Qifeng Chen. Scaling instruction-based video editing with a high-quality synthetic dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 37971–37981, 2026.

Bernini Team. Bernini: Latent semantic planning for video diffusion. *arXiv preprint arXiv:2605.22344*, 2026.

ByteDance Seed. Seedance 2.0: Advancing video generation for world complexity. *arXiv preprint arXiv:2604.14148*, 2026.

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *Advances in Neural Information Processing Systems*, 37:24081–24125, 2024.

Junyi Chen, Tong He, Zhoujie Fu, Pengfei Wan, Kun Gai, and Weicai Ye. Vino: A unified visual generator with interleaved omnimodal context. *arXiv preprint arXiv:2601.02358*, 2026.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022.

DecartAI Team. Lucy edit: Open-weight text-guided video editing. Technical report, DecartAI, 2025. URL https://d2drjpuinn46lb.cloudfront.net/Lucy_Edit__High_Fidelity_Text_Guided_Video_Editing.pdf.

Tianrui Feng, Zhi Li, Shuo Yang, Haocheng Xi, Muyang Li, Xiuyu Li, Lvmin Zhang, Keting Yang, Kelly Peng, Song Han, et al. Streamdiffusionv2: A streaming system for dynamic and interactive video generation. *arXiv preprint arXiv:2511.07399*, 2025.

Yuwei Guo, Ceyuan Yang, Hao He, Yang Zhao, Meng Wei, Zhenheng Yang, Weilin Huang, and Dahua Lin. End-to-end training for autoregressive video diffusion via self-resampling. *arXiv preprint arXiv:2512.15702*, 2025.

Haoyang He, Jie Wang, Jiangning Zhang, Zhucun Xue, Xingyuan Bu, Qiangpeng Yang, Shilei Wen, and Lei Xie. Openve-3m: A large-scale high-quality dataset for instruction-guided video editing. *arXiv preprint arXiv:2512.07826*, 2025.

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.

Zeyinzi Jiang, Zhen Han, Chaojie Mao, Jingfeng Zhang, Yulin Pan, and Yu Liu. Vace: All-in-one video creation and editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 17191–17202, 2025.

Kuaishou Technology. Kling o1 launches as the world's first unified multimodal video model. https://ir.kuaishou.com/news-releases/news-release-details/kling-o1-launches-worlds-first-unified-multimodal-video-model-0/, 2025. Accessed: 2026-08-03.

Kuaishou Technology. Kling ai 3.0 and video 3.0 omni. https://ir.kuaishou.com/news-releases/news-release-details/kling-ai-launches-30-model-ushering-era-where-everyone-can-be, 2026. Accessed: 2026-07-28.

Runjia Li, Moayed Haji-Ali, Ashkan Mirzaei, Chaoyang Wang, Arpit Sahni, Ivan Skorokhodov, Aliaksandr Siarohin, Tomas Jakab, Junlin Han, Sergey Tulyakov, et al. Egoedit: Dataset, real-time streaming model, and benchmark for egocentric video editing. *arXiv preprint arXiv:2512.06065*, 2025.

Xinyao Liao, Xianfang Zeng, Ziye Song, Zhoujie Fu, Gang Yu, and Guosheng Lin. In-context learning with unpaired clips for instruction-based video editing. *arXiv preprint arXiv:2510.14648*, 2025.

Yiqi Lin, Guoqiang Liang, Ziyun Zeng, Zechen Bai, Yanzhe Chen, and Mike Zheng Shou. Kiwi-edit: Versatile video editing via instruction and reference guidance. *arXiv preprint arXiv:2603.02175*, 2026.

Paulius Micikevicius, Dusan Stosic, Neil Burgess, Marius Cornea, Pradeep Dubey, Richard Grisenthwaite, Sangwon Ha, Alexander Heinecke, Patrick Judd, John Kamalu, et al. Fp8 formats for deep learning. *arXiv preprint arXiv:2209.05433*, 2022.

Kaihang Pan, Qi Tian, Jianwei Zhang, Weijie Kong, Jiangfeng Xiong, Yanxin Long, Shixue Zhang, Haiyi Qiu, Tan Wang, Zheqi Lv, Yue Wu, Liefeng Bo, Siliang Tang, and Zhao Zhong. Omniweaving: Towards unified video generation with free-form composition and reasoning. *arXiv preprint arXiv:2603.24458*, 2026.

PixVerse. Pixverse launches v6, advancing ai video generation across creative and agentic workflows. https://pixverse.ai/en/blog/pixverse-launches-v6-advancing-ai-video-generation, 2026. Accessed: 2026-08-03.

Chenyang Qi, Xiaodong Cun, Yong Zhang, Chenyang Lei, Xintao Wang, Ying Shan, and Qifeng Chen. Fatezero: Fusing attentions for zero-shot text-based video editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15932–15942, 2023.

Runway. Runway aleph: A new way to edit, transform and generate video. https://runwayml.com/research/introducing-runway-aleph, 2025.

Lin Song, Wenbo Li, Guoqing Ma, Wei Tang, Bo Wang, Yuan Zhang, Yijun Yang, Yicheng Xiao, Jianhui Liu, Yanbing Zhang, Guohui Zhang, Wenhu Zhang, Hang Xu, Nan Jiang, Xin Han, Haoze Sun, Maoquan Zhang, Haoyang Huang, and Nan Duan. Awaking spatial intelligence in unified multimodal understanding and generation. *arXiv preprint arXiv:2605.04128*, 2026.

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

Cong Wei, Quande Liu, Zixuan Ye, Qiulin Wang, Xintao Wang, Pengfei Wan, Kun Gai, and Wenhu Chen. Univideo: Unified understanding, generation, and editing for videos. *arXiv preprint arXiv:2510.08377*, 2025.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations*, pp. 21875–21895, 2024.

Xmax AI. Xmax x2.0: Real-time interactive video model. https://xmax.ai/, 2026. Accessed: 2026-07-28.

Shuai Yang, Wei Huang, Ruihang Chu, Yicheng Xiao, Yuyang Zhao, Xianbang Wang, Muyang Li, Enze Xie, Yingcong Chen, Yao Lu, et al. Longlive: Real-time interactive long video generation. *arXiv preprint arXiv:2509.22622*, 2025.

Zixuan Ye, Xuanhua He, Quande Liu, Qiulin Wang, Xintao Wang, Pengfei Wan, Di Zhang, Kun Gai, Qifeng Chen, and Wenhan Luo. Unic: Unified in-context video editing. *arXiv preprint arXiv:2506.04216*, 2025.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and Bill Freeman. Improved distribution matching distillation for fast image synthesis. *Advances in Neural Information Processing Systems*, 37:47455–47487, 2024.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22963–22974, 2025.

Shenghai Yuan, Yuanyang Yin, Zongjian Li, Xinwei Huang, Xiao Yang, and Li Yuan. Helios: Real real-time long video generation model. *arXiv preprint arXiv:2603.04379*, 2026.

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.

APPENDIX

A FOUNDATION TRAINING AND DATA

A.1 TRAINING CURRICULUM

Image and video generation. We pretrain on text-to-image (T2I) generation at 256×256 and 512×512 resolution to establish text–visual alignment and spatial generation priors. This checkpoint initializes text-to-video (T2V) training, which introduces temporal modeling while retaining T2I supervision. Training begins with low-resolution videos at 12 and 24 fps, followed by increased spatial resolution and frame rate. Supervised fine-tuning (SFT) on higher-quality 480p data and subsequent continual training (CT) improve visual quality, motion coherence, and prompt alignment.

Image and bidirectional video editing. We retain T2V and T2I supervision while introducing I2I, then jointly train V2V and IV2V editing with these tasks (Jiang et al., 2025). For source, reference, and target latents z^s, z^r, and z_0, we sample $\sigma \in [0, 1]$ and Gaussian noise ϵ and optimize

$$\begin{aligned} z_\sigma &= (1-\sigma)z_0 + \sigma\epsilon, \\ \mathcal{L}_{\text{V2V}} &= \mathrm{E}\Big[\|v_{\theta_{\text{bi}}}(z_\sigma, \sigma; c, z^s, z^r) - (\epsilon - z_0)\|_2^2\Big]. \end{aligned} \tag{5}$$

V2V samples omit reference latents. Target attention is bidirectional at this stage, providing full-clip context before causal autoregressive adaptation. Table 6 summarizes the curriculum.

Table 6: **Foundation training curriculum.** Volume denotes the number of training samples; for T2V stages, it reports the video portion of the mixture.

Phase	Task	Resolution	fps	LR	Volume
Stage 1: Text-to-Image Pretraining					
Pretrain	T2I	256×256	–	1×10^{-4}	4.3B
Pretrain	T2I	512×512	–	1×10^{-4}	793M
Stage 2: Text-to-Video Training					
Pretrain	T2V/T2I	256p	12/24	1×10^{-4}	370M
Pretrain	T2V/T2I	360p	24	5×10^{-5}	108M
SFT	T2V/T2I	480p	24	5×10^{-5}	30M
CT	T2V/T2I	480p	24	5×10^{-5}	6.5M
Stage 3: Image-Editing Training					
SFT	I2I/T2V/T2I	720p	–	1×10^{-4}	3.2M
Stage 4: Bidirectional Video-Editing Training					
SFT	V2V/IV2V/I2I/T2V/T2I	720p	24	5×10^{-5}	5.2M
CT	V2V/IV2V/I2I/T2V/T2I	720p	24	5×10^{-5}	1.1M

4D rotary positional encoding. We compose 3D spatiotemporal RoPE with a parameter-free source-identity rotation on the same head channels, rather than partitioning the channels into four groups. Each attention head has $D_{\text{head}} = 128$ channels, allocated as $(D_t, D_h, D_w) = (16, 56, 56)$ to temporal, height, and width coordinates. Concatenating the three axis-specific rotary phases yields $\varphi_m^{\text{3D}}(t, h, w)$ for channel-pair index $m \in \{0, \ldots, D_{\text{head}}/2 - 1\}$, where t denotes frame position rather than diffusion time, and h, w index the spatial patch grid. The source-identity coordinate u is 0 for noisy and clean target tokens, 1 for source-video tokens, and 2 for reference-image tokens; these IDs are fixed by input category, not assigned separately to individual references. With an even source-rotation dimension $D_{\text{src}} \le D_{\text{head}}$ and frequency base β_{src}, the combined phase is

$$\begin{aligned} \varphi_m^{\text{4D}}(t, h, w, u) &= \varphi_m^{\text{3D}}(t, h, w) + u\,\omega_m^{\text{src}}, \\ \omega_m^{\text{src}} &= \begin{cases} \beta_{\text{src}}^{-2m/D_{\text{src}}}, & 0 \le m < D_{\text{src}}/2, \\ 0, & D_{\text{src}}/2 \le m < D_{\text{head}}/2. \end{cases} \end{aligned} \tag{6}$$

We use $D_{\text{src}} = D_{\text{head}} = 128$ and $\beta_{\text{src}} = 256$, so source rotation covers all head channels. Phase addition is implemented by multiplying the corresponding complex rotation factors, and the combined rotation is applied to visual queries and keys after query–key normalization, leaving values unchanged. Noisy and clean targets share the same spatiotemporal coordinates; reference images

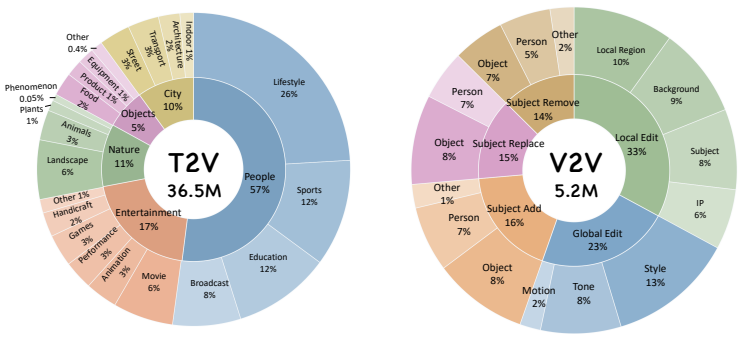

Figure 11: **Training data distributions.** T2V continual training (CT) and supervised fine-tuning (SFT) cover diverse domains; V2V SFT includes global, local, and subject-level editing tasks.

use $t = 0$, while source videos reuse target temporal indices when their frame counts match and otherwise use indices starting at zero. The additional source phases cancel for same-source query–key pairs, whereas cross-source pairs retain a phase offset proportional to their source-ID difference. Noisy and clean target tokens at the same position therefore have identical rotary encodings.

A.2 Foundation Data Curation

The generation corpus covers people, lifestyle, entertainment, nature, objects, and urban scenes. Images undergo sharpness and quality assessment, black-border detection, and perceptual-hash- and embedding-based deduplication. Videos are additionally screened for aesthetics, motion magnitude, camera stability, and temporal validity; blurred, rotated, presentation-style, and corrupted clips are removed. We cluster the filtered data and downsample dominant categories while retaining long-tail concepts. Motion-based filtering counterbalances the tendency of aesthetic selection to favor static videos. For high-quality fine-tuning subsets, manual inspection further removes prominent text, broken structures, implausible motion, and violations of basic physical consistency. Figure 11 summarizes the generation and editing data distributions.

B Distillation Implementation Details

B.1 Resilient-DMD Objective

The causal generator G_θ, trainable fake-score model F_ψ, and frozen real-score teacher R are initialized from the adapted editor under a shared-backbone LoRA configuration. For the perturbed chunk $\tilde{z}_{i,\sigma} = (1-\sigma)z_i + \sigma\epsilon$, we obtain the source-guided real-score estimate from the teacher's guided velocity v_i^{SA}. We optimize the generator using the distribution-matching gradient derived from the discrepancy between this real-score estimate and the fake-score estimate from F_ψ. Source anchoring modifies only the real-score estimate, while retaining the standard DMD update formulation. The fake-score model is trained by flow matching on detached generator samples. We average EMA-normalized ratios over the selected generator layers $\mathcal{L}$ and a causal window of $K = 2$ chunks. The controller uses $(\lambda_{\mathrm{base}}, \lambda_{\max}, \delta, \kappa) = (0.5, 1.0, 0.05, 2)$, with a quadratic response above the dead zone.

Shared notation. Dataset $\mathcal{D}$ contains pairs (X, c), with X a sequence of source chunks x_i and c the condition tokens in Section 3.1. Both algorithms use a bounded history cache $\mathcal{C}$, EMA baselines $\mathbf{b} = \{b_{\ell,s,d}\}$, and detached monitoring state $\mathcal{U}$ storing recent normalized ratios and per-bin attention accumulators. g_θ and g_ψ accumulate parameter gradients for G_θ and F_ψ, respectively; sg denotes stop-gradient. CacheUpdate retains the first-chunk sink and recent history window; OptStep takes one optimizer step; EMAUpdate updates $\mathbf{b}$ from the per-bin attention statistics in $\mathcal{U}$.

Chunk operators. Given G_θ, a few-step schedule $\mathcal{T}$, and layer set $\mathcal{L}$, ExitRollout returns the random-exit prediction z_i, source-attention ratios $\{\rho_i^\ell\}_{\ell\in\mathcal{L}}$, and exit-step noise bin s_i. DepthBin

maps global chunk index i to rollout-depth bin d_i. MonitorUpdate returns the updated $\mathcal{U}$ and valid layer–chunk pairs $\mathcal{V}_i$ in the causal K-chunk window. TeacherGuidance evaluates the frozen teacher R under three conditioning settings sharing the perturbed chunk and history, returning $\bar{v}_i^{\text{txt}}$ and Δv_i^{src} as defined in Section 3.2. Let $\hat{z}_i^{\text{real}} = \tilde{z}_{i,\sigma} - \sigma v_i^{\text{SA}}$ and $\hat{z}_i^{\text{fake}}$ denote the guided teacher and fake-score clean estimates, respectively. DMDLoss defines the generator loss ℓ_i^G, averaged over valid latent frames, using the detached distillation signal

$$g_i = \text{sg}\left[\frac{\hat{z}_i^{\text{fake}} - \hat{z}_i^{\text{real}}}{\text{mean}(|z_i - \hat{z}_i^{\text{real}}|)}\right]. \tag{7}$$

Here, mean averages over the latent elements of chunk i; source anchoring affects both numerator and denominator through $\hat{z}_i^{\text{real}}$. The loss ℓ_i^G backpropagates g_i only through z_i to θ. The flow-matching loss ℓ_i^F, computed by FMLoss using F_ψ on $\text{sg}(z_i)$, updates only ψ.

Training procedure. Algorithm 1 summarizes Resilient-DMD with the notation above; EMA baselines remain fixed within each rollout and are updated only afterward.

Algorithm 1 Resilient-DMD

Require: SFT-trained editor; dataset $\mathcal{D}$; few-step schedule $\mathcal{T}$
Require: $\mathcal{L}$, K, γ, $(\lambda_{\text{base}}, \lambda_{\max}, \delta, \kappa)$
Ensure: Two-step causal generator G_θ

1: $G_\theta, F_\psi, R \leftarrow$ SFT-trained editor; freeze R
2: Initialize $\mathbf{b}$
3: **while** not converged **do**
4: $(X, c) \sim \mathcal{D}$; $N \leftarrow |X|$
5: $\mathcal{C} \leftarrow \emptyset, \mathcal{U} \leftarrow \emptyset$, $g_\theta \leftarrow 0$, $g_\psi \leftarrow 0$
6: **for** $i = 0, \ldots, N-1$ **do**
7: $(z_i, \{\rho_i^\ell\}_{\ell\in\mathcal{L}}, s_i) \leftarrow \text{ExitRollout}(G_\theta, x_i, c, \mathcal{C}; \mathcal{T}, \mathcal{L})$
8: $d_i \leftarrow \text{DepthBin}(i)$; $r_i^\ell \leftarrow b_{\ell,s_i,d_i}/\rho_i^\ell$
9: $(\mathcal{U}, \mathcal{V}_i) \leftarrow \text{MonitorUpdate}(\mathcal{U}, \{\rho_i^\ell, r_i^\ell\}_{\ell\in\mathcal{L}}, s_i, d_i, i; K)$
10: $\bar{r}_i \leftarrow \frac{1}{|\mathcal{V}_i|}\sum_{(\ell,j)\in\mathcal{V}_i} r_j^\ell$
11: $\lambda_i \leftarrow \text{sg}\Big(\text{clip}_{[\lambda_{\text{base}},\lambda_{\max}]}\Big[\lambda_{\text{base}}\left(\frac{\bar{r}_i}{1+\delta}\right)^\kappa\Big]\Big)$
12: Sample σ and $\epsilon \sim \mathcal{N}(0, I)$; $\tilde{z}_{i,\sigma} \leftarrow (1-\sigma)z_i + \sigma\epsilon$
13: $(\bar{v}_i^{\text{txt}}, \Delta v_i^{\text{src}}) \leftarrow \text{TeacherGuidance}(R, \tilde{z}_{i,\sigma}, \sigma, x_i, c, \mathcal{C}; \gamma)$
14: $v_i^{\text{SA}} \leftarrow \bar{v}_i^{\text{txt}} + \lambda_i \Delta v_i^{\text{src}}$
15: $\hat{z}_i^{\text{real}} \leftarrow \tilde{z}_{i,\sigma} - \sigma v_i^{\text{SA}}$, $\hat{z}_i^{\text{fake}} \leftarrow \tilde{z}_{i,\sigma} - \sigma F_\psi(\tilde{z}_{i,\sigma}, \sigma, x_i, c, \mathcal{C})$
16: $\ell_i^G \leftarrow \text{DMDLoss}(z_i; \hat{z}_i^{\text{real}}, \hat{z}_i^{\text{fake}})$, $\ell_i^F \leftarrow \text{FMLoss}(F_\psi, \text{sg}(z_i); x_i, c, \mathcal{C})$
17: $g_\theta \leftarrow g_\theta + \nabla_\theta \ell_i^G$, $g_\psi \leftarrow g_\psi + \nabla_\psi \ell_i^F$
18: $\mathcal{C} \leftarrow \text{CacheUpdate}(\mathcal{C}, \text{sg}(z_i))$
19: **end for**
20: $\theta \leftarrow \text{OptStep}(\theta, g_\theta)$, $\psi \leftarrow \text{OptStep}(\psi, g_\psi)$
21: $\mathbf{b} \leftarrow \text{EMAUpdate}(\mathbf{b}, \mathcal{U})$
22: **end while**

We denote lines 7–16 of Algorithm 1 by RDMD, which takes G_θ, F_ψ, and R as explicit inputs and uses the same denoising schedule and controller settings. This operator returns $(z_i, \ell_i^G, \ell_i^F, \mathcal{U})$ without updating caches, model parameters, or EMA baselines, and is reused in Algorithm 2.

B.2 Long-Horizon Autoregressive Distillation

Following Section 3.3, LHAD extends finite source videos through indexed forward–reverse traversal, avoiding end-to-start content jumps without materializing repeated sequences. We supervise all chunks of the extended rollout with depth-calibrated source anchoring and partition the rollout into consecutive segments $\{\mathcal{G}_q\}$. The generator objective is

$$\mathcal{L}_{\text{long}} = \sum_q \mathcal{L}_{\text{DMD}}(\mathcal{G}_q), \tag{8}$$

where $\mathcal{L}_{\mathrm{DMD}}(\mathcal{G}_q) = \sum_{i \in \mathcal{G}_q} \ell_i^G$ sums the per-chunk losses defined above, each normalized over valid latent frames and using its own λ_i. History is detached after each chunk, as in Algorithm 1. Each segment loss is backpropagated separately, and its graph is released before the next segment. Gradients accumulate across segments before a single generator update. This bounds activation memory by segment length, enabling longer training rollouts that expose the editor to accumulated history errors and help mitigate exposure bias.

Long-horizon notation. Let $\Pi_i(X)$ retrieve the source chunk at global rollout index i under forward–reverse traversal. For a rollout of N chunks and maximum segment length B, define

$$Q = \lceil N/B \rceil, \qquad \mathcal{G}_q = \{(q-1)B, \ldots, \min(qB, N) - 1\}, \tag{9}$$

where $q = 1, \ldots, Q$. $\mathrm{Release}(\mathcal{G}_q)$ frees segment-local tensors and computation graphs while preserving the shared cache and monitoring state.

Training procedure. Algorithm 2 focuses on segmented generator optimization and omits fake-score update details; $\cdot$ denotes an unused return value. Global indices and monitoring state persist across segments; $\mathrm{EMAUpdate}$ is applied only after the full rollout, as in Algorithm 1.

Algorithm 2 Long-Horizon Autoregressive Distillation (LHAD)

Require: $\mathcal{D}$, G_θ, F_ψ, frozen R, EMA baselines $\mathbf{b}$; rollout length N, segment limit B
Require: Few-step schedule $\mathcal{T}$ and Resilient-DMD settings from Algorithm 1
Ensure: Long-horizon editor G_θ

while not converged **do**
  $(X, c) \sim \mathcal{D}$; form $\{\mathcal{G}_q\}_{q=1}^{Q}$ using Equation (9)
  $\mathcal{C} \leftarrow \emptyset, \mathcal{U} \leftarrow \emptyset, g_\theta \leftarrow 0$
  for $q = 1, \ldots, Q$ **do**
    for $i \in \mathcal{G}_q$ in increasing order **do**
      $(z_i, \ell_i^G, \cdot, \mathcal{U}) \leftarrow \mathrm{RDMD}(G_\theta, F_\psi, R, \Pi_i(X), c, \mathcal{C}, \mathcal{U}, \mathbf{b}, i)$
      $\mathcal{C} \leftarrow \mathrm{CacheUpdate}(\mathcal{C}, \mathrm{sg}(z_i))$
    end for
    $\mathcal{L}_{\mathrm{DMD}}(\mathcal{G}_q) \leftarrow \sum_{i \in \mathcal{G}_q} \ell_i^G$
    $g_\theta \leftarrow g_\theta + \nabla_\theta \mathcal{L}_{\mathrm{DMD}}(\mathcal{G}_q)$
    $\mathrm{Release}(\mathcal{G}_q)$
  end for
  $\theta \leftarrow \mathrm{OptStep}(\theta, g_\theta)$
  $\mathbf{b} \leftarrow \mathrm{EMAUpdate}(\mathbf{b}, \mathcal{U})$
end while

C Streaming Engine Implementation Details

Condition preparation and precision. The MLLM encodes the instruction and first source frame once per session. An optional reference image is VAE-encoded once, and its tokens are prefilled as globally visible KV-cache entries. Attention projections and feed-forward linear layers in the visual and text streams use FP8 (Micikevicius et al., 2022) with per-output-channel weight scaling, dynamic per-token activation scaling, and BF16 outputs. We fuse operators and use FlashAttention-based kernels (Dao et al., 2022); the full B200 configuration in Table 11 uses FlashAttention-4.

Bounded cache and CUDA graphs. The causal KV cache retains the first-chunk sink and a fixed recent history window, with rotary indices restricted to the trained range. Both conditional denoising steps reuse this context, followed by a zero-noise store pass that updates the cache from the clean generated chunk. Fixed-shape buffers and a static KV pool enable joint CUDA-graph capture of denoising, the store pass, and cache commit.

Single-frame VAE state handoff. A single boundary frame carries context between chunks without persistent internal VAE feature caches. After single-frame initialization, each steady-state encoding call processes the preceding boundary frame followed by eight new source frames and retains only the final latent frame. For decoding, the last emitted frame is re-encoded into a pseudo latent

and prepended to the current latent frame; the decoder produces nine frames, of which only the final eight are emitted. VAE calls use fixed shapes, BF16 precision, channels-last layouts, and compilation with kernel autotuning (Ansel et al., 2024), as specified in Table 8.

Asynchronous scheduling. As shown in Figure 4, bounded queues and CUDA events coordinate DiT execution with VAE processing and I/O. DiT denoising for chunk i overlaps with decoding and re-encoding of chunk $i-1$ and source encoding of chunk $i+1$ when dependencies permit. Prefetched source frames do not condition chunk i, preserving causality. Frame ingestion and output packing also overlap with model execution. For fixed resolution, chunk size, and condition length, temporal state and per-chunk work remain bounded as the stream grows.

D LongV2VBench Overview

Related work spans text-guided video editing (DecartAI Team, 2025; Liao et al., 2025), unified video modeling (Wei et al., 2025; Pan et al., 2026; Chen et al., 2026), and egocentric streaming video editing (Li et al., 2025). LongV2VBench comprises 229 editing instances on one-minute videos, all drawn from natural, real-world footage. We curate a broadly balanced set of tasks spanning background change, global style, local addition, local modification, and local removal, with the aim of reflecting practical user editing needs. These long-video tasks mimic open-ended streaming scenarios, assessing sustained editing quality, temporal consistency, and fidelity to the evolving source. Figure 12 summarizes the task distribution and shows representative inputs.

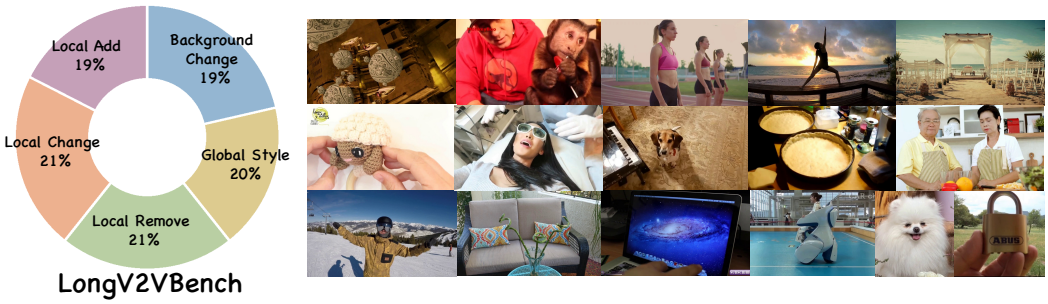

Figure 12: **LongV2VBench overview.** Task distribution and examples of five editing categories.

E Additional Experimental Results

E.1 OpenVE-Bench Quality–Throughput Comparison

We follow the five-category V2V protocol of OpenVE-Bench (He et al., 2025). A Gemini multimodal judge assigns scores from 1 to 5 for global style, local change, background change, local removal, and local addition. Figure 13 compares overall editing quality on OpenVE-Bench with DiT throughput for offline and streaming methods.

E.2 Streaming Engine Analysis

We profile VAE execution, DiT execution, and pipeline scaling on B200, RTX PRO 6000, and RTX 5090 at the resolutions specified in each table.

Deployment comparison. Table 7 compares full-pipeline, DiT, and VAE latency and throughput across streaming editors on 81-frame inputs with batch size 1. We report our own measurements of SANA-Streaming (Zhao et al., 2026) using its public implementation.

VAE execution. Table 8 compares memory-layout and compilation settings for source encoding, decoding, and pseudo encoding, with BF16 precision fixed across configurations.

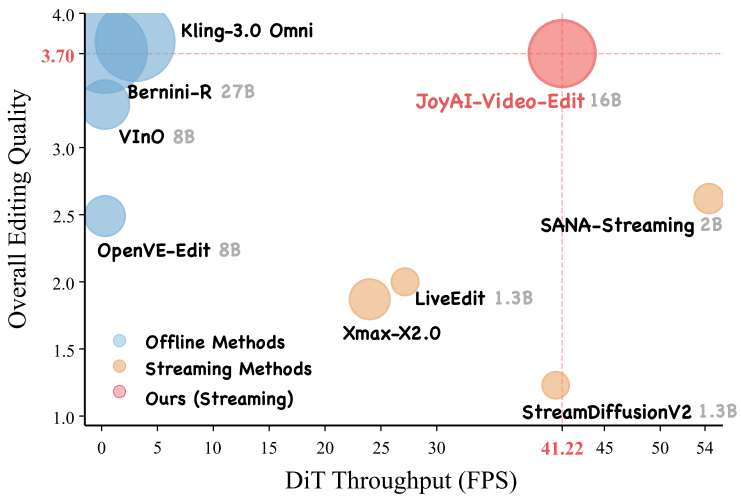

Figure 13: **Editing quality and DiT throughput on OpenVE-Bench.** The horizontal axis reports DiT throughput rather than full-pipeline throughput.

Table 7: Latency and throughput comparison of streaming video editing methods at batch size 1. Latency reports the total processing time for an 81-frame input, whereas FPS reports steady-state throughput excluding initialization. The two metrics use different timing windows; FPS is not computed from the reported clip-level latency. Bold denotes the best result in each column.

Method	Resolution	Full Pipeline		DiT		VAE	
		Latency (s) ↓	FPS ↑	Latency (s) ↓	FPS ↑	Latency (s) ↓	FPS ↑
StreamDiffusionV2	480×832	4.48	18.07	1.99	40.64	2.19	37.06
LiveEdit	480×832	5.24	15.45	2.98	27.16	2.17	37.26
SANA-Streaming	704×1280	5.58	14.51	**1.49**	**54.36**	2.99	27.12
JoyAI-Video-Edit (Ours)	720×1248	**2.32**	**34.87**	1.94	41.22	**0.37**	**217.14**

DiT execution. Table 11 compares operator fusion, numerical precision, attention backends, CUDA-graph execution, and fast accumulation. Latency includes two denoising steps and clean-KV maintenance per eight-frame chunk.

Multi-GPU pipeline scaling. Table 12 reports the complete measurements for one, two, and four GPUs, with DiT running on one GPU and Q limiting the number of chunks in flight. The captions specify timing boundaries, statistical aggregation, and speedup baselines.

Quality impact of deployment optimizations. The editing-quality results in Tables 1 and 2 use BF16 inference, whereas the Full deployment configurations in Table 11 use FP8, with fast accumulation additionally enabled on RTX 5090. As shown in Table 10, switching from BF16 to FP8 on B200 reduces the OpenVE-Bench score from 3.72 to 3.70, while both configurations score 3.61 on LongV2VBench at the reported precision. On RTX 5090, enabling fast accumulation changes the OpenVE-Bench and LongV2VBench scores from 3.60 to 3.63 and from 3.54 to 3.51, respectively. Within each GPU and resolution setting, the reported score changes are at most 0.03, indicating limited quality changes under the tested deployment optimizations.

Table 8: **Single-GPU VAE ablation.** BF16 (bfloat16) weights and autocast. Channels-Last + Compile uses `default` mode. Compile (Max-Autotune) uses `torch.compile` in `max-autotune-no-cudagraphs` mode. Full combines Channels-Last and Compile (Max-Autotune). Source Encode: one nine-frame window with posterior sampling, normalization and final-latent selection. Decode: two latent frames to nine frames, retaining the last eight. Pseudo Encode: one frame. Mean ± sample standard deviation of three run means. Standard deviations below 0.01 ms are displayed as 0.01 ms.

GPU / Resolution ($H \times W$)	Configuration	Source Encode (ms)	Decode (ms)	Pseudo Encode (ms)
B200 720×1248	BF16 Eager	186.08 ± 0.48	345.06 ± 0.16	72.86 ± 0.27
	Channels-Last	30.39 ± 0.01	57.24 ± 0.01	7.69 ± 0.01
	Channels-Last + Compile	14.18 ± 0.03	25.52 ± 0.06	3.85 ± 0.01
	Compile (Max-Autotune)	47.02 ± 0.01	69.56 ± 0.01	10.12 ± 0.01
	Full	$\mathbf{13.84 \pm 0.02}$	$\mathbf{23.46 \pm 0.11}$	$\mathbf{3.73 \pm 0.01}$
PRO6000 720×1248	BF16 Eager	212.37 ± 0.07	390.49 ± 0.18	69.85 ± 0.01
	Channels-Last	89.19 ± 0.02	165.52 ± 0.01	18.60 ± 0.01
	Channels-Last + Compile	54.49 ± 0.01	100.77 ± 0.04	13.15 ± 0.01
	Compile (Max-Autotune)	74.95 ± 0.01	117.39 ± 0.07	16.22 ± 0.01
	Full	$\mathbf{44.44 \pm 0.03}$	$\mathbf{77.56 \pm 0.07}$	$\mathbf{11.88 \pm 0.01}$
RTX 5090 480×840	BF16 Eager	131.82 ± 0.05	247.57 ± 0.10	50.74 ± 0.04
	Channels-Last	43.68 ± 0.03	79.64 ± 0.06	10.03 ± 0.01
	Channels-Last + Compile	29.29 ± 0.34	52.99 ± 0.14	7.38 ± 0.01
	Compile (Max-Autotune)	41.98 ± 0.37	67.05 ± 0.01	9.62 ± 0.01
	Full	$\mathbf{27.32 \pm 0.14}$	$\mathbf{46.21 \pm 0.01}$	$\mathbf{7.25 \pm 0.01}$

Table 9: **Pairwise human preference.** Win, Tie, and Loss denote preference for our method, equal preference, and preference for the counterpart, respectively. Bold marks the larger of Win and Loss.

Counterpart	**Win (%) ↑**	**Tie (%)**	**Loss (%) ↓**
Streaming counterparts			
LiveEdit	**90**	8	2
StreamDiffusionV2	**87**	11	2
SANA-Streaming	**87**	9	4
XMax-X2.0	**81**	13	6
Offline counterparts			
Bernini-R	**48**	8	44
Seedance 2.0	35	9	**56**
Kling-3.0 Omni	34	10	**56**

Table 10: **Editing quality under deployment optimizations.** Columns report overall benchmark scores. B200 with FP8 and RTX 5090 with FP8 and fast accumulation correspond to the respective Full configurations in Table 11.

GPU / Resolution ($H \times W$)	**DiT Precision**	**Fast Accumulation**	**OpenVE-Bench Overall ↑**	**LongV2VBench Overall ↑**
B200 720×1248	BF16	Off	3.72	3.61
	FP8	Off	3.70	3.61
RTX 5090 480×840	FP8	Off	3.60	3.54
	FP8	On	3.63	3.51

Table 11: **Single-GPU DiT ablation.** BF16: bfloat16. FP8 uses E4M3 weights and activations in image/text attention projections and feed-forward linear layers, with per-output-channel weight scales, dynamic per-token activation scales and BF16 outputs. Full uses Fused Operators, FP8 and CUDA Graph. RTX 5090 configurations all use FP8; Full also enables Fast Accumulation, using FP16 accumulation within each tile and FP32 accumulation across tiles, with a $1/16$ FP8 quantization range. Latency includes two denoising steps and clean-KV maintenance per eight-frame chunk. DiT FPS: $8000/L$ (L: mean latency in ms/chunk). Latency: mean $\pm$ sample standard deviation of three run means. Speedup is relative to the corresponding cuDNN baseline, using BF16 Eager on B200/PRO6000 and FP8 on RTX 5090.

GPU / Resolution ($H \times W$)	Configuration	Attention Backend	Latency (ms/chunk)	DiT (FPS)	Speedup ($\times$)
B200 720×1248	BF16 Eager	cuDNN	322.31 ± 0.05	24.82	1.00
	BF16 Eager	PyTorch FlashAttention	388.51 ± 0.09	20.59	0.83
	Fused Operators	cuDNN	226.23 ± 0.23	35.36	1.43
	Fused Operators	FlashAttention-4	223.74 ± 0.26	35.76	1.44
	Fused + FP8	FlashAttention-4	222.62 ± 0.14	35.94	1.45
	FP8 + CUDA Graph	FlashAttention-4	283.12 ± 0.09	28.26	1.14
	Fused + CUDA Graph	FlashAttention-4	204.13 ± 0.55	39.19	1.58
	Fused + FP8 + CUDA Graph	cuDNN	197.09 ± 0.20	40.59	1.64
	Full	FlashAttention-4	$\mathbf{194.10 \pm 0.07}$	**41.22**	**1.66**
PRO6000 720×1248	BF16 Eager	cuDNN	722.71 ± 0.02	11.07	1.00
	BF16 Eager	PyTorch FlashAttention	729.61 ± 0.23	10.97	0.99
	Fused Operators	cuDNN	617.04 ± 0.48	12.97	1.17
	Fused + FP8	cuDNN	427.52 ± 0.38	18.71	1.69
	FP8 + CUDA Graph	cuDNN	509.58 ± 0.26	15.70	1.42
	Fused + CUDA Graph	cuDNN	601.19 ± 0.69	13.31	1.20
	Full	cuDNN	$\mathbf{411.15 \pm 0.76}$	**19.46**	**1.76**
RTX 5090 480×840	FP8	cuDNN	351.15 ± 0.88	22.78	1.00
	FP8	PyTorch FlashAttention	355.35 ± 0.42	22.51	0.99
	Fused + FP8	cuDNN	304.25 ± 0.59	26.29	1.15
	Fused + FP8	SageAttention 2.2.0	288.12 ± 0.61	27.77	1.22
	Fused + FP8 + Fast Accumulation	SageAttention 2.2.0	235.58 ± 0.57	33.96	1.49
	FP8 + Fast Accumulation + CUDA Graph	SageAttention 2.2.0	256.35 ± 2.76	31.21	1.37
	Fused + FP8 + CUDA Graph	SageAttention 2.2.0	269.06 ± 0.12	29.73	1.31
	Full	SageAttention 2.2.0	$\mathbf{220.89 \pm 1.04}$	**36.22**	**1.59**

Table 12: **Pipeline scaling across GPUs.** One pipeline partitioned across GPUs; DiT runs on one GPU. Q limits chunks in flight. Throughput: mean $\pm$ sample standard deviation of FPS across three runs; speedup: relative to the single-GPU baseline of the same type. Latency is measured from submission of a complete CPU input chunk until encoded output is ready on CPU, separately at each Q; camera frame accumulation, prompt encoding and network transfer are excluded.

		Throughput ($Q = 4$)		Latency ($Q = 1$)		Latency ($Q = 4$)	
GPU / Resolution ($H \times W$)	GPU Count	Rate (FPS)	Speedup ($\times$)	P50 (ms)	P95 (ms)	P50 (ms)	P95 (ms)
B200 720×1248	1	34.87 ± 0.10	1.00	261.06	269.45	917.62	926.07
	2	40.49 ± 0.05	1.16	260.22	269.51	789.50	794.73
	4	40.44 ± 0.05	1.16	265.31	268.61	790.26	797.43
PRO6000 720×1248	1	15.20 ± 0.05	1.00	573.92	578.75	2102.19	2113.87
	2	19.37 ± 0.02	1.28	578.10	581.59	1651.02	1661.02
	4	19.39 ± 0.01	1.28	561.52	570.84	1650.43	1655.50
RTX 5090 480×840	1	27.76 ± 0.40	1.00	337.08	345.81	1159.81	1166.34
	2	32.99 ± 0.32	1.19	335.40	343.40	966.66	1180.73
	4	33.55 ± 0.31	1.21	329.18	332.96	957.85	1181.40

E.3 Human Evaluation

Table 9 summarizes pairwise preferences for anonymized outputs from single-shot videos shorter than 10 seconds, with ties permitted. Our method receives 90%, 87%, 81%, and 87% of the votes against LiveEdit, SANA-Streaming, XMax-X2.0, and StreamDiffusionV2, respectively. Against offline editors, it receives 48% against 44% for Bernini-R, while Kling-3.0 Omni and Seedance 2.0 (ByteDance Seed, 2026) each receive 56% in their respective comparisons with our model.

E.4 Additional Qualitative Results

Figures 15 and 16 provide additional examples across diverse editing tasks. Figures 17 to 19 provide additional comparisons with streaming baselines on one-minute videos. Figure 14 further shows a real-time online user session with text-guided appearance, style, and background edits applied to live camera input. JoyAI-Video-Edit follows source changes, while our asynchronous engine overlaps VAE processing, denoising, and I/O to reduce end-to-end latency. Together, our model and engine provide responsive visual feedback for interactive online editing.

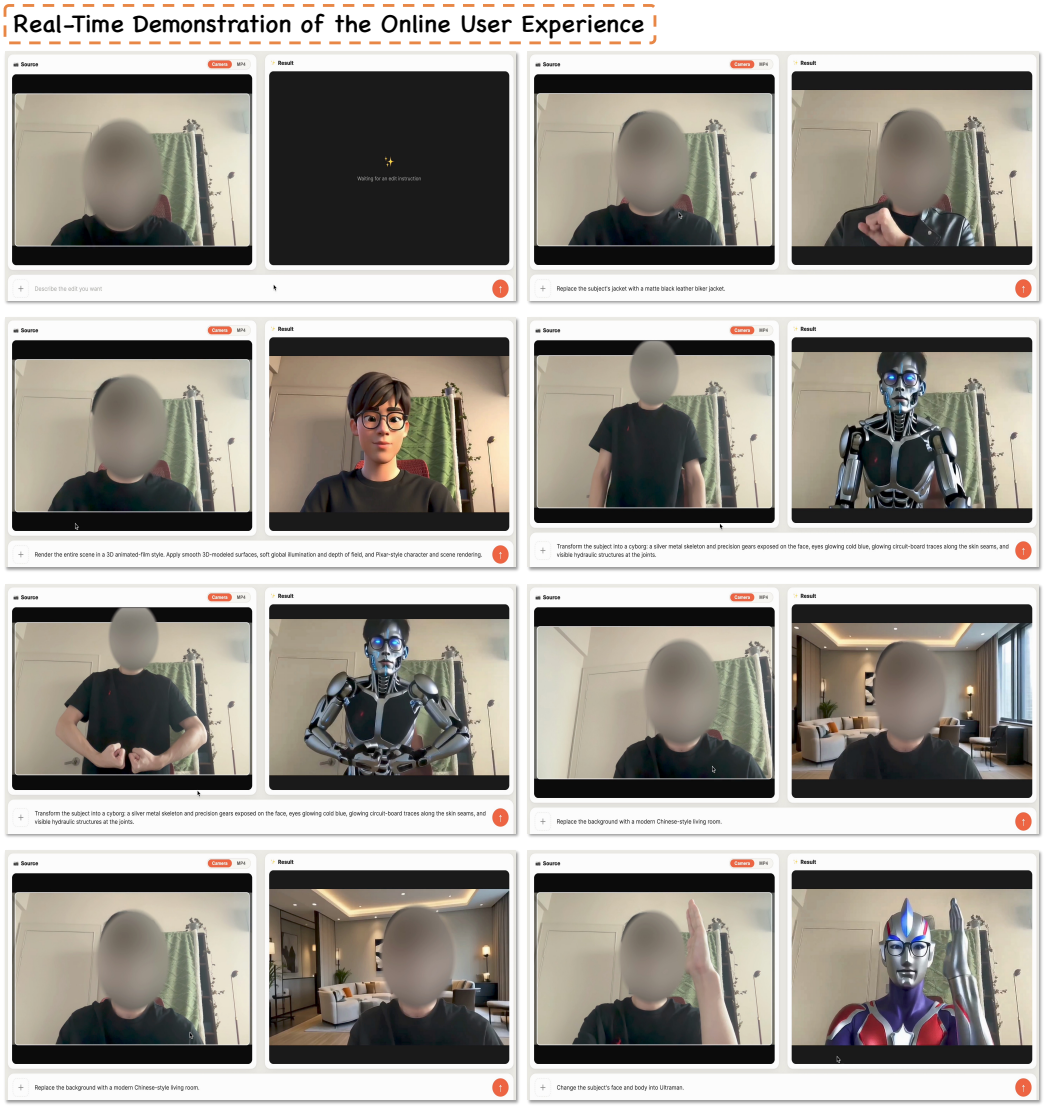

Figure 14: **Real-time online video editing in a live user session.** The interface displays live camera input alongside edited outputs for different user instructions.

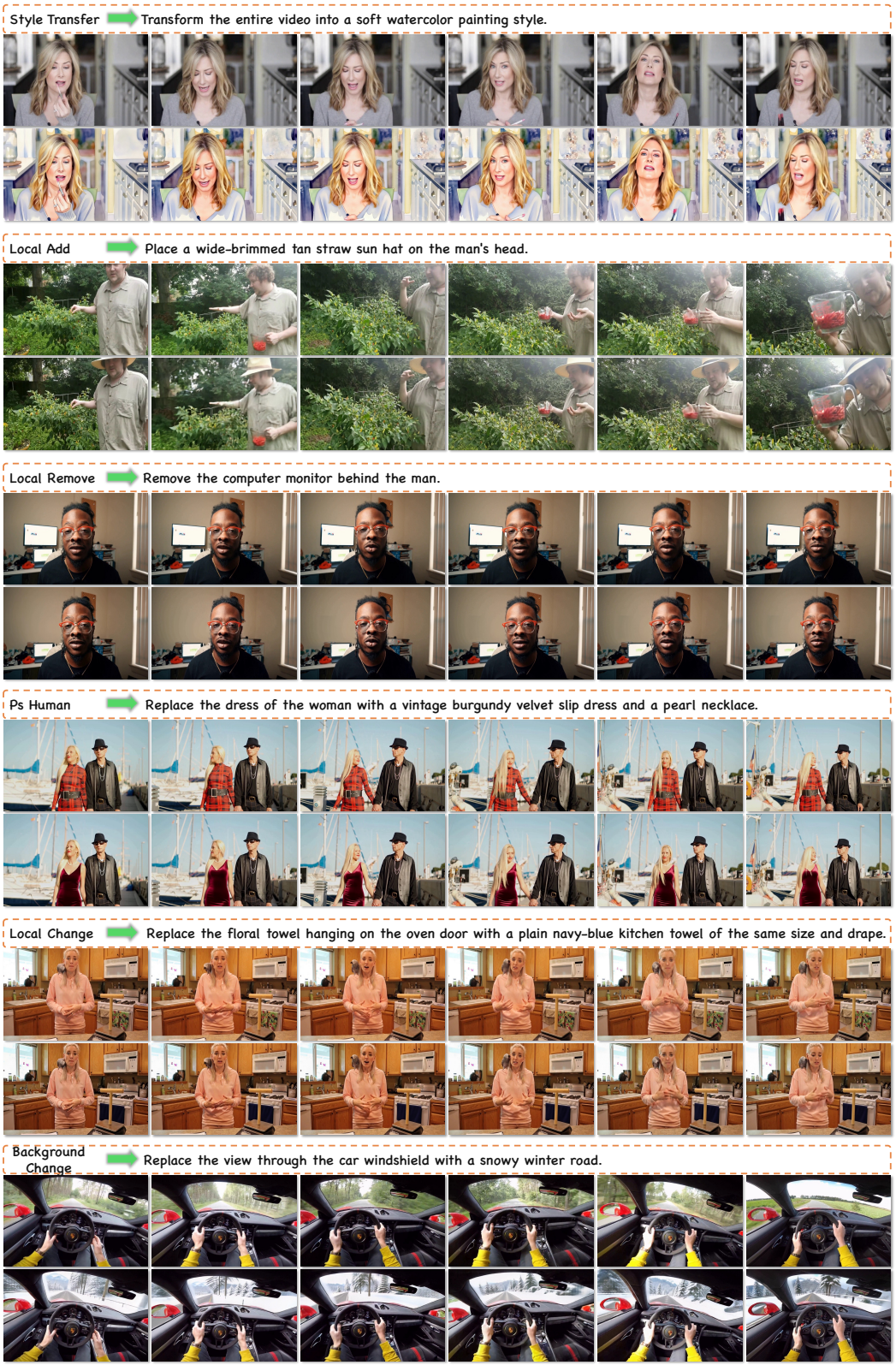

Figure 15: Additional qualitative results of JoyAI-Video-Edit across diverse editing tasks.

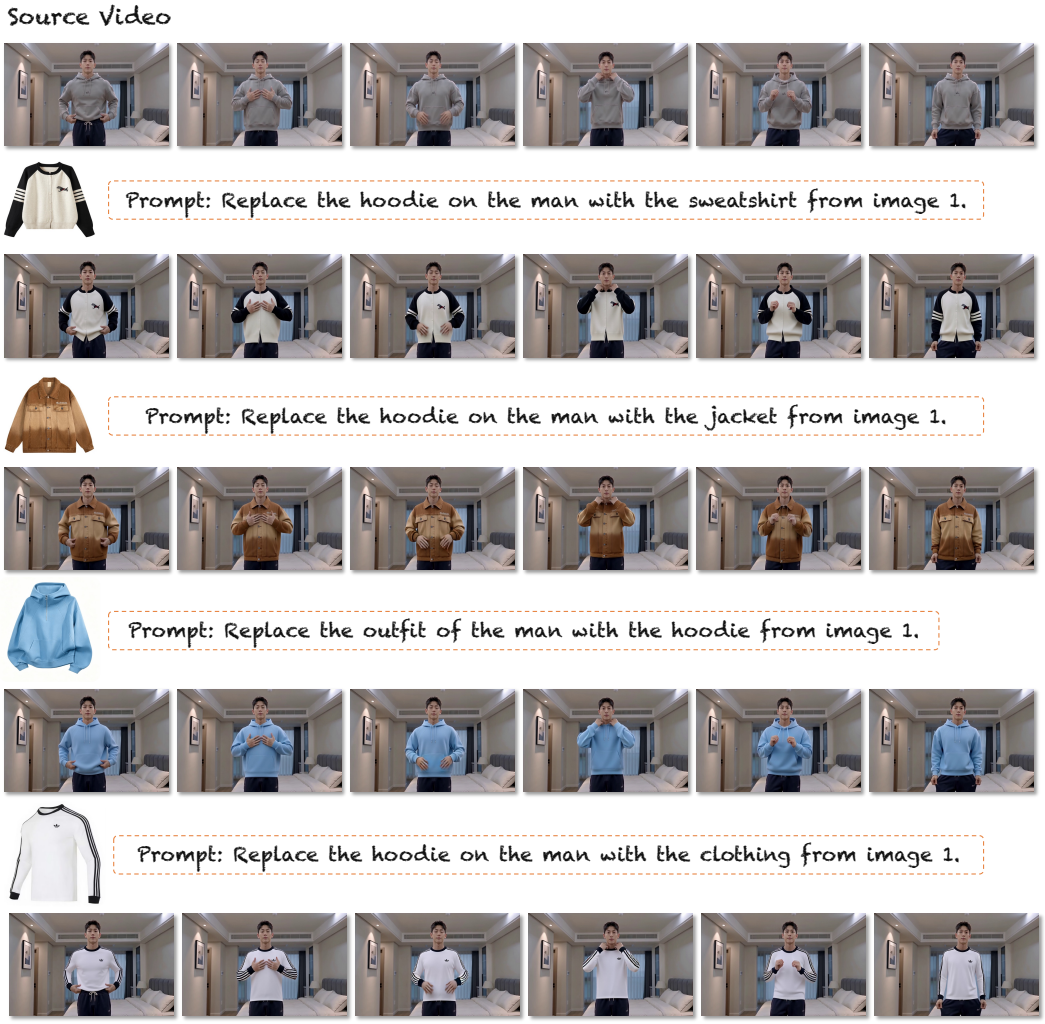

Figure 16: Additional qualitative results of JoyAI-Video-Edit across diverse reference images.



Figure 17: **Additional qualitative comparison for colored-pencil stylization.** JoyAI-Video-Edit and streaming baselines are compared on a one-minute video.



Figure 18: **Additional qualitative comparison for claymation stylization.** JoyAI-Video-Edit and streaming baselines are compared on a one-minute video.



Figure 19: **Additional qualitative comparison for text addition.** JoyAI-Video-Edit and streaming baselines are instructed to add "SUNSET HORIZON" to a one-minute video.