SteerVTE: Seamless Video Text Editing with Style and Glyph Control

Kai Zeng[1,2] **Moran Li**[2] **Zhengwei Wang**[2] **Yingchen Yu**[3,§] **Yiheng Lin**[2]
Ruichuan An[1] **Ming Lu**[1] **Qi She**[2,†] **Wentao Zhang**[1,4,5§]

[1] Peking University [2] ByteDance China [3] ByteDance Singapore [4] Zhongguancun Academy
[5] Beijing Key Laboratory of Data Intelligence and Security (Peking University)



Figure 1: **Challenges in video text editing.** Existing approaches suffer from fundamental trade-offs: frame-wise image editing [33, 57, 63] ensures text accuracy but breaks style and temporal consistency. Image-to-video methods [1, 30] improve temporal coherence but sacrifice content fidelity. General video editing [11, 38, 62] lacks text priors and often produces garbled text. Our method tackles these trade-offs, enabling accurate, temporally consistent, and style-preserving video text editing.

Abstract

Visual text editing aims to precisely modify text in images and videos while preserving stylistic consistency and visual realism. Despite significant advances in the image domain, video text editing remains largely unexplored: it is a localized task demanding stroke-level precision within small text regions, which compounds the challenges of cross-frame accuracy, temporal coherence, and stylistic fidelity. We introduce SteerVTE, a unified framework that **steer**s a frozen video diffusion model to perform precise **V**ideo **T**ext **E**diting through style and glyph control. Built on a frozen diffusion transformer, SteerVTE attaches a lightweight text context adapter with two complementary modules: a style encoder capturing the original text's visual attributes, and dual-granularity glyph encoders encoding the target text at both the line and character levels. To overcome the inherently weak text rendering priors of video foundation models, we further propose a glyph-aware spatial-focal loss and a three-stage progressive training curriculum that scales from image to video data. To support large-scale training, we also develop an automatic synthesis

[†]Project Leader.
[§]Corresponding Authors. yingchen001@e.ntu.edu.sg and wentao.zhang@pku.edu.cn.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

pipeline and construct SteerVTE-1M, a dataset of one million triplets spanning diverse scenes, fonts, and stylistic effects. Extensive experiments demonstrate that SteerVTE substantially outperforms existing video editing baselines across text accuracy, style consistency, and temporal coherence. Our project page is available at `https://zengkaiya.github.io/SteerVTE/`.

1 Introduction

Visual text editing aims to modify text in images or videos while ensuring accurate and seamless background integration. This task has broad applications, including advertisement design, film post-production, and subtitle correction. It requires models to understand text attributes such as font, size, and color, and to render glyphs accurately. In the image domain, advances in diffusion models [15, 25, 54] and powerful foundations like Stable Diffusion 3 [16], FLUX.1 [32], and Qwen-Image [63] have enabled strong text editing performance with minimal task-specific fine-tuning.

However, in the video domain, seamless text editing demands not only accurate modifications and style consistency but also temporal coherence across frames. As illustrated in Fig. 1, straightforward solutions all fall short. Applying image text editing frame-by-frame ignores cross-frame dependencies, inducing temporal flickering and style drift. Editing only the first frame and propagating it via image-to-video (I2V) generation enforces temporal consistency, but without conditioning on subsequent source frames, the model hallucinates motion and scene dynamics, breaking fidelity to the original. End-to-end video-to-video (V2V) approaches appear promising, yet pre-trained video foundation models (e.g., LTXVideo [23], Wan [60], HunyuanVideo [30]) are optimized primarily for general scene generation rather than fine-grained text rendering. Consequently, V2V models built on these backbones tend to struggle with the character-level precision that text editing requires. Meanwhile, existing controllable video editing methods [5, 11, 28, 38, 62] are tailored for global or object-level edits such as stylization and object insertion/deletion, and consequently offer limited support for capturing small-region text styles or generating stroke-level glyphs.

To break this impasse, we introduce **SteerVTE**, a framework that fundamentally rethinks video text editing along three axes: model architecture, training strategy, and large-scale data synthesis. At its core is a simple yet pivotal insight: rather than retraining a video diffusion backbone and risking the collapse of its rich generative prior, we freeze a pre-trained text-to-video DiT and, through a lightweight Text Context Adapter, infuse it with exactly the missing editing awareness. This adapter delivers three carefully designed signals: (1) a spatiotemporal mask for precise region localization; (2) a style encoder, built on a native-resolution Vision-Language Model, for extracting clean stylistic features from the cropped editing region; (3) a dual-granularity glyph module fusing sentence-level and character-level representations for accurate text rendering. Together, these signals equip the frozen backbone with the region awareness, style fidelity, and content accuracy, enabling seamless video text editing across spatial, temporal, and semantic dimensions.

Existing image text editors still struggle when scaling from single-word to sentence-level edits and small-font scenarios. We address this limitation with an **GL**yph-**A**ware **S**patial-focal Loss (**GLAS Loss**) that concentrates supervision on text regions and provides character-level guidance. To overcome the inherently weak text rendering in video foundation models, we introduce a progressive **three-stage curriculum** that bridges image and video domains: the first stage establishes basic glyph priors on simple image text editing data; the second stage introduces complex real-world image data to learn font, color, and layout variations; the final stage fine-tunes on video data with the flow matching loss and GLAS loss for temporally stable editing. This curriculum learning strategy progressively builds robust text editing capabilities and avoids the difficulty of simultaneously mastering glyph rendering, stylistic fidelity, and temporal coherence directly on video data.

The three-stage curriculum above requires a corresponding data strategy, yet large-scale real video text editing pairs are unavailable. We therefore design a fully automatic synthesis pipeline with four stages: text configuration sampling, paired subtitle texture rendering (same style, different text), random video compositing, and OCR-based filtering to ensure clear and recognizable text regions. This pipeline involves no generative models, guaranteeing pixel-perfect editing accuracy and exact background consistency. However, models trained solely on synthetic data generalize poorly to real footage. To bridge this gap, we incorporate real-world image editing data derived from a raw OCR dataset, which provides stylistic diversity not present in synthetic renderings. Together, the two

sources form **SteerVTE-1M**, a one-million-triplet dataset that gives the curriculum both precise supervision from synthetic data and realistic appearance variations from real scenes.

Beyond training data, evaluation is equally lacking: no dedicated benchmark exists for video text editing. We therefore construct **VTE-Bench**, the first benchmark of its kind, comprising 100 synthetic videos, 100 real-world videos, and 200 challenging scene text videos. VTE-Bench systematically evaluates text accuracy, style consistency, temporal coherence, and background preservation. Quantitative and qualitative results show that SteerVTE achieves state-of-the-art performance, outperforming all baselines. In particular, it achieves more than **twice** the Sentence Accuracy of the proprietary model Seedance 2.0 (77% vs. 32%), along with a 34-point improvement in NED (95% vs. 61%) and a 25-point improvement in Style-Sim (59% vs. 34%).

We summarize our contributions as follows:

- We present SteerVTE, to our knowledge, the first **end-to-end** model for seamless video text editing. Built on a frozen DiT with a Text Context Adapter, it integrates a VLM-based style encoder and a dual-granularity glyph module to deliver style-preserving and content-accurate edits.
- We propose GLAS Loss and a three-stage curriculum to counteract the deficient text priors in video foundation models, jointly improving performance on sentence-level and small-region text editing.
- To address the lack of training data and standardized evaluation, we construct SteerVTE-1M, a one-million-triplet dataset combining synthetic and real-world supervision, and VTE-Bench, the first benchmark for video text editing spanning diverse scenarios.

2 Related Work

Visual Generation Foundation Models. Diffusion models [15, 25, 50, 54] have driven significant progress in image and video generation. More recently, scalable diffusion transformers [47] and flow-matching [16] have further improved generation quality and efficiency, enabling large-scale image foundation models (e.g., Flux [32], Qwen-Image [63]) and video foundation models (e.g., HunyuanVideo [30], Wan [60]), which serve as backbones for a wide range of downstream tasks.

Image Text Generation and Editing. The practical value of glyphs in applications like advertisements and films has driven growing interest in visual text generation and editing. Most existing works [8, 9, 41, 53, 57, 58, 72] follow pathways similar to those for natural images, typically fine-tuning diffusion models [50] in a parameter-efficient manner [26]. Fine-grained text variations, including color, font, and layout, have inspired diverse research focuses. For instance, DiffUTE [7] and GlyphDraw [42] embed glyph images into text embeddings, while Anytext2 [57] uses multiple encoders to capture fine-grained attributes. More recently, FLUX-Text [33], UM-Text [43], and UTDesign [71] adopt DiT architectures for enhanced text generation. Nevertheless, current image text editing models remain limited to single-word editing, and video text editing remains underexplored.

Controllable Video Editing. Video editing tasks have evolved alongside advances in diffusion models. Early training-free methods [12, 17, 19, 29, 31, 35, 48] perform DDIM inversion and denoising without training but suffer from limited generalization, high latency, and reliance on paired captions. One-shot methods [2, 3, 22, 45, 46, 64] overfit to individual videos for better quality yet require costly per-video optimization. With text-to-video (T2V) and I2V foundation models [30, 60, 66], recent works [6, 10, 27, 40, 67, 70] build large-scale datasets for training. Leveraging MLLM-DiT architectures, these methods [36, 38, 56, 62] achieve powerful multimodal reference generation under joint instruction and visual guidance. However, existing video generation models have extremely limited text rendering capabilities, let alone fine-grained editing that preserves content and style consistency. Another line of work, STRIVE [55], adopts a cascaded pipeline: it sequentially detects text regions of interest (ROIs), selects a reference frame, applies an image-based text replacement model, and propagates the edited reference result to other frames through geometry modeling. Its text editing capability therefore comes from an external image editor, and it is prone to cascading errors. To address this gap, we propose the first end-to-end video text editing model.

3 Methodology

The overview of SteerVTE is illustrated in Fig. 2. We first detail the SteerVTE model architecture (Sec. 3.1) and its designated optimization objective, the GLyph-Aware Spatial-focal (GLAS) Loss

(Sec. 3.2). Subsequently, we introduce the supporting SteerVTE-1M dataset (Sec. 3.3) and the progressive Text Editing Curriculum Learning strategy tailored for stable optimization (Sec. 3.4).

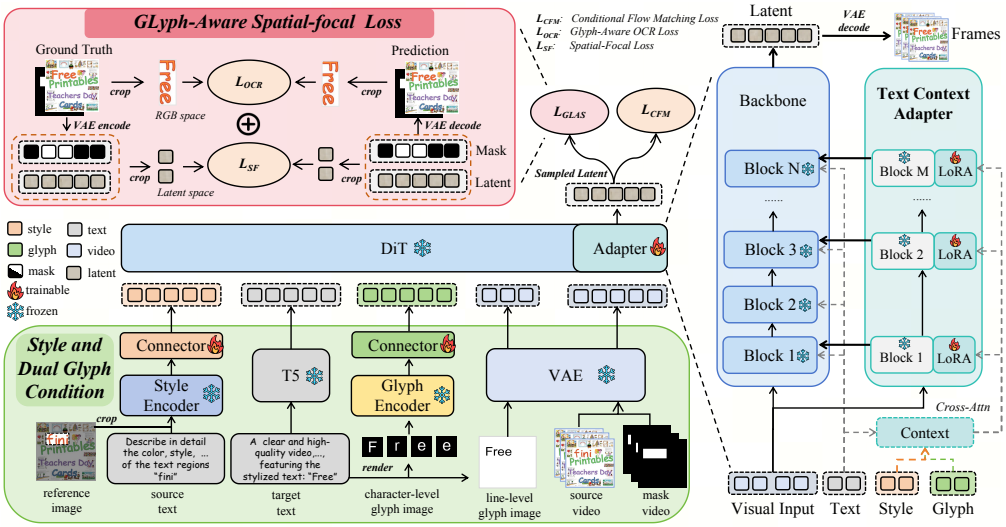

Figure 2: Overview of the proposed **SteerVTE**. A frozen text-to-video DiT is augmented with a lightweight Text Context Adapter that injects three editing signals via cross-attention: a spatiotemporal mask for region localization, VLM-encoded style features from a reference image cropped from the corresponding masked frame, and dual-granularity (line- and character-level) glyph features of the target text. The GLAS Loss further combines latent-space mask reweighting and RGB-space OCR supervision to improve sentence-level and small-region editing.

3.1 SteerVTE Model

3.1.1 Preliminaries and Notation

Text-to-Video Generation Prior. Text-to-video generation models are typically built upon Flow Matching (FM) [39] with a DiT backbone v_θ. To alleviate the computational overhead of pixel-space, these models operate in the latent space of a pretrained Variational Autoencoder (VAE) (i.e., encoder $\mathcal{E}$, decoder $\mathcal{D}$). Given a source video $\mathbf{V}_{\text{src}}$, Gaussian noise $\mathbf{z}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and a timestep $t \in [0, 1]$, the noisy latent $\mathbf{z}_t$ is constructed via interpolation $\mathbf{z}_t = t\mathbf{z}_1 + (1-t)\mathbf{z}_0$, where $\mathbf{z}_1 = \mathcal{E}(\mathbf{V}_{\text{src}})$. Conditioned on a text prompt $\mathbf{c}$, v_θ is optimized to estimate the velocity field driving the transition from noise to real latents. The general flow-matching objective is formulated as:

$$\mathcal{L}_{\text{FM}} = \|v_\theta(\mathbf{z}_t, t; \mathbf{c}) - (\mathbf{z}_1 - \mathbf{z}_0)\|_2^2. \tag{1}$$

Context Adapter. To leverage strong pretrained text-to-video priors for controllable generation, additional spatiotemporal conditions $\mathbf{C}$ are incorporated. Instead of employing computationally expensive parallel architectures like ControlNet [69], we adopt a parameter-efficient approach utilizing Low-Rank Adaptation (LoRA) [26]. Specifically, a lightweight adapter $\mathcal{A}_\phi$ extracts features from $\mathbf{C}$, which are subsequently injected into the DiT modules. The conditional flow-matching loss is:

$$\mathcal{L}_{\text{CFM}} = \|v_\theta(\mathbf{z_t}, t; \mathbf{c}, \mathcal{A}_\phi(\mathbf{C})) - (\mathbf{z}_1 - \mathbf{z}_0)\|_2^2. \tag{2}$$

3.1.2 Text Context Adapter

While general adapters suffice for coarse spatial control, they struggle in fine-grained video text editing, failing to precisely localize text, losing typographical attributes (e.g., font, color, and stroke), and yielding blurred or misspelled glyphs. To overcome this, we introduce the *Text Context Adapter*, a dedicated conditioning module that jointly encodes positional, stylistic, and glyph-level cues (Fig. 2).

Mask-Guided Positional Conditioning. To enable precise foreground editing while preserving background fidelity, following VACE [28], we decompose the source video $\mathbf{V}_{\text{src}}$ into foreground and background components via mask $\mathbf{M}$: $\mathbf{V}_{\text{f}} = \mathbf{V}_{\text{src}} \odot \mathbf{M}$, $\mathbf{V}_{\text{b}} = \mathbf{V}_{\text{src}} \odot (\mathbf{1} - \mathbf{M})$. Both components

are encoded by the VAE encoder $\mathcal{E}$ and channel-wise concatenated with the down-sampled mask $\tilde{\mathbf{M}}$ for spatiotemporal alignment, yielding the positional condition: $\mathbf{C}_{\text{pos}} = \text{Concat}(\mathcal{E}(\mathbf{V}_{\text{b}}), \mathcal{E}(\mathbf{V}_{\text{f}}), \tilde{\mathbf{M}})$.

Native-Resolution VLM for Style Encoding. Faithful style transfer relies on preserving the geometric and textural cues of source typography, such as stroke thickness, character spacing, and aspect ratio, which are highly sensitive to spatial distortion. Prior methods learn style implicitly via masked inpainting, lacking explicit control. To inject target style features, we introduce a dedicated style encoder. However, conventional encoders like CLIP [49] or DINO [44] operate on square-resized inputs, warping the elongated geometry of text and collapsing fine-grained typographical cues. We argue that this limitation stems not from feature quality, but from fixed-resolution tokenization. Instead, we leverage native-resolution Vision-Language Models (VLMs) like Qwen2.5-VL [4], which tokenize images at their original aspect ratio and thus preserve the intrinsic spatial statistics of text, making them naturally suited for capturing typographic style across arbitrary shapes and orientations. Assuming stylistic temporal consistency within consecutive frames, we extract style from an arbitrary frame k: $\mathbf{C}_{\text{style}} = \text{VLM}(\mathbf{V}^k_{\text{src}} \odot \mathbf{M}^k, P(\mathbf{T}_{\text{src}}))$, where $P(\cdot)$ is a prompt template steering the VLM toward the stylistic attributes of the source text $\mathbf{T}_{\text{src}}$. The final two VLM hidden states are concatenated and projected via a style connector for cross-attention into the v_θ.

Dual-Granularity Glyph Control. Accurate text rendering hinges on three coupled questions: *where* each character is placed, *what* its strokes look like, and *how* it is stylized. Stylization is handled by the VLM style encoding introduced previously, whereas structural placement and stroke fidelity remain underserved by the diffusion backbone. We therefore inject explicit glyph guidance at two complementary granularities, both rendered in a *single fixed neutral font* so that the branches expose only content and structure, while font style is modeled exclusively by the VLM style encoder. *i). Line-level layout prior.* We render the target text $\mathbf{T}_{\text{tgt}}$ to obtain $\mathbf{I}_{\text{layout}} = \mathcal{R}_{\text{line}}(\mathbf{T}_{\text{tgt}})$, encode it via the VAE to $\mathbf{C}_{\text{layout}} = \mathcal{E}(\mathbf{I}_{\text{layout}})$, and concatenate it with the positional condition, $\mathbf{C} = \text{Concat}(\mathbf{C}_{\text{pos}}, \mathbf{C}_{\text{layout}})$, yielding a spatially aligned, style-agnostic prior that dictates *where* glyphs appear. *ii). Character-level stroke prior.* Since VAE compression inevitably erodes fine strokes, the layout prior alone cannot prevent character-level distortions. We thus render each character $g_i \in G_{\text{tgt}} = \{g_1, \ldots, g_N\}$ individually. To explicitly capture the canonical stroke topology, we encode characters using a pre-trained OCR network $\mathcal{O}$ [14] and pass through a glyph connector. The resulting stroke features are then injected via cross-attention: $\mathbf{C}_{\text{stroke}} = \text{Proj}_{\text{glyph}}\big(\text{Concat}_{i=1}^{N}\, \mathcal{O}(\mathcal{R}_{\text{char}}(g_i))\big)$. Sharing the neutral font ensures this prior provides precise *what* guidance without leaking specific typeface details.

Altogether, the Text Context Adapter aggregates position, style, and glyph signals into a unified conditioning representation. The base training objective is the conditional flow-matching loss $\mathcal{L}_{\text{CFM}}$.

3.2 GLyph-Aware Spatial-focal Loss

Standard flow matching optimizes a training objective exclusively within the latent space, treating all spatial locations indiscriminately. This inherently introduces two critical bottlenecks for precise text editing: *i). optimization bias*, in that gradients originating from spatially sparse textual regions are suppressed by the dominant non-text background; and *ii). structural ambiguity*, i.e., global distribution matching in the latent space provides no explicit inductive bias toward fine-grained character topology; To overcome these limitations, we propose the **GL**yph-**A**ware **S**patial-focal (GLAS) loss (see Fig. 2). GLAS instantiates a synergistic dual-space paradigm: a *glyph-aware* OCR-based supervision in the pixel space that enforces character-level structural fidelity, coupled with a *spatial-focal* reweighting in the latent space that prioritizes text-bearing regions.

Spatial-Focal Loss. To counteract spatial gradient suppression, we reweight the flow matching objective with a normalized focal map $\mathbf{W} \in [0, 1]$ derived from the spatial mask $\mathbf{M}$, concentrating the optimization on text regions:

$$\mathcal{L}_{\text{SF}} = \big\|\mathbf{W} \odot \big(v_\theta(\cdot) - (\mathbf{z}_1 - \mathbf{z}_0)\big)\big\|_2^2. \tag{3}$$

Glyph-Aware OCR Loss. The spatial-focal loss prioritizes text regions yet remains blind to glyph structure, since latent-space supervision cannot reach character-level topology. We thus impose pixel-space supervision via an off-the-shelf OCR recognizer $\mathcal{O}$. Specifically, the predicted latent $\hat{\mathbf{z}}_0$ is decoded and temporally sampled to obtain aligned frame triplets: $(\mathbf{I}_{\text{gt}}, \mathbf{I}_{\text{pred}}, \mathbf{I}_{\text{m}}) = \mathcal{S}\big(\mathbf{V}_{\text{gt}},\, \mathcal{D}(\hat{\mathbf{z}}_0),\, \mathbf{M}\big)$, where $\mathcal{S}(\cdot)$ uniform temporal sampling. Upon these triplets, we enforce glyph

correctness at two complementary granularities:

$$\mathcal{L}_{\text{OCR}} = \underbrace{\left\|\mathcal{O}(\mathbf{I}_{\text{pred}}) - \mathcal{O}(\mathbf{I}_{\text{gt}})\right\|_2^2}_{\mathcal{L}_{\text{text}}\text{: stroke-level fidelity}} + \underbrace{\text{CTC}\big(\mathcal{O}(\mathbf{I}_{\text{pred}}),\, y\big)}_{\mathcal{L}_{\text{CTC}}\text{: character-level recognizability}}, \tag{4}$$

where y denotes the target transcription and $\text{CTC}(\cdot)$ the connectionist temporal classification objective [21]. The choice of recognizer and full formulations are deferred to Appx. B.2.

Hence, with empirical hyperparameters ($\gamma = 3.0$, $\lambda = 0.1$), the overall GLAS Loss is formulated as:

$$\mathcal{L}_{\text{GLAS}} = \gamma \mathcal{L}_{\text{SF}} + \lambda \mathcal{L}_{\text{OCR}}. \tag{5}$$

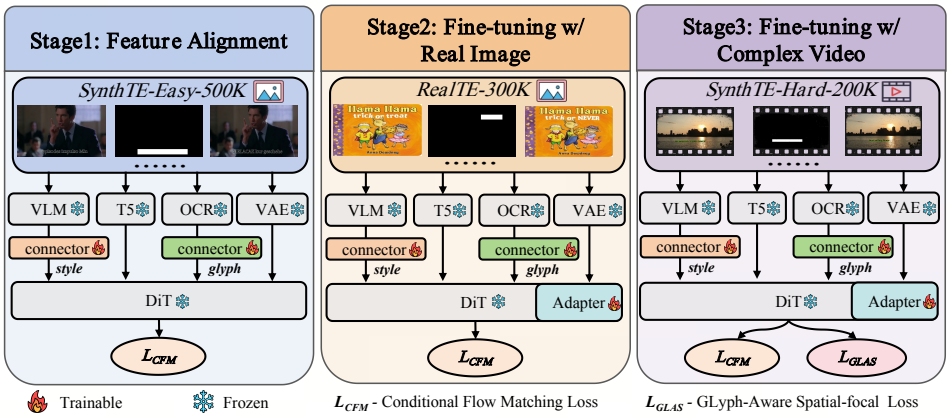

Figure 3: **Illustration of the three training stages.** In the first stage, the connector is primarily trained to align style and glyph features. In the second stage, the connector and DiT are jointly trained. In the third stage, the GLAS loss is introduced, and training proceeds on the video dataset.

3.3 Dataset Construction

To empower end-to-end video text editing, we construct SteerVTE-1M, a large-scale dataset comprising two complementary components: SynthTE (stylized synthetic data) and RealTE (complex real-world data). Detailed construction pipelines are provided in Appx. B.1.

SynthTE Dataset. To compensate for the weak text-rendering capabilities of video foundation models, we synthesize data with varying difficulty levels. We collect 1,933 TrueType fonts from Google Fonts [20], an English text corpus [18] of 65 letters and symbols, and background videos from OpenSora-Plan [37]. The easy version uses simple fonts with short text and large font sizes. The hard version uses all available fonts, including artistic and textured styles, featuring longer text and smaller font sizes. As illustrated in Fig. 5(a), the pipeline randomly samples text attributes and paired text to render subtitle images, selects background videos, and generates bounding boxes covering 8% to 15% of the frame for pasting. It inserts one to four subtitle pairs, selects one as the training region, and strictly retains only samples that achieve 100% OCR recognition accuracy. This process produces 500k image pairs for SynthTE-Easy and 200k video pairs for SynthTE-Hard.

RealTE Dataset. Given the scarcity of real-world video text editing data, we construct an image-based dataset derived from AnyWord-3M [58]. We filter 500k samples with fully legible text via PP-OCRv3, edit them using a state-of-the-art text editing model [68], and reapply OCR verification. To preserve stylistic consistency, we employ a Text Style Similarity metric [52] and discard samples with scores below 0.4. This pipeline yields the RealTE Dataset, comprising 300k image pairs.

3.4 Text Editing Curriculum Learning

Training video text editing from scratch is challenging. We therefore propose a progressive three-stage curriculum, leveraging our SteerVTE-1M dataset, to systematically transition from fundamental glyph alignment to complex temporal dynamics.

Stage 1: Feature Alignment. To establish an initial visual-textual mapping, we freeze the style and glyph encoders, training only their linear connectors. The Text Context Adapter is optimized via the standard flow-matching loss on 500k synthetic images (SynthTE-Easy).

Stage 2: In-the-Wild Image Fine-Tuning. To bridge the synthetic-to-real domain gap, we activate LoRA modules within the adapter. Both the connectors and LoRAs are jointly optimized on 300k real-world samples (RealTE), adapting the model to complex typographical layouts that synthetic data inherently lacks. Although the ground truth in this subset is generated by an image text editing model, rigorous OCR and style filtering ensure sample quality. We validate the effectiveness in Tab. 5.

Stage 3: Temporal Video Fine-Tuning. Image-only pretraining inevitably introduces temporal flickering. To enforce kinematic coherence, we extend training to 200k video clips (SynthTE-Hard). Having acquired robust foundational character priors from the previous stages, we seamlessly integrate the GLAS loss here to explicitly enforce spatial focus and fine-grained glyph fidelity. The ultimate optimization objective is: $\mathcal{L} = \mathcal{L}_{\text{CFM}} + \mathcal{L}_{\text{GLAS}}$.

4 Experiments

4.1 Experimental Setup

Benchmark and Metrics. We construct **VTE-Bench**, a benchmark for video text editing, comprising VTE-Bench-Synth (50 easy and 50 hard synthetic samples with ground-truth edited videos), VTE-Bench-Real (100 real-world samples without ground truth), and VTE-Bench-SceneText (200 hard scene text videos from STRIVE [55] and the web). Each sample provides a source video, editing mask, and editing prompt. We evaluate *text accuracy* via PP-OCRv4 (precision, recall, F-score, NED, sentence accuracy) and *video quality* via SSIM, PSNR, FID, Style-Sim, FVD, plus background preservation metrics SSIM_{bg} and PSNR_{bg}. The evaluation OCR (PP-OCRv4) differs from the PP-OCRv3 used during training (Sec. 3.2) to ensure independent measurement. A user study on VTE-Bench-Real provides an additional Quality Rate. More details are provided in Appx. C.1.

Implementation. In Stage 1, we initialize the style encoder, character glyph encoder, and backbone with Qwen2.5-VL-3B [4], PP-OCRv3 [14], and VACE 14B [28], respectively, and train only the two randomly-initialized connectors. In Stages 2 and 3, we additionally fine-tune the Text Context Adapter via LoRA [26] (rank 64). Full details in Appx. C.2.

4.2 Quantitative Evaluation

Table 1: **Quantitative comparison on VTE-Bench-Synth.** SteerVTE consistently outperforms other approaches across all metrics, including OCR accuracy and style consistency. **Bold** is the best.

Method	Text Editing					Video Quality						
	Precision↑	Recall↑	F-Score↑	NED↑	Accuracy↑	SSIM↑	PSNR↑	FID↓	FVD↓	Style-Sim↑	SSIM_{bg}↑	PSNR_{bg}↑
FLUX-Text [33]	0.5601	0.5303	0.5447	0.2790	-	0.2268	11.02	149.7	314.9	0.4273	0.9589	36.14
VACE [28]	0.6168	0.6082	0.6124	0.1533	-	0.2841	11.54	105.5	14.80	0.4827	0.9579	35.79
VideoPainter [5]	0.1653	0.1111	0.1328	0.0367	-	0.1313	7.463	206.0	62.24	0.2446	0.3070	8.031
VIVA [11]	0.3446	0.2933	0.3168	0.1126	-	0.3148	11.99	160.1	30.88	0.3570	0.9007	28.04
UniVideo [62]	0.5480	0.5496	0.5487	0.1523	-	0.2162	10.74	104.8	15.54	0.4476	0.6446	21.19
Kiwi-Edit [38]	0.5417	0.4839	0.5111	0.1280	-	0.2078	10.58	118.7	18.81	0.4268	0.7265	21.85
Seedance 2.0 [51]	0.9416	0.9144	0.9278	0.8628	0.3300	0.3203	12.10	121.1	54.71	0.3442	0.7015	22.30
SteerVTE (Ours)	**0.9908**	**0.9825**	**0.9866**	**0.9624**	**0.6100**	**0.4034**	**12.30**	**65.46**	**14.55**	**0.5936**	**0.9606**	**37.09**

We compare SteerVTE with frame-by-frame image editors and video editors. Image editors include FLUX-Text [33]; video editors include mask-based (VACE [28], VideoPainter [5]), instruction- and reference-guided (VIVA [11], UniVideo [62], Kiwi-Edit [38]), and closed-source (Seedance 2.0 [51]) models. Results on VTE-Bench-Synth and VTE-Bench-Real are shown in Tab. 1 and Tab. 2. For text accuracy, SteerVTE achieves near-perfect Precision, Recall, F-Score, and NED. On the stricter sentence-level accuracy, which requires exact character matching, SteerVTE reaches 61% on Synth and 77% on Real, more than doubling Seedance 2.0 with only 33% and 32%, while other video baselines lack text priors and perform poorly. For style and temporal coherence, SteerVTE leads on foreground style metrics and achieves an FVD of 14.55 on Synth, whereas FLUX-Text's per-frame editing causes severe flickering reflected by an FVD of 314.9. Our mask conditioning further enables near-perfect background preservation. Since Seedance 2.0 is the only baseline producing competitive editing results, we further conduct a user study to measure Quality Rate, defined as the fraction of videos judged visually satisfactory. SteerVTE achieves 72% compared to only 23% for Seedance 2.0.

In addition, as shown in Tab. 3, although our model is trained solely on real image data and synthetic video data, it generalizes to a wide range of static, dynamic, and deformable scene text cases.

Table 2: **Quantitative comparison on VTE-Bench-Real.** Other methods perform poorly under complex real-world scenarios, whereas SteerVTE achieves strong performance. **Bold** is the best.

Method	Text Editing					Video Quality		
	Precision↑	Recall↑	F-Score↑	NED↑	Accuracy↑	$SSIM_{bg}$ ↑	$PSNR_{bg}$ ↑	Quality Rate↑
FLUX-Text [33]	0.6221	0.6058	0.6138	0.3808	-	0.9609	33.40	-
VACE [28]	0.4224	0.4185	0.4204	0.1082	-	0.9626	34.33	-
VideoPainter [5]	0.0667	0.0385	0.0488	0.0113	-	0.3044	7.965	-
VIVA [11]	0.4030	0.3704	0.3860	0.1680	-	0.8688	22.26	-
UniVideo [62]	0.3783	0.3910	0.3845	0.1197	-	0.7399	20.66	-
Kiwi-Edit [38]	0.3892	0.3651	0.3767	0.0978	-	0.7277	20.24	-
Seedance 2.0 [51]	0.7577	0.7400	0.7487	0.6106	0.3200	0.7779	22.37	0.2300
SteerVTE (Ours)	**0.9785**	**0.9769**	**0.9776**	**0.9579**	**0.7700**	**0.9648**	**35.55**	**0.7200**

Table 3: **Quantitative comparison on VTE-Bench-SceneText.** Our method generalizes to unseen scene text scenarios. **Bold** indicates the best result. STRIVE* denotes our reproduced implementation.

Method	Text Editing					Video Quality		
	Precision↑	Recall↑	F-Score↑	NED↑	Accuracy↑	$SSIM_{bg}$ ↑	$PSNR_{bg}$ ↑	Quality Rate↑
STRIVE* [55]	0.5014	0.4943	0.4889	0.3834	0.0700	**0.9794**	**38.98**	0.0100
TextCtrl [68]	0.7069	0.6670	0.6746	0.5927	0.3300	**0.9794**	**38.98**	0.0600
AnyText2 [57]	0.5727	0.5850	0.5636	0.4337	0.1000	0.8291	23.23	0.0100
FLUX2 [32]	0.5581	0.5277	0.5332	0.4413	0.2000	0.6677	18.17	0.0700
SteerVTE (Ours)	**0.7631**	**0.7494**	**0.7475**	**0.6908**	**0.5100**	0.9682	36.21	**0.4100**

4.3 Qualitative Evaluation

We conduct qualitative comparisons with state-of-the-art methods, including FLUX-Text, VACE, VIVA, UniVideo, Kiwi-Edit, and Seedance 2.0 on real-world scenarios (Fig. 4). Our method achieves precise text editing, consistent text style, and consistent background. Seedance 2.0 produces reasonable results but lacks perfect accuracy and makes erroneous background modifications. Other video editing models generally lack text editing capabilities: they either leave text unmodified or exhibit deficiencies in accuracy, style, and background consistency, sometimes even drastically altering the entire video. More examples of scene text editing results can be found in Appx. F.

4.4 Ablation Studies

4.4.1 Model Architecture

We train each variant on the SynthTE-Hard video dataset and evaluate on VTE-Bench-Synth. Starting from VACE as the baseline, we gradually add components until reaching the full SteerVTE.

Table 4: **Ablation of model architecture.** We ablate input, glyph condition, and style encoder, selecting MV2V, dual-level glyph, and VLM for SteerVTE. Best results per sub-group are **bold**.

Architecture		Text Editing					Video Quality						
		Precision↑	Recall↑	F-Score↑	NED↑	Accuracy↑	SSIM↑	PSNR↑	FID↓	FVD↓	Style-Sim↑	$SSIM_{bg}$↑	$PSNR_{bg}$↑
Input	V2V	0.9340	0.8917	0.9123	0.7541	0.1300	0.2901	**10.86**	89.54	20.56	0.4847	0.9537	32.02
	MV2V	**0.9669**	**0.9528**	**0.9597**	**0.8557**	**0.2400**	**0.3063**	10.59	**88.92**	**18.98**	**0.5029**	**0.9617**	**36.04**
+Glyph	Line-Level	0.9788	0.9714	0.9750	0.9380	0.3900	0.3276	11.14	81.54	**17.28**	0.5041	**0.9620**	36.63
	Character-Level	0.9659	0.9553	0.9605	0.8704	0.2700	0.3085	10.89	85.25	18.62	0.5050	0.9582	36.42
	Dual-Level	**0.9891**	**0.9798**	**0.9844**	**0.9561**	**0.5000**	**0.3467**	**11.74**	**80.73**	17.64	**0.5163**	0.9586	**36.77**
+Style	ViT	0.9665	0.9546	0.9605	0.9111	0.4000	0.2572	9.374	112.5	20.85	0.4387	0.9618	36.42
	VLEmbedding	**0.9889**	**0.9829**	**0.9858**	**0.9523**	0.5100	0.3280	11.14	80.45	**16.31**	0.5256	0.9622	36.84
	VLM	0.9836	0.9770	0.9802	0.9466	**0.5300**	**0.3562**	**11.71**	**79.59**	16.90	**0.5417**	**0.9625**	**37.06**

As shown in Tab. 4, the original V2V paradigm achieves only 13% accuracy. Adding a mask improves accuracy by 11% and enhances background preservation. Building on this mask-based V2V paradigm,



Figure 4: **Qualitative comparison.** Prior baselines fail in distinct ways: inaccurate or flickering edits (FLUX-Text), no edit at all (VACE), garbled or style-inconsistent glyphs (Kiwi-Edit, VIVA, UniVideo), or unintended modifications to background text (Seedance 2.0). Our method achieves seamless edits that preserve text accuracy, style consistency, and background integrity.

we incorporate glyph prior information. Character-level glyph information raises accuracy to 27%, line-level information raises it to 39%, and using both granularities further boosts accuracy to 50%. For the style encoder, we compare three variants: ViT, VLEmbedding, and VLM. ViT and VLM are derived from Qwen2.5-VL-3B [4], while VLEmbedding comes from Qwen3-VL-Embedding-2B [34]. ViT yields poor results, possibly due to a representation mismatch with DiT, whereas both VLEmbedding and VLM achieve strong improvements in accuracy and style consistency. We select VLM as the final style encoder due to its higher accuracy.

4.4.2 Training Strategy

To validate our three-stage curriculum learning strategy, we conducted ablation studies on training data and losses. All experimental configurations were kept consistent with the main experiment.

Table 5: **Ablation study on training data and loss.** Best results per setting are **bold**.

Training Data	Text Editing					Video Quality						
	Precision↑	Recall↑	F-Score↑	NED↑	Accuracy↑	SSIM↑	PSNR↑	FID↓	FVD↓	Style-Sim↑	$SSIM_{bg}$↑	$PSNR_{bg}$↑
Video-Only	0.8922	0.8925	0.8923	0.7563	0.3700	0.3371	11.73	84.67	17.88	0.5363	0.9589	36.58
Hybrid Data	**0.9916**	0.9783	0.9849	0.9530	0.5300	0.3467	10.69	83.67	15.49	0.5324	0.9523	35.92
Curriculum Learning	0.9908	**0.9825**	**0.9866**	**0.9624**	**0.6100**	**0.4034**	**12.30**	**65.46**	**14.55**	**0.5936**	**0.9606**	**37.09**

Training Loss		Text Editing					Video Quality						
OCR	SF	Precision↑	Recall↑	F-Score↑	NED↑	Accuracy↑	SSIM↑	PSNR↑	FID↓	FVD↓	Style-Sim↑	$SSIM_{bg}$↑	$PSNR_{bg}$↑
×	×	0.9848	0.9774	0.9810	0.9474	0.5100	0.3481	11.24	79.26	16.44	0.5268	0.9604	35.51
✓	×	0.9928	**0.9902**	**0.9914**	**0.9751**	**0.6500**	0.2881	10.04	108.1	18.03	0.4794	0.9605	36.09
×	✓	**0.9929**	0.9817	0.9872	0.9613	0.5600	0.3756	11.33	69.84	**13.99**	0.5691	**0.9608**	35.71
✓	✓	0.9908	0.9825	0.9866	0.9624	0.6100	**0.4034**	**12.30**	**65.46**	14.55	**0.5936**	0.9606	**37.09**

Training Data. As shown at the top of Tab. 5, the *Video-Only* setup, which uses only video data without pretraining on image data, leads to degraded performance across all metrics, confirming that video foundation models have weak text priors and benefit from training data organized by increasing difficulty. *Hybrid Data*, which mixes all data in a single training stage, improves text accuracy but yields suboptimal results due to conflicts between image and video characteristics. Our three-stage *Curriculum Learning* first uses image data for basic text editing, then video data for temporal consistency, achieving optimal results on nearly all metrics.

Training Loss. We conduct this experiment during the third training stage using flow matching loss as the baseline. As shown at the bottom of Tab. 5, the baseline struggles with both text accuracy and style consistency. Adding a Glyph-Aware OCR loss raises text accuracy by 14 points but degrades style consistency by biasing generation toward simple fonts and colors. Adding a Spatial-Focal Loss instead concentrates supervision on the editing region, improving style consistency with little gain in text accuracy. To combine the best of both worlds, GLAS Loss amalgamates the two losses, striking an optimal balance between text accuracy and style fidelity.

5 Conclusion

This work introduces SteerVTE, the first end-to-end architecture for seamless video text editing. At its core, a frozen DiT backbone is augmented with three precise editing signals: a spatiotemporal mask that localizes the target region, a VLM-based style encoder that captures fine-grained visual attributes, and a dual-granularity glyph module that fuses character- and sentence-level representations. To overcome the inherently weak glyph priors of video foundation models, we design a three-stage curriculum that progresses from elementary glyph learning, through complex real-world text, and finally to full temporal-coherence training. We further propose a GLAS loss that concentrates supervision on text regions, boosting sentence-level fidelity and style consistency. To support training and evaluation, we construct SteerVTE-1M, a million-triplet dataset spanning diverse scenes and text, and VTE-Bench, the first dedicated benchmark for video text editing.

Limitations and Future Work. SteerVTE currently supports only English and is designed for single-line editing. To overcome these limitations, future work will extend it to other languages using larger multilingual datasets and enable multi-line editing through improved foundation models. Beyond font diversity, dynamic subtitles with richer text effects are also promising. While our model achieves over 95% normalized edit distance, accuracy remains paramount since even single-character errors are often unacceptable. Reinforcement learning, such as GRPO, could help close the remaining gap. Model distillation further offers a path toward lightweight real-time subtitle editing.

Limited by the capabilities of video foundation models, long-video text editing remains a limitation of our approach. Future work could address this through more efficient model architectures or algorithms that improve long-context modeling. More importantly, real-world video text editing data remain scarce, especially paired scene text editing videos that cover cases where the text moves with the scene, or where the text is deformed, zoomed, and otherwise transformed. Training on such data is expected to yield better results than the hybrid image-video training data we currently use.

6 Acknowledgments

We would like to thank Yiming Zhao and Siyu Jiao for many insightful discussions. This work is supported by Beijing Municipal Natural Science Foundation (Grant No. L2603008), Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China (JYB2025XDXM113), National Natural Science Foundation of China (92470121, 62402016), National Key R&D Program of China (2024YFA1014003), Zhongguancun Academy (C20250204, C20250602), Beijing Major Science and Technology Project (Z251100008125043, Z251100008425023), and High-performance Computing Platform of Peking University.

Competing Interests. The authors declare no competing interests beyond the affiliations and funding sources disclosed in this paper.

References

[1] Alibaba Cloud. Tongyi wan 2.7 video generation, 2026. https://tongyi.aliyun.com/wan/generate/video/generate?model=wan2.7.

[2] Ruichuan An, Sihan Yang, Renrui Zhang, Ming Lu, Tianyi Jiang, Kai Zeng, Yulin Luo, Jiajun Cao, Hao Liang, Ying Chen, et al. Mc-llava: Multi-concept personalized vision-language model. *arXiv preprint arXiv:2411.11706*, 2024.

[3] Ruichuan An, Kai Zeng, Ming Lu, Sihan Yang, Renrui Zhang, Huitong Ji, Hao Liang, and Wentao Zhang. Concept-as-tree: A controllable synthetic data framework makes stronger personalized vlms. In *European Conference on Computer Vision*, pages 474–491. Springer, 2026.

[4] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

[5] Yuxuan Bian, Zhaoyang Zhang, Xuan Ju, Mingdeng Cao, Liangbin Xie, Ying Shan, and Qiang Xu. Videopainter: Any-length video inpainting and editing with plug-and-play context control. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pages 1–12, 2025.

[6] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18392–18402, 2023.

[7] Haoxing Chen, Zhuoer Xu, Zhangxuan Gu, Yaohui Li, Changhua Meng, Huijia Zhu, Weiqiang Wang, et al. Diffute: Universal text editing diffusion model. *Advances in Neural Information Processing Systems*, 36:63062–63074, 2023.

[8] Jingye Chen, Yupan Huang, Tengchao Lv, Lei Cui, Qifeng Chen, and Furu Wei. Textdiffuser: Diffusion models as text painters. *Advances in Neural Information Processing Systems*, 36:9353–9387, 2023.

[9] Jingye Chen, Yupan Huang, Tengchao Lv, Lei Cui, Qifeng Chen, and Furu Wei. Textdiffuser-2: Unleashing the power of language models for text rendering. In *European Conference on Computer Vision*, pages 386–402. Springer, 2024.

[10] Jiaxin Cheng, Tianjun Xiao, and Tong He. Consistent video-to-video transfer using synthetic dataset. *arXiv preprint arXiv:2311.00213*, 2023.

[11] Xiaoyan Cong, Haotian Yang, Angtian Wang, Yizhi Wang, Yiding Yang, Canyu Zhang, and Chongyang Ma. Viva: Vlm-guided instruction-based video editing with reward optimization. *arXiv preprint arXiv:2512.16906*, 2025.

[12] Yuren Cong, Mengmeng Xu, Christian Simon, Shoufa Chen, Jiawei Ren, Yanping Xie, Juan-Manuel Perez-Rua, Bodo Rosenhahn, Tao Xiang, and Sen He. Flatten: optical flow-guided attention for consistent text-to-video editing. *arXiv preprint arXiv:2310.05922*, 2023.

[13] Wikipedia contributors. Peak signal-to-noise ratio, 2026. https://en.wikipedia.org/w/index.php?title=Peak_signal-to-noise_ratio&oldid=1210897995.

[14] Cheng Cui, Ting Sun, Manhui Lin, Tingquan Gao, Yubo Zhang, Jiaxuan Liu, Xueqing Wang, Zelun Zhang, Changda Zhou, Hongen Liu, et al. Paddleocr 3.0 technical report. *arXiv preprint arXiv:2507.05595*, 2025.

[15] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[16] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

[17] Xiang Fan, Anand Bhattad, and Ranjay Krishna. Videoshop: Localized semantic video editing with noise-extrapolated diffusion inversion. *arXiv preprint arXiv:2403.14617*, 2024.

[18] first20hours. google-10000-english, 2026. `https://github.com/first20hours/google-10000-english`.

[19] Michal Geyer, Omer Bar-Tal, Shai Bagon, and Tali Dekel. Tokenflow: Consistent diffusion features for consistent video editing. *arXiv preprint arXiv:2307.10373*, 2023.

[20] Google. Google fonts, 2026. `https://fonts.google.com/`.

[21] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376, 2006.

[22] Yuchao Gu, Yipin Zhou, Bichen Wu, Licheng Yu, Jia-Wei Liu, Rui Zhao, Jay Zhangjie Wu, David Junhao Zhang, Mike Zheng Shou, and Kevin Tang. Videoswap: Customized video subject swapping with interactive semantic point correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[23] Yoav HaCohen, Nisan Chiprut, Benny Brazowski, Daniel Shalem, Dudu Moshe, Eitan Richardson, Eran Levin, Guy Shiran, Nir Zabari, Ori Gordon, et al. Ltx-video: Realtime video latent diffusion. *arXiv preprint arXiv:2501.00103*, 2024.

[24] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[25] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[26] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

[27] Jiahao Hu, Tianxiong Zhong, Xuebo Wang, Boyuan Jiang, Xingye Tian, Fei Yang, Pengfei Wan, and Di Zhang. Vivid-10m: A dataset and baseline for versatile and interactive video local editing. *arXiv preprint arXiv:2411.15260*, 2024.

[28] 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*, pages 17191–17202, 2025.

[29] Ozgur Kara, Bariscan Kurtkaya, Hidir Yesiltepe, James M Rehg, and Pinar Yanardag. Rave: Randomized noise shuffling for fast and consistent video editing with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

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

[31] Max Ku, Cong Wei, Weiming Ren, Huan Yang, and Wenhu Chen. Anyv2v: A plug-and-play framework for any video-to-video editing tasks. *arXiv preprint arXiv:2403.14468*, 2024.

[32] Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, et al. Flux. 1 kontext: Flow matching for in-context image generation and editing in latent space. *arXiv preprint arXiv:2506.15742*, 2025.

[33] Rui Lan, Yancheng Bai, Xu Duan, Mingxing Li, Lei Sun, and Xiangxiang Chu. Flux-text: A simple and advanced diffusion transformer baseline for scene text editing. *arXiv preprint arXiv:2505.03329*, 2025.

[34] Mingxin Li, Yanzhao Zhang, Dingkun Long, Keqin Chen, Sibo Song, Shuai Bai, Zhibo Yang, Pengjun Xie, An Yang, Dayiheng Liu, et al. Qwen3-vl-embedding and qwen3-vl-reranker: A unified framework for state-of-the-art multimodal retrieval and ranking. *arXiv preprint arXiv:2601.04720*, 2026.

[35] Xirui Li, Chao Ma, Xiaokang Yang, and Ming-Hsuan Yang. Vidtome: Video token merging for zero-shot video editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[36] 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.

[37] Bin Lin, Yunyang Ge, Xinhua Cheng, Zongjian Li, Bin Zhu, Shaodong Wang, Xianyi He, Yang Ye, Shenghai Yuan, Liuhan Chen, et al. Open-sora plan: Open-source large video generation model. *arXiv preprint arXiv:2412.00131*, 2024.

[38] 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.

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

[40] Shaoteng Liu, Tianyu Wang, Jui-Hsien Wang, Qing Liu, Zhifei Zhang, Joon-Young Lee, Yijun Li, Bei Yu, Zhe Lin, Soo Ye Kim, et al. Generative video propagation. *arXiv preprint arXiv:2412.19761*, 2024.

[41] Zeyu Liu, Weicong Liang, Zhanhao Liang, Chong Luo, Ji Li, Gao Huang, and Yuhui Yuan. Glyph-byt5: A customized text encoder for accurate visual text rendering. In *European Conference on Computer Vision*, pages 361–377. Springer, 2024.

[42] Jian Ma, Mingjun Zhao, Chen Chen, Ruichen Wang, Di Niu, Haonan Lu, and Xiaodong Lin. Glyphdraw: Seamlessly rendering text with intricate spatial structures in text-to-image generation. *arXiv preprint arXiv:2303.17870*, 2023.

[43] Lichen Ma, Xiaolong Fu, Gaojing Zhou, Zipeng Guo, Ting Zhu, Yichun Liu, Yu Shi, Jason Li, and Junshi Huang. Um-text: A unified multimodal model for image understanding. *arXiv preprint arXiv:2601.08321*, 2026.

[44] 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. *arXiv preprint arXiv:2304.07193*, 2023.

[45] Hao Ouyang, Qiuyu Wang, Yuxi Xiao, Qingyan Bai, Juntao Zhang, Kecheng Zheng, Xiaowei Zhou, Qifeng Chen, and Yujun Shen. Codef: Content deformation fields for temporally consistent video processing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[46] Wenqi Ouyang, Yi Dong, Lei Yang, Jianlou Si, and Xingang Pan. I2VEdit: First-Frame-Guided Video Editing via Image-to-Video Diffusion Models. *arXiv preprint arXiv:2405.16537*, 2024.

[47] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023.

[48] 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*, 2023.

[49] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[50] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[51] Team Seedance, De Chen, Liyang Chen, Xin Chen, Ying Chen, Zhuo Chen, Zhuowei Chen, Feng Cheng, Tianheng Cheng, Yufeng Cheng, et al. Seedance 2.0: Advancing video generation for world complexity. *arXiv preprint arXiv:2604.14148*, 2026.

[52] Yongdeuk Seo, Hyun-seok Min, and Sungchul Choi. Stellar: Scene text editor for low-resource languages and real-world data. *arXiv preprint arXiv:2511.09977*, 2025.

[53] Wenda Shi, Yiren Song, Dengming Zhang, Jiaming Liu, and Xingxing Zou. Fonts: Text rendering with typography and style controls. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18463–18474, 2025.

[54] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[55] Jeyasri Subramanian, Varnith Chordia, Eugene Bart, Shaobo Fang, Kelly Guan, Raja Bala, et al. Strive: Scene text replacement in videos. *arXiv preprint arXiv:2109.02762*, 2021.

[56] Zhiyu Tan, Hao Yang, Luozheng Qin, Jia Gong, Mengping Yang, and Hao Li. Omni-video: Democratizing unified video understanding and generation. *arXiv preprint arXiv:2507.06119*, 2025.

[57] Yuxiang Tuo, Yifeng Geng, and Liefeng Bo. Anytext2: Visual text generation and editing with customizable attributes. *arXiv preprint arXiv:2411.15245*, 2024.

[58] Yuxiang Tuo, Wangmeng Xiang, Jun-Yan He, Yifeng Geng, and Xuansong Xie. Anytext: Multilingual visual text generation and editing. *arXiv preprint arXiv:2311.03054*, 2023.

[59] Thomas Unterthiner, Sjoerd Van Steenkiste, Karol Kurach, Raphaël Marinier, Marcin Michalski, and Sylvain Gelly. Fvd: A new metric for video generation. 2019.

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

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

[62] 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.

[63] Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*, 2025.

[64] Jay Zhangjie Wu, Yixiao Ge, Xintao Wang, Stan Weixian Lei, Yuchao Gu, Yufei Shi, Wynne Hsu, Ying Shan, Xiaohu Qie, and Mike Zheng Shou. Tune-a-video: One-shot tuning of image diffusion models for text-to-video generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[65] Weijia Wu, Yuanqiang Cai, Debing Zhang, Sibo Wang, Zhuang Li, Jiahong Li, Yejun Tang, and Hong Zhou. A bilingual, openworld video text dataset and end-to-end video text spotter with transformer. *arXiv preprint arXiv:2112.04888*, 2021.

[66] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024.

[67] Kai Zeng, Zhanqian Wu, Kaixin Xiong, Xiaobao Wei, Xiangyu Guo, Zhenxin Zhu, Kalok Ho, Lijun Zhou, Bohan Zeng, Ming Lu, et al. Rethinking driving world model as synthetic data generator for perception tasks. In *International Conference on Learning Representations*, volume 2026, pages 135507–135534, 2026.

[68] Weichao Zeng, Yan Shu, Zhenhang Li, Dongbao Yang, and Yu Zhou. Textctrl: Diffusion-based scene text editing with prior guidance control. *Advances in Neural Information Processing Systems*, 37:138569–138594, 2024.

[69] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3836–3847, 2023.

[70] Zhenghao Zhang, Zuozhuo Dai, Long Qin, and Weizhi Wang. Effived: Efficient video editing via text-instruction diffusion models. *arXiv preprint arXiv:2403.11568*, 2024.

[71] Yiming Zhao, Yuanpeng Gao, Yuxuan Luo, Jiwei Duan, Shisong Lin, Longfei Xiong, and Zhouhui Lian. Utdesign: A unified framework for stylized text editing and generation in graphic design images. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*, pages 1–11, 2025.

[72] Yuanzhi Zhu, Jiawei Liu, Feiyu Gao, Wenyu Liu, Xinggang Wang, Peng Wang, Fei Huang, Cong Yao, and Zhibo Yang. Visual text generation in the wild. In *European Conference on Computer Vision*, pages 89–106. Springer, 2024.

A Outlines

The supplementary material presents the following sections to strengthen the main manuscript:

- Section B.1 presents the complete construction pipeline of our SteerVTE-1M dataset and Section B.2 includes a detailed derivation of the GLAS Loss.
- Section C.1 introduces the composition of our VTE-Bench evaluation suite and the specifics of each evaluation metric, while Section C.2 elaborates on training and inference details.
- Section D presents extra quantitative comparisons with existing image editing methods.
- Section E provides additional visualizations of our ablation studies.
- Section F presents extra qualitative comparisons with existing video editing methods.

B Additional Method Details

B.1 Dataset Construction

This appendix provides a comprehensive description of the SteerVTE-1M dataset construction, complementing the summary in Sec. 3.3. The dataset comprises two complementary components: SynthTE, a stylized synthetic dataset with diverse scenes, fonts, and effects, and RealTE, a real-world image-based text editing dataset. Both come with fine-grained annotations, including text strings, bounding boxes, and editing region masks. Fig. 5 illustrates the construction pipeline.

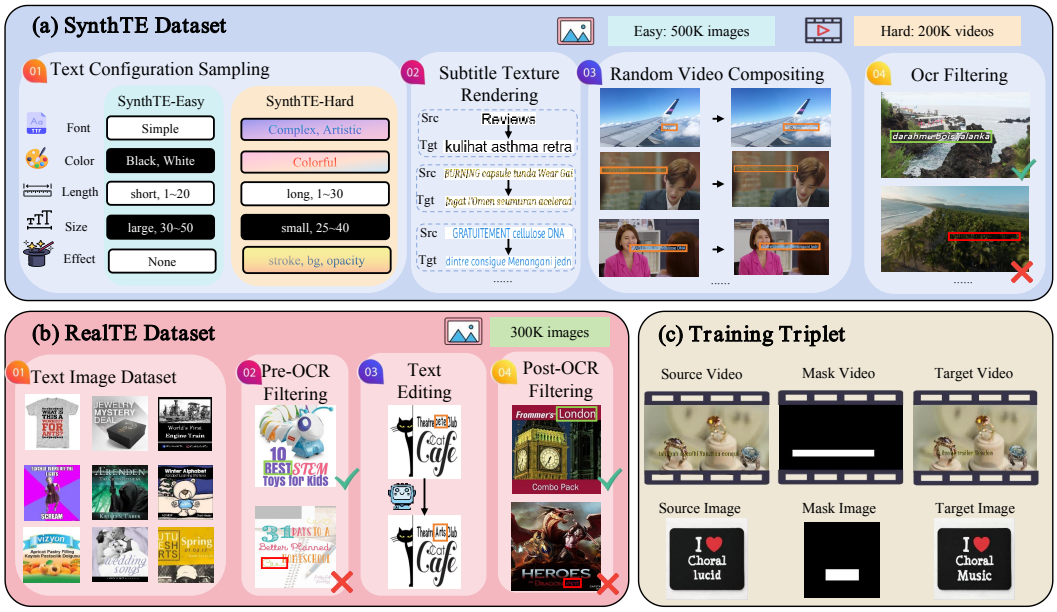

Figure 5: **Construction process of the visual text editing training dataset.** (a) Using the same procedure with two different text configurations, we synthesize the SynthTE-Easy and SynthTE-Hard. (b) We construct the image text editing RealTE dataset from real-world OCR data. (c) Example samples from the video and image datasets.

For SynthTE, we collect 1,933 TrueType fonts from Google Fonts [20], an English text corpus [18] of 65 letters and symbols, and background videos from opensora-plan [37]. Given the weak text rendering of video foundation models, we split SynthTE into easy and hard versions to support different training stages. The easy version uses 894 simple black-and-white fonts, restricts text length to under 20 characters, and sets the font size from 30 to 50. The hard version uses all available fonts, including artistic and textured styles, randomly samples colors, and adds transparency, background boxes, and strokes, with text length up to 30 and font size from 25 to 40.

As illustrated in Fig. 5(a), the pipeline randomly samples text attributes and paired text to render subtitle images, selects background videos, and generates bounding boxes covering 8% to 15% of the frame for pasting. It inserts one to four subtitle pairs, selects one as the training region, and retains only samples that achieve 100% OCR recognition accuracy. This process produces 500,000 images for SynthTE-Easy and 200,000 videos for SynthTE-Hard.

For RealTE, we note that SynthTE alone cannot address real-world scenarios, yet real video text editing data is scarce. Since no existing model preserves style and temporal consistency when editing videos, paired data is unavailable. We therefore curated a real image-based text editing dataset from design images (Fig. 5(b)), leveraging OCR-annotated images and image text editors. Our experiments show that editing capabilities learned from images generalize to videos. From AnyWord-3M [58], we used an OCR model [14] to retain only fully recognizable editing regions, yielding 500k samples. We then applied a text editing model [68] and OCR filtering to ensure clear text before and after editing. To preserve stylistic consistency, we also employ a Text Style Similarity metric [52] and discard samples with scores below 0.4. This pipeline produces the RealTE Dataset with 300k image samples.

We present additional examples from our diverse SteerVTE-1M dataset, including samples from all three training stages (SynthTE-Easy, RealTE, and SynthTE-Hard), as illustrated in Figures 13–15. For a better visual experience, please visit our anonymous website `https://steervte-paper.github.io/`, which includes video demonstrations of qualitative comparisons and dataset samples.

B.2 GLAS Loss Details

We provide the detailed formulation of the OCR supervision term in GLAS. Given the model prediction at timestep t, we first recover the clean latent estimate $\hat{\mathbf{z}}_0$ and decode it with the VAE decoder $\mathcal{D}$ to obtain the reconstructed video

$$\hat{\mathbf{V}} = \mathcal{D}(\hat{\mathbf{z}}_0). \tag{6}$$

We then uniformly sample aligned frame triplets from the ground-truth video, reconstructed video, and mask video:

$$(\mathbf{I}_{\mathrm{gt}}, \mathbf{I}_{\mathrm{pred}}, \mathbf{I}_{\mathrm{m}}) = \mathcal{S}\Big(\mathbf{V}_{\mathrm{gt}},\, \hat{\mathbf{V}},\, \mathbf{M}\Big)\,. \tag{7}$$

We use a pre-trained OCR recognizer $\mathcal{O}$ [14] and distinguish its outputs for different purposes. For an input text patch $\mathbf{x}$, the recognizer produces

$$\big\{\mathcal{O}_{\mathrm{neck}}(\mathbf{x}),\ \mathcal{O}_{\mathrm{feat}}(\mathbf{x}),\ \mathcal{O}_{\mathrm{logits}}(\mathbf{x})\big\}, \tag{8}$$

where $\mathcal{O}_{\mathrm{neck}}$ denotes the neck feature used in the main text for cross-attention injection, $\mathcal{O}_{\mathrm{feat}}$ denotes the multi-scale feature maps used in $\mathcal{L}_{\text{text}}$, and $\mathcal{O}_{\mathrm{logits}}$ denotes the character logits used in $\mathcal{L}_{\text{CTC}}$.

Given the sampled mask $\mathbf{I}_{\mathrm{m}}$, we crop the text-bearing regions from $\mathbf{I}_{\mathrm{gt}}$ and $\mathbf{I}_{\mathrm{pred}}$, and normalize them with the OCR preprocessing pipeline:

$$\tilde{\mathbf{P}}_{\mathrm{gt}} = \mathcal{N}(\mathrm{Crop}(\mathbf{I}_{\mathrm{gt}}, \Omega(\mathbf{I}_{\mathrm{m}})))\,, \quad \tilde{\mathbf{P}}_{\mathrm{pred}} = \mathcal{N}(\mathrm{Crop}(\mathbf{I}_{\mathrm{pred}}, \Omega(\mathbf{I}_{\mathrm{m}})))\,, \tag{9}$$

where $\Omega(\mathbf{I}_{\mathrm{m}})$ denotes the text region indicated by the mask.

Text perceptual loss. To explicitly constrain glyph topology, we match OCR feature maps between the predicted and target text patches. Let

$$\mathcal{O}_{\mathrm{feat}}(\tilde{\mathbf{P}}_{\mathrm{gt}}) = \{\mathbf{F}_{\mathrm{gt}}^{(l)}\}_{l=1}^{L}, \qquad \mathcal{O}_{\mathrm{feat}}(\tilde{\mathbf{P}}_{\mathrm{pred}}) = \{\mathbf{F}_{\mathrm{pred}}^{(l)}\}_{l=1}^{L}, \tag{10}$$

where $\mathbf{F}_{\mathrm{gt}}^{(l)}, \mathbf{F}_{\mathrm{pred}}^{(l)} \in \mathbb{R}^{H_l \times W_l \times C_l}$. We define

$$\mathcal{L}_{\text{text}} = \sum_{l=1}^{L} \frac{1}{H_l W_l C_l} \left\| \mathbf{F}_{\mathrm{pred}}^{(l)} - \mathbf{F}_{\mathrm{gt}}^{(l)} \right\|_2^2. \tag{11}$$

CTC recognition loss. To further ensure semantic correctness, we apply a CTC loss on the OCR character logits:

$$\mathbf{Y}_{\mathrm{pred}} = \mathcal{O}_{\mathrm{logits}}(\tilde{\mathbf{P}}_{\mathrm{pred}}) \in \mathbb{R}^{T \times |\mathcal{V}|}, \tag{12}$$

$$\mathcal{L}_{\text{CTC}} = \mathrm{CTC}(\mathbf{Y}_{\mathrm{pred}}, y) = -\log p(y \mid \mathbf{Y}_{\mathrm{pred}}), \tag{13}$$

where y is the target transcription.

The OCR supervision term is thus

$$\mathcal{L}_{\text{OCR}} = \mathcal{L}_{\text{text}} + \mathcal{L}_{\text{CTC}}. \tag{14}$$

It is worth noting that, to ensure the effectiveness of the OCR loss computation and training stability, we only apply the OCR loss at low-noise timesteps ($t < 0.3$). At these timesteps, the predicted latent variable $\hat{\mathbf{z}}_0$ is more likely to decode into a clear image, preventing blurry and artifact-ridden frames from causing the OCR model to produce random and meaningless gradients.

For completeness, the latent-space spatial-focal loss in the main text is

$$\mathcal{L}_{\text{SF}} = \|\mathbf{W} \odot (v_\theta(\cdot) - (\mathbf{z}_1 - \mathbf{z}_0))\|_2^2, \tag{15}$$

where $\mathbf{W}$ is the normalized focal map derived from the text mask $\mathbf{M}$.

Finally, the overall GLAS objective is

$$\mathcal{L}_{\text{GLAS}} = \gamma \mathcal{L}_{\text{SF}} + \lambda \mathcal{L}_{\text{OCR}}, \tag{16}$$

where we use $\gamma = 3.0$ and $\lambda = 0.1$ in all experiments.

C Additional Experiment Details

C.1 Benchmark and Metrics

Construction details. For VTE-Bench-Synth, we synthesize 50 test samples from SynthTE-Easy and 50 from SynthTE-Hard following the procedure described in Sec. 3.3. Each sample contains a source video, a random editing mask, an editing prompt, and the ground-truth edited video. For VTE-Bench-Real, we build 100 test samples from real-world video text detection datasets [65] without ground-truth edited videos. For VTE-Bench-SceneText, we build 200 test samples from STRIVE [55] and the web, covering static, dynamic, and deformable scene text. This subset represents the most challenging setting for video text editing.

The training videos used in this paper are sourced from this [37]. The original authors permit academic use, and the selected videos are released under the MIT License. The test videos are sourced from this [65]. The original authors permit academic use, and the selected videos are released under the Creative Commons Attribution 4.0 International License.

Metric computation details. Except for FVD [59], all metrics are computed by uniformly sampling five frames from each video and averaging the results across frames. For text editing accuracy, we use PP-OCRv4 [14] for text detection and recognition. Our SteerVTE model employs PP-OCRv3 as the glyph encoder and for OCR-loss computation due to its more accessible architecture, while the stronger PP-OCRv4 is adopted solely for evaluation to ensure fairness. Based on the OCR predictions and ground-truth texts, we report precision, recall, F-score, normalized edit distance (NED), and sentence accuracy. A higher NED indicates better performance:

$$\text{NED} = 1 - \frac{\text{ED}(\text{pred}, \text{gt})}{\max(\text{len}(\text{pred}), \text{len}(\text{gt}))},$$

where ED is the Levenshtein edit distance between the OCR output pred and the ground-truth text gt. Sentence accuracy is defined as 1 if the full OCR string exactly matches the target, and 0 otherwise. Sentence accuracy is stricter than NED, since any single character error causes the sentence to fail.

For video quality assessment, we use SSIM [61], PSNR [13], and FID [24] to measure visual similarity between the ground-truth and edited frames; Text Style Similarity [52] (denoted as Style-Sim) to evaluate style consistency; and FVD to measure temporal coherence. Since the edited regions occupy only 10%–15% of each video, these metrics are computed on the cropped edited regions. To further assess background preservation, we compute SSIM and PSNR on frames with the edited regions masked out, denoted as SSIM_{bg} and PSNR_{bg}, respectively. As VTE-Bench-Real does not provide ground-truth edited videos, we additionally conduct a user study involving twenty-five participants from varied backgrounds (non-researchers, industry, and students). Each participant assigns a binary score to each result: 1 if the edit contains accurate text, preserves style consistency, and leaves the background unchanged, and 0 otherwise. The final Quality Rate is computed as the average of these scores. Moreover, since SSIM_{bg} and PSNR_{bg} do not require edited-video ground truth, we compute them directly between the edited videos and the original input videos.

C.2 Implementation Details

More implementation details. Stage 1 uses SynthTE Easy with 500k image samples at a resolution of 832×480 for 40k optimization steps. Stage 2 uses RealTE with 300k image samples at a resolution of 512×512 for 20k steps. Stage 3 uses SynthTE Hard with 200k video samples at a resolution of 832×480 with 49 frames per video for 23k steps. All training experiments are conducted on 16 NVIDIA H100 GPUs with a learning rate of 1×10^{-5} using the AdamW optimizer and a batch size of 1. The resolution of the line-level glyph image matches the original image resolution, while the resolution of the character-level glyph image is 80×80.

Model Input. For our method, the model takes as input the source video, an editing mask video, a text prompt, and both line-level and character-level glyph images rendered from the target text. The style reference image is obtained by extracting the masked text region from the first frame of the source video. The prompt for the style encoder is: "Describe in detail the typography, color, style, text material, and rendering effects of the text regions {source text} in this image." The prompt for the T5 encoder is: "A clear and high-quality video with smooth motion, featuring the stylized text: {target text}." For other mask-based video editing baselines, the input consists of the source video, the editing mask video, a line-level glyph image, and a text prompt. To explicitly indicate the editable region, the prompt is set to: "Change the text from {source text} to {target text}, while keeping the text style consistent." For instruction-guided models, reference-guided models, and Seedance 2.0, which do not support editing masks, the input includes the source video, a line-level glyph image, and the same text prompt used for mask-based video editing models.

Inference speed. We report the average inference time of each model on a single 49-frame video at a resolution of 832×480. As shown in Table 6, Kiwi-Edit is the fastest method, while our method achieves competitive efficiency comparable to other 14B-class baselines.

Note. All open-source methods, including ours, use 20 inference steps, since we empirically observe negligible differences compared with 50 steps. For open-source baselines, we use their official implementations. For Seedance 2.0, we use the official 480p API.

Table 6: Average inference time on a 49-frame video with resolution 832×480.

Method	Time (s)
VACE [28] (14B)	103
VideoPainter [5] (12B + 5B)	81
VIVA [11] (8B + 13B)	161
UniVideo [62] (7B + 13B)	277
Kiwi-Edit [38] (3B + 5B)	8
Seedance 2.0 [51]	126
Ours (3B + 14B)	108

D Additional Quantitative Evaluation

We additionally compare with STRIVE [55], TextCtrl [68], AnyText2 [57], and FLUX2 [32] on VTE-Bench-Synth and VTE-Bench-Real. As shown in Tab. 7 and Tab. 8, image text editors struggle with sentence-level correctness and temporal stability in videos. In contrast, SteerVTE directly edits videos, achieving higher text accuracy, better style consistency, and more stable results. STRIVE and TextCtrl edit only foreground text regions and reuse the original backgrounds, which explains their higher background scores.

Table 7: **Quantitative comparison on VTE-Bench-Synth.** Comparison with additional text editing methods. **Bold** is the best. STRIVE* is our reproduced implementation.

Method	Text Editing					Video Quality						
	Precision↑	Recall↑	F-Score↑	NED↑	Accuracy↑	SSIM↑	PSNR↑	FID↓	FVD↓	Style-Sim↑	$SSIM_{bg}$↑	$PSNR_{bg}$↑
STRIVE*	0.8574	0.8135	0.8271	0.6913	0.0400	0.2895	12.47	107.1	16.49	0.5199	**0.9773**	**39.85**
TextCtrl	0.8176	0.5832	0.6559	0.3700	0.0100	0.3430	**12.69**	98.10	127.8	0.5336	**0.9773**	**39.85**
AnyText2	0.7393	0.6516	0.6803	0.3999	0.0300	0.1832	10.97	163.4	884.2	0.4056	0.8197	23.42
FLUX2	0.7707	0.6736	0.7088	0.5706	0.0800	0.2333	11.37	108.7	372.2	0.4538	0.7031	22.22
SteerVTE (Ours)	**0.9908**	**0.9825**	**0.9866**	**0.9624**	**0.6100**	**0.4034**	12.30	**65.46**	**14.55**	**0.5936**	0.9606	37.09

E Additional Ablation Study

As shown in Fig. 6, our ablation study evaluates both model architecture and training loss. In Fig. 6(a), the V2V baseline fails to accurately localize the editing region and often modifies the wrong text. MV2V improves region targeting but still suffers from glyph errors. Introducing dual-granularity

Table 8: **Quantitative comparison on VTE-Bench-Real.** Comparison with additional text editing methods. SteerVTE achieves the best text editing performance. **Bold** is the best.

Method	Text Editing					Video Quality	
	Precision↑	Recall↑	F-Score↑	NED↑	Accuracy↑	$SSIM_{bg}$↑	$PSNR_{bg}$↑
STRIVE*	0.6900	0.6754	0.6752	0.6005	0.1300	**0.9809**	**40.35**
TextCtrl	0.8476	0.7855	0.8088	0.6702	0.2700	**0.9809**	**40.35**
AnyText2	0.7058	0.7061	0.6968	0.5300	0.0800	0.8473	22.11
FLUX2	0.6538	0.5904	0.6082	0.4652	0.0900	0.6860	18.24
SteerVTE (Ours)	**0.9785**	**0.9769**	**0.9776**	**0.9579**	**0.7700**	0.9648	35.55

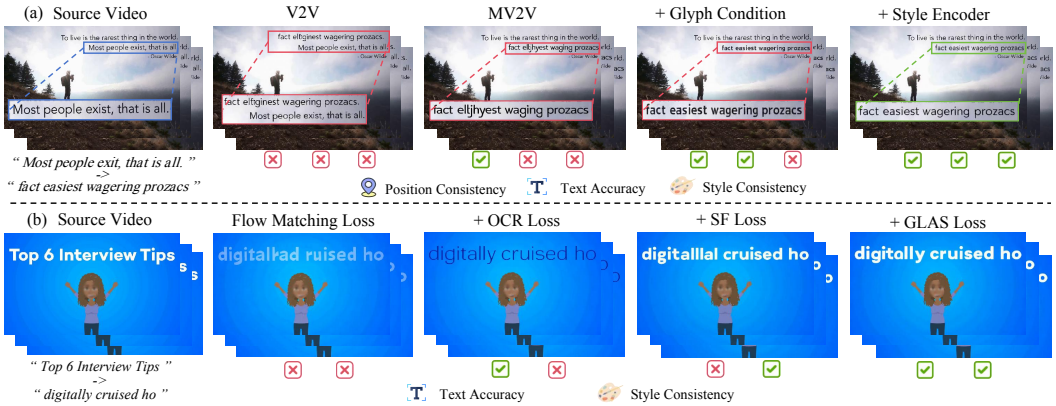

Figure 6: **Visualization of ablation experiments.** (a) Model architecture: MV2V enables accurate region localization; glyph conditioning improves text accuracy; and the style encoder ensures style consistency. (b) Training loss: Using either the OCR loss or the SF loss alone yields suboptimal results. The GLAS loss achieves both text accuracy and style consistency.

glyph conditioning corrects the text content, although style inconsistency remains. Finally, equipping the model with VLM enables precise text editing while preserving style consistency. In Fig. 6(b), the flow matching loss alone leads to suboptimal text accuracy and style quality. Adding the OCR loss improves text accuracy but degrades style consistency. The mask reweighting loss enhances style consistency yet still leaves minor character errors. Our GLAS loss combines the strengths of both and achieves the best trade-off between text accuracy and style consistency.

F Additional Qualitative Comparison

We provide additional qualitative comparisons between our method and other video editing methods on all synthetic, real-world and scenetext scenarios in Figures 8–12. For a better visual experience, please visit our project page `https://zengkaiya.github.io/SteerVTE/`.

G Broader Impacts

SteerVTE has the potential to advance research on fine-grained and controllable video editing by enabling precise text modification in videos while preserving stylistic consistency, visual realism, and temporal coherence. Such capabilities may benefit applications including video localization, media production, accessibility, and creative content editing. At the same time, realistic video text editing techniques could be misused to manipulate on-screen information in deceptive ways, potentially contributing to misinformation or fraud. In addition, performance may vary across languages, writing systems, fonts, and visual contexts due to biases in training data. To mitigate these concerns, we emphasize that research on video text editing should be accompanied by responsible deployment practices, including usage restrictions, transparency mechanisms, and continued study of detection and attribution methods for edited media. By releasing our findings responsibly, we aim to support progress in controllable video editing while promoting ethical consideration of its societal impact.

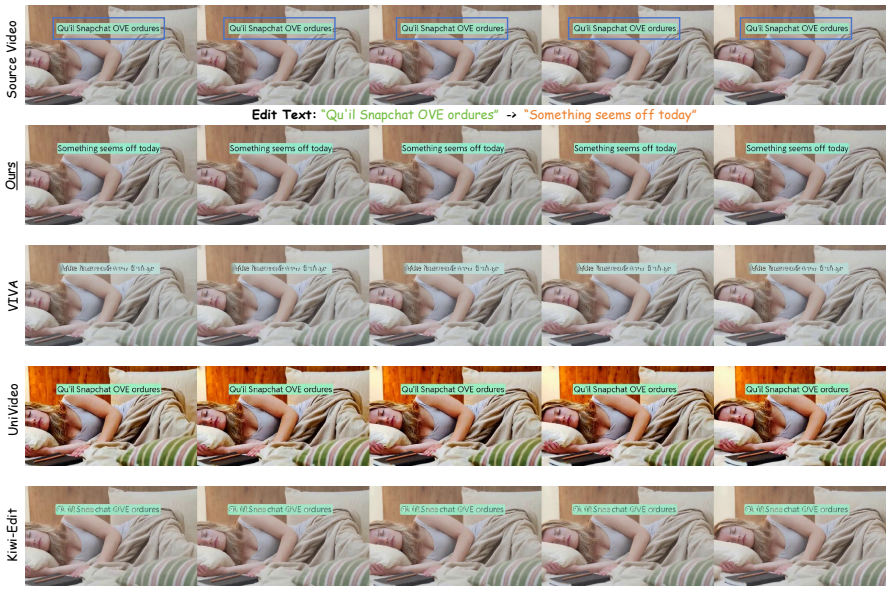

Figure 7: Visual comparison on synthetic scenarios.

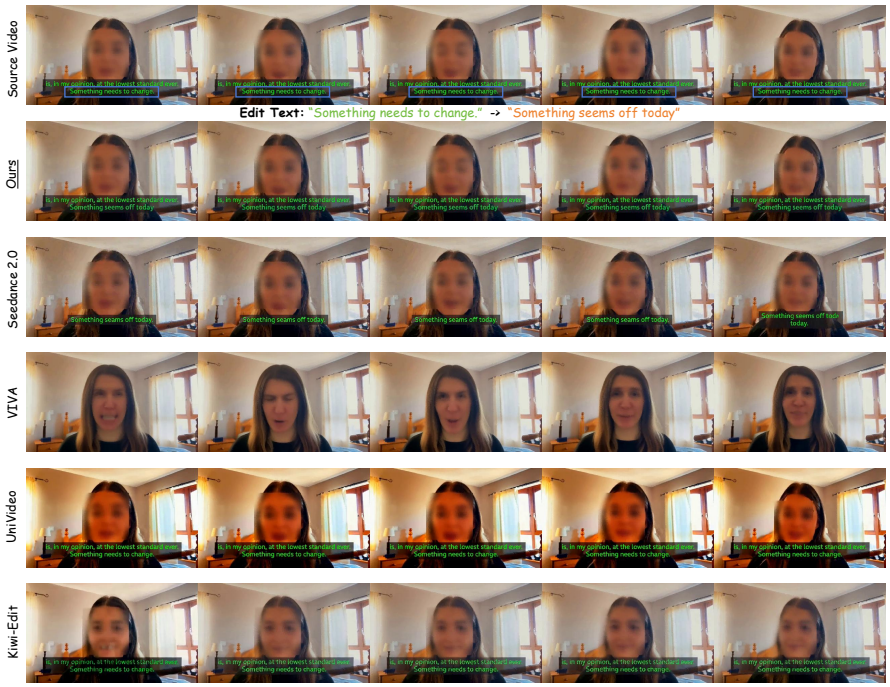

Figure 8: Visual comparison on real-world scenarios.



Figure 9: Visual comparison on scenetext scenarios.



Figure 10: Visual comparison on scenetext scenarios.



Figure 11: Visual comparison on scenetext scenarios.



Figure 12: Visual comparison on scenetext scenarios.



Figure 13: Examples from our stage-1 image training dataset: SynthTE-Easy.

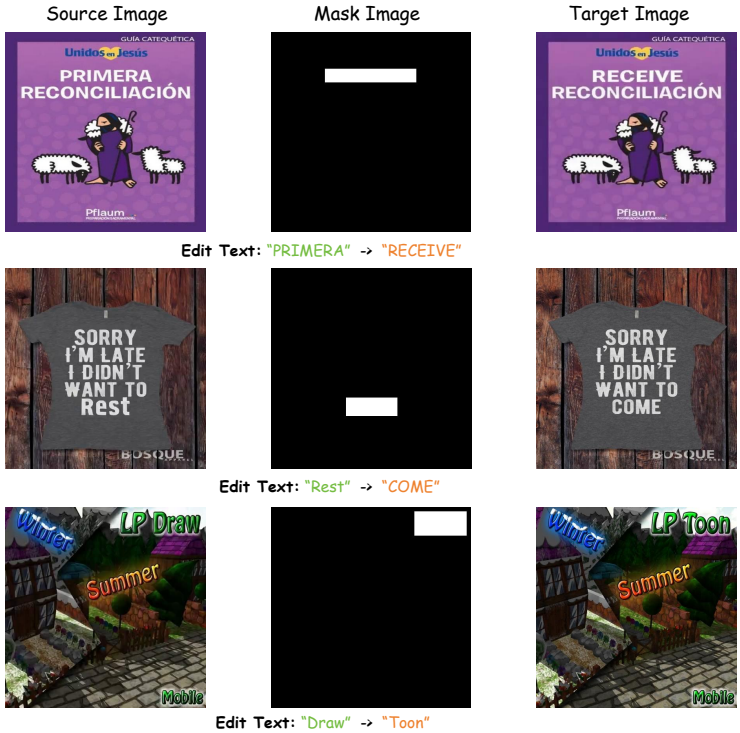

Figure 14: Examples from our stage-2 image training dataset: RealTE.

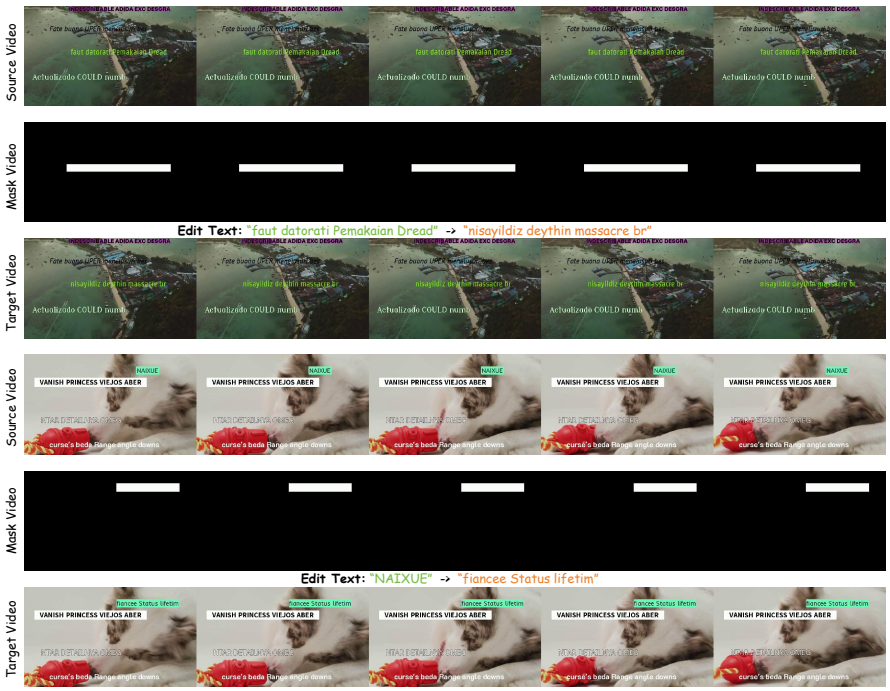

Figure 15: Examples from our stage-3 video training dataset: SynthTE-Hard.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction accurately summarize the proposed method, its main contributions, and the scope supported by our experiments. Supporting evidence is provided in Fig. 1, Sec. 4 and Appx. C.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations are discussed within the conclusion section.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [No]

Justification: The paper does not present theoretical results requiring formal assumptions or proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The paper describes the model architecture, training pipeline, datasets, evaluation metrics, and implementation details needed to reproduce the main results. Additional experimental details are provided in the appendix.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: We do not provide open access to the code and data at submission time. We include experimental details in the paper and appendix to facilitate reproducibility, and will consider releasing resources after the review process.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies the datasets, training and inference settings, optimization details, and evaluation protocol in the main paper and appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We do not report error bars or formal statistical significance tests, as the experiments were conducted using fixed evaluation protocols and single trained models due to computational cost.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The appendix reports the hardware used for training and evaluation, including GPU type, memory, and approximate training time.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and believe that this research complies with its principles.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper discusses both potential benefits, such as efficient video text editing and localization, and possible risks, including misuse for deceptive or misleading media manipulation.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [No]

    Justification: We do not release a public system or model in this submission, and thus do not describe a deployment-time safeguard mechanism. We acknowledge the potential misuse risks of video editing technology in the broader impacts discussion.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper cites the datasets, pretrained models, and codebases used in our experiments, and their licenses or terms of use are documented in the supplemental material.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The paper introduces a new dataset, SteerVTE-1M, and describes its automatic construction pipeline, scale, and intended use in the main paper and appendix.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: The paper describes the user study protocol, including the evaluation task and participant judgment criteria. The study was conducted with volunteer participants from our lab.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [No]

Justification: The paper does not involve human subjects research and therefore does not require IRB approval.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core methodology does not rely on LLMs as an important, original, or non-standard component. Any language models used serve as standard text encoders rather than a novel methodological contribution.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.