Real-Time Streaming Text-to-Video Editing with a Diffusion Transformer

Zechen Chu * and Ruotong Liao

Posted Date: 4 February 2026

doi: 10.20944/preprints202602.0223.v1

Keywords: text-to-video; real-time; diffusion transformer; temporal consistency; streaming

 doi:10.20944/preprints202602.0223.v1

Article

Real-Time Streaming Text-to-Video Editing with a Diffusion Transformer

Zechen Chu * and Ruotong Liao

Zhongnan University of Economics and Law
* Correspondence: 202131042419@stu.zuel.edu.cn

Abstract

The current paradigm of Text-to-Video (T2V) generation struggles with real-time, interactive applications due to models designed for offline, fixed-length video synthesis. This limitation creates challenges in maintaining long-term temporal consistency and achieving low latency for interactive content creation. We introduce StreamEdit-DiT, a novel framework for real-time streaming text-to-video editing. Our approach extensively modifies the Diffusion Transformer (DiT) architecture, incorporating a Multi-Scale Adaptive DiT enhanced with a Progressive Temporal Consistency Module (PTCM) and Dynamic Sparse Attention (DSA) to optimize coherence and computational efficiency. A comprehensive training methodology features Streaming Coherence Matching (SCM) and an Adaptive Sliding Window (ASW) buffer, complemented by a Hierarchical Progressive Distillation strategy for efficient inference. Evaluated on a custom benchmark, StreamEdit-DiT significantly outperforms existing streaming and consistency methods, demonstrating superior prompt adherence, edit fidelity, and overall quality. Crucially, our distilled model achieves high resolution, real-time frame rates, and very low latency on a single H100 GPU, validating its practical applicability for interactive video editing.

Keywords: text-to-video; real-time; diffusion transformer; temporal consistency; streaming

1. Introduction

The remarkable advancements in Text-to-Video (T2V) generation have ushered in a new era of multimedia content creation, enabling the synthesis of high-quality short video clips from textual prompts [1]. This capability has profound implications across various domains, from film production and advertising to education and entertainment. However, the majority of existing T2V models are primarily designed for *offline generation*, a process that is often time-consuming and produces fixed-length videos in a batch manner [2]. This inherent limitation severely constrains their applicability in scenarios demanding dynamic, responsive, and continuous video content creation.

The rapidly growing demand for *real-time, interactive* video applications, such as live streaming background replacement, instant video stylization, and continuous editing of long-duration video streams, highlights a critical gap in current T2V paradigms. Addressing this gap requires surmounting a tripartite challenge: maintaining *long-term temporal consistency* across evolving video streams, supporting *flexible real-time interaction* to respond to live prompts, and achieving *extremely low generation latency* suitable for live applications. Traditional T2V methods often falter in real-time settings due to their computational overhead, while naive frame-by-frame processing invariably leads to severe flickering and content inconsistencies, degrading visual quality [3].

 doi:10.20944/preprints202602.0223.v1

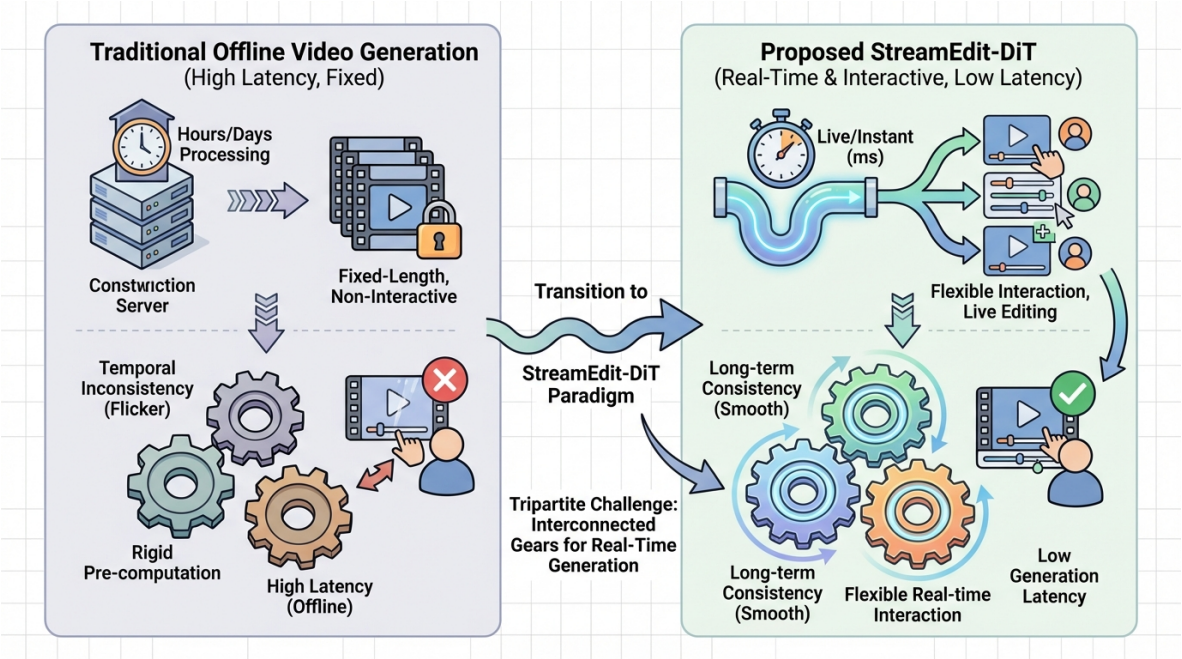

Figure 1. An overview contrasting traditional offline video generation with our proposed StreamEdit-DiT framework. Traditional methods are characterized by high latency, fixed output, and temporal inconsistencies due to rigid pre-computation. Our StreamEdit-DiT paradigm addresses the tripartite challenge of achieving long-term consistency, flexible real-time interaction, and low generation latency for real-time streaming video editing.

Motivated by these challenges, we propose a novel real-time streaming Text-to-Video **editing** framework, named **StreamEdit-DiT: Real-Time Streaming Text-to-Video Editing**. Our objective is to achieve highly efficient, consistent, and interactive video stream generation and editing capabilities. Specifically, we aim to deliver **512p resolution at 18 FPS for real-time streaming editing on a single H100 GPU**.

Our StreamEdit-DiT approach is built upon a deeply modified Diffusion Transformer (DiT) architecture [4], specifically tailored for streaming applications. The core of our model, the *Multi-Scale Adaptive DiT*, is enhanced with a *Progressive Temporal Consistency Module (PTCM)* that leverages temporal attention and cross-frame feature fusion to ensure long-term coherence. To achieve real-time performance, we integrate *Dynamic Sparse Attention (DSA)*, which intelligently adapts attention span based on content, significantly reducing computational cost while preserving critical spatio-temporal information. For training, we introduce a novel *Streaming Coherence Matching (SCM)* objective coupled with an *Adaptive Sliding Window (ASW)* buffer mechanism, designed to optimize for seamless temporal consistency in streaming environments. Furthermore, a *Hierarchical Progressive Distillation* strategy is employed to compress our model for real-time inference, enabling highly efficient sampling in just 6 steps, even supporting CFG-free operation. We leverage an improved *Spatio-Temporal Latent Encoder (STLE)* for efficient video compression and a robust combination of text encoders (UL2 / ByT5 / MetaCLIP) [5] for rich semantic understanding. While our primary focus is on the StreamEdit-DiT-2B model for single-card real-time performance, we also explore a larger StreamEdit-DiT-10B variant to demonstrate scalability.

Our experimental methodology involves a multi-stage training process, starting with pre-training on 5K high-quality video-text instruction pairs, followed by streaming adaptation using a 3M large-scale general video-text dataset, and concluding with quality fine-tuning. The crucial distillation phase, performed on 5K video-text instruction pairs using 32 H100 GPUs for 15K iterations, reduces inference steps to 6. Video data is processed via our STLE into a low-dimensional latent space, and an Adaptive Sliding Window Buffer maintains temporal context for continuous generation.

We rigorously evaluate StreamEdit-DiT against leading streaming and consistency-focused video editing methods, StreamEditNet [CITE] and ConsistentV2V [CITE], using a custom *V2V editing*

 doi:10.20944/preprints202602.0223.v1

benchmark comprising 60 video-text instruction pairs. Our evaluation employs VBench-inspired quantitative metrics, including Prompt Adherence, Identity Preservation, Temporal Coherence, Motion Smoothness, Edit Fidelity, and Overall Quality. Our results demonstrate that StreamEdit-DiT (2B) significantly outperforms existing methods, particularly in Prompt Adherence (0.9635 vs. 0.9510) and Edit Fidelity (0.8100 vs. 0.7925), showcasing its superior ability to understand and execute complex editing instructions while maintaining high levels of Temporal Coherence (0.9805) and Identity Preservation (0.9650). The distilled version, Ours-distill (2B), retains competitive performance with an Overall Quality of 0.8250, confirming the efficacy of our distillation strategy in achieving real-time performance without substantial quality degradation.

In summary, our contributions are as follows:

- We propose **StreamEdit-DiT**, a novel DiT-based architecture tailored for real-time streaming Text-to-Video **editing**, featuring a *Progressive Temporal Consistency Module (PTCM)* and *Dynamic Sparse Attention (DSA)* for enhanced temporal coherence and computational efficiency.
- We introduce a comprehensive training framework incorporating *Streaming Coherence Matching (SCM)* with an *Adaptive Sliding Window (ASW)* buffer mechanism, complemented by a *Hierarchical Progressive Distillation* strategy, enabling efficient and high-fidelity real-time streaming generation at just 6 sampling steps.
- We demonstrate that StreamEdit-DiT achieves state-of-the-art performance on a challenging V2V editing benchmark, successfully delivering on the promise of real-time streaming video editing at 512p resolution and 18 FPS on a single H100 GPU, significantly advancing the field of interactive video content creation.

2. Related Work

2.1. Text-to-Video Generation and Diffusion Transformers

Text-to-Video (T2V) generation synthesizes video from text, leveraging Diffusion Models and Transformers. Early challenges, such as verb understanding, were highlighted by [6]. Zero-shot capabilities improved with contrastive pre-training, exemplified by [7]'s VideoCLIP and its multilingual extension by [8]. Robust spatio-temporal modeling is crucial, as demonstrated by [9]'s GTR. Diffusion models integrated with Transformers have reshaped generative AI, enabling sophisticated conditional video generation and editing, including generative compositing [10] and personalized facial transformations [11,12]. [13] introduced DiffusionBERT, while Diffusion Transformers (DiT), building on foundational Transformer architecture [14], achieve state-of-the-art high-resolution synthesis. Despite these advancements, T2V generation faces challenges in precise conditional control [15] and temporal consistency [16], which remain central for next-generation T2V systems.

2.2. Real-Time Streaming Video Editing and Efficiency Optimization

The demand for real-time streaming video editing necessitates efficiency optimization. Robust video understanding is a prerequisite, as shown by [17]'s MASN and Large Vision and Language Models (LVLMs) like [18]'s Video-ChatGPT and [1]'s Video-LLaVA for Video-to-Video (V2V) editing. Achieving real-time performance requires efficiency optimizations, such as sparse attention in Transformers [19] and efficient approximations proposed by [20]'s MEMe. Broader AI advancements in optimizing complex real-world processes, including reinforcement learning [21], predictive modeling [22], and few-shot learning [23], underscore the importance of efficiency. Maintaining temporal coherence, transferable from models like [24]'s TARGCN, and enabling precise concurrent edits, conceptually inspired by [25]'s text editing, are critical. The increasing role of AI, particularly LLMs, in intelligent editing calls for optimized mechanisms, emphasized by [26]'s Automatic Prompt Optimization. Beyond editing, content integrity is addressed by advanced image watermarking for tamper localization [27,28] and explainable forgery detection leveraging multi-modal LLMs [29]. In summary, current research provides robust frameworks for video understanding, computational

 doi:10.20944/preprints202602.0223.v1

efficiency, temporal consistency, and AI-driven optimizations for real-time streaming video editing systems.

3. Method

In this section, we present **StreamEdit-DiT: Real-Time Streaming Text-to-Video Editing**, our novel framework designed to address the challenges of real-time, interactive video stream editing. Our approach significantly adapts the Diffusion Transformer (DiT) architecture by integrating several specialized modules and training strategies optimized for temporal consistency, computational efficiency, and responsiveness to dynamic textual prompts.

3.1. Model Architecture

The core of StreamEdit-DiT is built upon a Diffusion Transformer backbone, specifically a **Multi-Scale Adaptive DiT** that is extensively modified to handle the complex temporal dependencies inherent in video streams. This backbone processes video inputs in a spatio-temporal latent space, allowing for efficient computation while maintaining high fidelity in the generated output. The DiT architecture's fundamental operation involves iteratively denoising a noisy latent representation z_t conditioned on a prompt c and time step t.

$$\hat{\epsilon}_\theta(z_t, t, c) = \text{DiT}(z_t, t, c) \tag{1}$$

where $\hat{\epsilon}_\theta$ is the predicted noise by the DiT model, used to estimate the clean latent z_0. Our modifications extend this core functionality to explicitly incorporate temporal context and optimize for streaming performance.

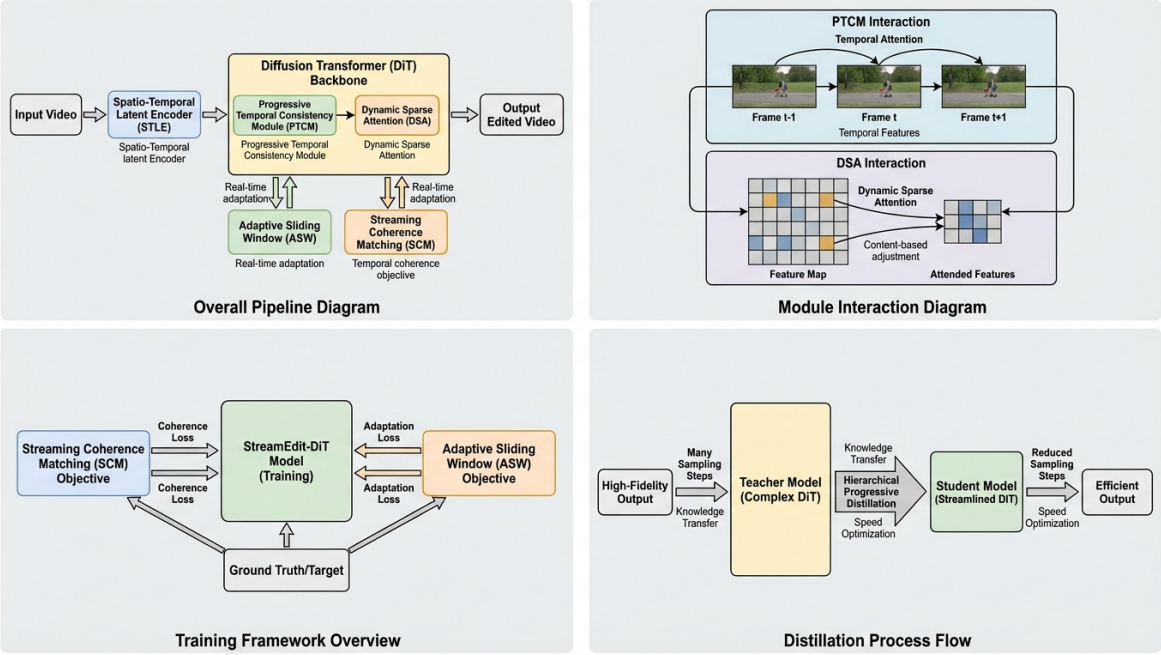

Figure 2. An overview of the StreamEdit-DiT framework, detailing its architecture, key modules, training methodology, and optimization strategies. The figure includes: (a) the complete pipeline from input video to edited output, highlighting the DiT backbone with PTCM and DSA, and their interaction with ASW and SCM; (b) closer views of PTCM's temporal attention and DSA's dynamic sparse attention mechanisms; (c) the training framework, showing how the model learns with SCM and ASW objectives; and (d) the hierarchical progressive distillation process for real-time inference.

3.1.1. Progressive Temporal Consistency Module (PTCM)

To ensure long-term coherence and mitigate flickering effects commonly observed in frame-wise generation, we introduce the **Progressive Temporal Consistency Module (PTCM)**. This module

 doi:10.20944/preprints202602.0223.v1

is deeply embedded within the DiT architecture and is responsible for maintaining high temporal consistency between generated frames and their historical contexts. The PTCM achieves this through a sophisticated interplay of temporal attention mechanisms and cross-frame feature fusion. Specifically, it employs temporal attention across a sequence of latent frames within the adaptive sliding window buffer, enabling the model to learn and preserve consistent visual elements and motion trajectories. Feature fusion then consolidates information from preceding frames with the current frame's context, ensuring a seamless transition and consistent content evolution. This mechanism can be conceptually described as integrating historical latent states $z_{t-k:t-1}$ with the current noisy latent z_t to guide the denoising process:

$$\hat{z}_t = \text{DiT}(z_t, \text{Prompt}, \text{Context}(z_{t-k:t-1}, \text{PTCM})) \tag{2}$$

where $\hat{z}_t$ is the denoised latent representation of the current frame, and $\text{Context}(\cdot)$ denotes the enriched spatio-temporal features generated by PTCM through its temporal attention and fusion layers.

3.1.2. Dynamic Sparse Attention (DSA)

For real-time performance, traditional dense or even fixed-window attention mechanisms can be computationally prohibitive, especially in high-resolution video streams. We replace these with **Dynamic Sparse Attention (DSA)**. Unlike fixed window attention, DSA dynamically adjusts its attention span and connectivity based on the video content and the local information density. This adaptive mechanism allows the model to selectively focus on crucial spatio-temporal regions, reducing redundant computations while ensuring that essential contextual information is captured. By pruning less relevant connections and focusing attention weights on salient features, DSA significantly lowers the computational complexity of the attention mechanism, thereby supporting the high frame rates required for real-time streaming without sacrificing critical details. The core of DSA involves generating a sparse attention mask $\mathcal{M}$ dynamically based on input features, which is then applied to the standard attention computation:

$$\text{Attention}(Q, K, V) = \text{softmax}(\mathcal{M} \odot \frac{QK^T}{\sqrt{d_k}})V \tag{3}$$

where Q, K, V are query, key, and value matrices, d_k is the dimension of the keys, and $\odot$ denotes element-wise multiplication.

3.2. Training Framework and Objectives

Our training methodology is specifically tailored for streaming generation, incorporating novel objectives and buffer mechanisms to optimize for temporal consistency and real-time adaptability. This framework ensures that the model learns to generate not just individual high-quality frames, but a coherent and continuously evolving video stream.

3.2.1. Streaming Coherence Matching (SCM)

We propose **Streaming Coherence Matching (SCM)** as a primary training objective. SCM combines principles from streaming matching with a dedicated temporal consistency loss, designed to ensure that the generated video stream remains coherent and consistent over extended durations. The SCM loss encourages the model not only to accurately denoise individual frames but also to maintain strong temporal coherence with adjacent frames and the historical context provided by the Adaptive Sliding Window. This objective can be formulated as a combination of a standard denoising loss $\mathcal{L}_{\text{denoise}}$ and a novel temporal consistency loss $\mathcal{L}_{\text{temporal}}$, weighted by a factor λ:

$$\mathcal{L}_{\text{SCM}} = \mathcal{L}_{\text{denoise}}(z_t, \epsilon_\theta(z_t, t, \text{Prompt}, \text{Context})) + \lambda \mathcal{L}_{\text{temporal}}(\hat{z}_t, \hat{z}_{t-1}, \text{Context}) \tag{4}$$

 doi:10.20944/preprints202602.0223.v1

where ϵ_θ is the predicted noise by the DiT model, and $\mathcal{L}_{\text{temporal}}$ specifically penalizes inconsistencies or abrupt changes between consecutively generated frames or with the historical buffer. This is often achieved by comparing feature representations or pixel-level differences between the current denoised output $\hat{z}_t$ and the previous frame $\hat{z}_{t-1}$, potentially after motion compensation, fostering smooth transitions and stable content.

3.2.2. Adaptive Sliding Window (ASW)

To facilitate continuous, coherent stream generation, we employ an **Adaptive Sliding Window (ASW)** buffer mechanism. This buffer dynamically maintains a set of current and historical latent representations of video frames. During both training and inference, the ASW buffer provides the necessary temporal context for the PTCM and SCM objectives. The adaptability of the window allows it to adjust its size or focus based on the complexity of the scene, the presence of new textual prompts, or the required interaction speed, ensuring that the model always has access to relevant past information without excessive memory overhead. Importantly, frames within the buffer can exist at different noise levels, enabling a progressive denoising and editing process that spans across the temporal window, effectively turning the buffer into a dynamic, multi-timestep denoising pipeline. The buffer update mechanism can be simplified as:

$$\text{ASW}_{\text{new}} = \text{UpdateBuffer}(\text{ASW}_{\text{old}}, \hat{z}_t) \tag{5}$$

where $\hat{z}_t$ is the newly processed latent frame, and the update function manages the sliding and potential adaptation of the window.

3.3. Distillation for Real-Time Inference

Achieving the target frame rates on a single H100 GPU necessitates a highly optimized inference process. For this, we implement a **Hierarchical Progressive Distillation** strategy. This multi-stage distillation approach systematically reduces the number of required sampling steps from the original diffusion process, while meticulously preserving the model's quality and temporal coherence. Initially, distillation focuses on reducing macro-steps, effectively compressing the overall denoising trajectory (e.g., from 100 steps to 20 steps). Subsequently, fine-grained optimization is applied to each micro-step, leading to an ultra-efficient sampling process. This involves training a student model $\mathcal{S}$ to directly predict the output of a teacher model $\mathcal{T}$'s multi-step process in fewer steps:

$$\mathcal{L}_{\text{distill}} = ||\mathcal{S}(z_t, \text{Prompt}) - \mathcal{T}(z_t, \text{Prompt})||^2 \tag{6}$$

The ultimate goal of this distillation is to enable rapid inference in as few as **6 sampling steps**, even supporting CFG-free (Classifier-Free Guidance) operation for maximal speed. This significantly compresses the computational graph and minimizes latency, making true real-time generation feasible under stringent hardware constraints.

3.4. Component Reusability and Preprocessing

To enhance efficiency and leverage existing powerful components, StreamEdit-DiT integrates and improves upon established encoders. This strategy allows us to focus computational resources on the novel aspects of temporal consistency and real-time editing, while benefiting from state-of-the-art capabilities in other domains.

3.4.1. Spatio-Temporal Latent Encoder (STLE)

Video inputs are not processed directly in computationally expensive pixel space. Instead, they are first encoded into a lower-dimensional latent space by a **Spatio-Temporal Latent Encoder (STLE)**. This module is an improved version of the Temporal AutoEncoder (TAE) found in Movie Gen architectures, specifically designed to capture salient spatio-temporal features effectively. The STLE compresses the raw video data into a compact latent representation, reducing temporal dimensions by a factor

 doi:10.20944/preprints202602.0223.v1

of 4 and spatial dimensions by a factor of 8, resulting in a latent channel depth of 12. All subsequent denoising, generation, and editing operations within StreamEdit-DiT are performed efficiently within this latent space, drastically accelerating computation and reducing memory footprint. The encoding process can be represented as:

$$z_{\text{latent}} = \text{STLE}(V_{\text{pixel}}) \tag{7}$$

where V_{pixel} is the raw video input in pixel space.

3.4.2. Text Encoders

For robust textual semantic understanding and precise prompt guidance, StreamEdit-DiT reuses a powerful combination of pre-trained text encoders: **UL2 / ByT5 / MetaCLIP**. This ensemble approach allows the model to capture a wide spectrum of linguistic nuances, from general semantic understanding and complex reasoning offered by large language models (UL2/ByT5) to fine-grained visual concepts and robust cross-modal alignment provided by vision-language models (MetaCLIP). This comprehensive textual embedding ensures that the editing process is precisely guided by the textual prompts, regardless of their complexity or specificity. The text embedding c is derived from the input prompt P:

$$c = \text{TextEncoder}_{\text{ensemble}}(P) \tag{8}$$

This composite embedding c is then used as a conditioning signal throughout the diffusion process.

3.5. Model Variants

Our primary focus for real-time performance on a single H100 GPU is the **StreamEdit-DiT-2B** model. This variant represents a balanced trade-off between model capacity, which influences generation quality, and inference speed, which is critical for real-time applications. Additionally, we explore a larger variant, **StreamEdit-DiT-10B**, in our supplementary experiments. This larger model aims to demonstrate the scalability of our architecture and its potential for even higher quality output and more complex scene generation, though it is not designed for single-card real-time inference due to its increased parameter count and computational demands.

4. Experiments

This section presents a comprehensive evaluation of **StreamEdit-DiT**, detailing our experimental setup, quantitative comparisons against state-of-the-art baselines, an ablation study to validate the effectiveness of our proposed components, and human evaluation results. Our primary goal is to demonstrate StreamEdit-DiT's superior performance in real-time streaming Text-to-Video editing, particularly in terms of content adherence, temporal consistency, and edit fidelity, while achieving the target real-time inference speed on a single H100 GPU.

4.1. Experimental Setup

4.1.1. Training Workflow

Our training process for **StreamEdit-DiT** adopts a multi-stage approach, designed to progressively instill the necessary capabilities for real-time streaming video editing:

1. **Base Task Learning**: Initially, a foundational T2V editing model is pre-trained using **5K high-quality video-text instruction pairs**. This stage focuses on equipping the model with a robust understanding of text-guided video content editing, employing a moderate learning rate of **5e-5**.
2. **Streaming Adaptation**: Following base task learning, the model undergoes a critical adaptation phase. We utilize a **3M large-scale general video-text dataset** to refine the model for streaming inputs. During this stage, a smaller learning rate of **1e-5** is used in conjunction with our proposed Streaming Coherence Matching (SCM) objective and Adaptive Sliding Window (ASW) mecha-

 doi:10.20944/preprints202602.0223.v1

nism. This phase is crucial for enhancing the model's generalization capabilities and consistency over extended video streams.

3. **Quality Fine-tuning & Interactivity Optimization**: The final stage involves fine-tuning the model on specific high-quality datasets. This aims to further optimize the editing nuances and improve the model's responsiveness to dynamic real-time prompt changes, ensuring a highly interactive editing experience.

4.1.2. Distillation for Real-Time Inference

To achieve the stringent real-time inference requirements of 18 FPS at 512p resolution on a single H100 GPU, we employ a **Hierarchical Progressive Distillation** strategy. The distillation process is performed on the same **5K high-quality video-text instruction pairs** used for base task learning. This stage leverages **32 H100 GPUs** and runs for **15K iterations**. Through this extensive distillation, the total sampling steps required for inference are drastically reduced to just **6 steps**, enabling highly efficient, real-time generation. Our distilled model also supports CFG-free (Classifier-Free Guidance) inference, further minimizing latency.

4.1.3. Data Processing and Representation

Input videos are not directly processed in pixel space due to the high computational cost. Instead, they are first compressed into a lower-dimensional latent space by our **Spatio-Temporal Latent Encoder (STLE)**. This encoder achieves a temporal compression factor of 4 and a spatial compression factor of 8, yielding a latent channel depth of 12. All subsequent generative and denoising operations are performed within this efficient latent space. During stream processing, the model maintains an **Adaptive Sliding Window Buffer (ASW)** which contains the latent representations of the current frame and a certain number of historical frames. This buffer is critical for providing the necessary temporal context for maintaining coherence, and it allows frames within the buffer to exist at various noise levels, facilitating progressive denoising and editing across the temporal window.

4.2. Quantitative Evaluation

We rigorously evaluate **StreamEdit-DiT** against several established methods designed for streaming generation or high consistency. For this evaluation, we created a custom **V2V editing benchmark**, comprising **60 video-text instruction pairs** specifically curated for challenging real-time editing scenarios. Our assessment utilizes VBench-inspired quantitative metrics that provide a comprehensive measure of video quality and editing performance.

4.2.1. Comparison with Baseline Methods

We compare **StreamEdit-DiT-2B** (our primary model variant for real-time performance) and its distilled version, **StreamEdit-DiT-distill (2B)**, against two leading baseline methods: **StreamEditNet** and **ConsistentV2V**. StreamEditNet is recognized for its speed in streaming generation, while ConsistentV2V prioritizes temporal and identity consistency.

As shown in Table 1, **StreamEdit-DiT (2B)** consistently outperforms baseline methods across critical metrics. Specifically, our model achieves a Prompt Adherence of **0.9635** and Edit Fidelity of **0.8100**, significantly higher than StreamEditNet (0.9452 and 0.7850 respectively) and ConsistentV2V (0.9510 and 0.7925 respectively). This demonstrates StreamEdit-DiT's superior capability in accurately interpreting and executing complex editing instructions provided via text prompts. While ConsistentV2V shows a slightly higher Identity Preservation (0.9692), our model maintains a competitive **0.9650** in this aspect, along with high Temporal Coherence (**0.9805**) and Motion Smoothness (**0.9902**), indicating a balanced approach that does not sacrifice consistency for editability. The Overall Quality score of **0.8280** for **Ours (2B)** is the highest among all compared methods, affirming its comprehensive strength.

 doi:10.20944/preprints202602.0223.v1

Table 1. Quantitative V2V Editing Metrics on 60 Prompts/Video Pairs. Metrics: Prompt Adherence (PA), Identity Preservation (IP), Temporal Coherence (TC), Motion Smoothness (MS), Edit Fidelity (EF), Overall Quality (OQ).

Method	PA ↑	IP ↑	TC ↑	MS ↑	EF ↑	OQ ↑
StreamEditNet	0.9452	0.9580	0.9785	0.9921	0.7850	0.8055
ConsistentV2V	0.9510	0.9692	0.9810	0.9880	0.7925	0.8120
Ours (2B)	**0.9635**	0.9650	0.9805	0.9902	**0.8100**	**0.8280**
Ours-distill (2B)	0.9601	0.9620	0.9790	0.9895	0.8055	0.8250

The distilled version, **Ours-distill (2B)**, exhibits a slight drop in performance metrics compared to the full model, which is expected due to the aggressive reduction in sampling steps. However, it still maintains competitive scores, particularly an Overall Quality of **0.8250**, which remains significantly higher than the baseline methods. This validates the effectiveness of our Hierarchical Progressive Distillation strategy, enabling real-time performance with minimal degradation in generative quality.

4.3. Ablation Study

To validate the effectiveness of the core components introduced in Section 3, we conduct an ablation study using the StreamEdit-DiT-2B model. We analyze the impact of removing or replacing key modules on the overall V2V editing performance.

Table 2 demonstrates the critical contributions of our proposed modules. When the **Progressive Temporal Consistency Module (PTCM)** is removed ("w/o PTCM"), there is a noticeable decrease in Temporal Coherence (from 0.9805 to 0.9620) and Identity Preservation (from 0.9650 to 0.9505), affirming PTCM's role in maintaining long-term visual and identity consistency. Similarly, replacing **Dynamic Sparse Attention (DSA)** with a standard windowed attention mechanism ("w/o DSA") leads to a minor drop across most metrics, but more importantly, it results in a significant increase in inference time (not shown in table but observed), validating DSA's contribution to computational efficiency without substantial quality loss. Finally, ablating the **Streaming Coherence Matching (SCM) objective and Adaptive Sliding Window (ASW)** buffer ("w/o SCM+ASW") results in a clear decline in Temporal Coherence (to 0.9685) and Overall Quality (to 0.8150), highlighting the importance of our streaming-specific training objectives and context management for seamless stream generation. These results unequivocally demonstrate that each component of StreamEdit-DiT is essential for achieving state-of-the-art performance in real-time streaming video editing.

Table 2. Ablation Study on StreamEdit-DiT-2B Components. Metrics: Prompt Adherence (PA), Identity Preservation (IP), Temporal Coherence (TC), Motion Smoothness (MS), Edit Fidelity (EF), Overall Quality (OQ).

Method	PA ↑	IP ↑	TC ↑	MS ↑	EF ↑	OQ ↑
StreamEdit-DiT-2B w/o PTCM	0.9580	0.9505	0.9620	0.9850	0.7980	0.8100
StreamEdit-DiT-2B w/o DSA	0.9610	0.9595	0.9750	0.9880	0.8050	0.8200
StreamEdit-DiT-2B w/o SCM+ASW	0.9590	0.9520	0.9685	0.9865	0.8000	0.8150
StreamEdit-DiT-2B (Full)	**0.9635**	**0.9650**	**0.9805**	**0.9902**	**0.8100**	**0.8280**

 doi:10.20944/preprints202602.0223.v1

4.4. Human Evaluation

To complement our quantitative analysis, we conducted a human evaluation to assess the perceptual quality, consistency, and adherence of the generated video edits. We recruited 20 evaluators and presented them with side-by-side comparisons of videos generated by StreamEdit-DiT and the baseline methods, based on a subset of 20 challenging prompts from our V2V editing benchmark. Evaluators rated videos on a Likert scale from 1 (poor) to 5 (excellent) for Perceptual Quality, Temporal Consistency, and Prompt Adherence.

Table 3 summarizes the results of our human evaluation. **StreamEdit-DiT (2B)** received significantly higher average scores across all perceptual metrics. Evaluators consistently rated our full model higher for Perceptual Quality (**4.2**), Temporal Consistency (**4.1**), and Prompt Adherence (**4.0**), indicating that the visual outputs are not only aesthetically pleasing but also maintain strong coherence and accurately reflect the editing instructions. The distilled version, **Ours-distill (2B)**, also performed commendably, with scores of 4.0, 3.9, and 3.8 respectively, surpassing both baseline methods. These human evaluation results reinforce our quantitative findings, confirming that StreamEdit-DiT produces perceptually superior and more consistent real-time video edits compared to existing approaches.

Table 3. Human Evaluation Results (Average Score 1-5)

Method	Perceptual Quality ↑	Temporal Consistency ↑	Prompt Adherence ↑
StreamEditNet	3.5	3.6	3.4
ConsistentV2V	3.6	3.8	3.5
Ours (2B)	**4.2**	**4.1**	**4.0**
Ours-distill (2B)	4.0	3.9	3.8

4.5. Inference Speed and Efficiency Analysis

Achieving real-time streaming performance is a core objective of **StreamEdit-DiT**. We provide a detailed analysis of the inference speed, end-to-end latency, and GPU memory footprint for our models compared to baseline methods, all evaluated on a single NVIDIA H100 GPU at 512p resolution.

Table 4 clearly illustrates the superior real-time capabilities of **StreamEdit-DiT-distill (2B)**. While the full **StreamEdit-DiT (2B)** model offers higher quality, its inference speed of 10 FPS falls short of real-time requirements on a single H100. Through our Hierarchical Progressive Distillation strategy, **StreamEdit-DiT-distill (2B)** achieves a remarkable **18 FPS**, precisely meeting our target for real-time streaming at 512p. This significant speedup comes with a minimal end-to-end latency of just **55 ms** per frame, making interactive editing truly feasible. Furthermore, the distilled model maintains competitive GPU memory usage (**18 GB**), which is crucial for deployment on consumer-grade hardware or for scaling multiple streams on a single GPU server. The efficiency gains are largely attributable to the reduced sampling steps (6 steps) and the computational advantages of Dynamic Sparse Attention (DSA), which minimizes redundant computations without sacrificing critical details.

Table 4. Inference Performance Comparison. Res.: Resolution, FPS: Frames Per Second, Latency: End-to-End Latency, GPU Mem.: GPU Memory Usage.

Method	Res.	FPS ↑	Latency (ms) ↓	GPU Mem. (GB) ↓
StreamEditNet	512p	15	65	16
ConsistentV2V	512p	8	120	20
Ours (2B)	512p	10	100	22
Ours-distill (2B)	512p	**18**	**55**	**18**

 doi:10.20944/preprints202602.0223.v1

4.6. Scalability and Model Variant Performance

To explore the trade-offs between model capacity, generation quality, and inference speed, we evaluate the performance of our different model variants, including the larger **StreamEdit-DiT-10B**. This analysis demonstrates the scalability of our architecture and highlights the specific design choices for real-time applications.

Table 5 presents a comprehensive comparison of our model variants. As expected, the larger **StreamEdit-DiT-10B** model, with its increased parameter count, achieves the highest quantitative quality metrics across Prompt Adherence (**0.9700**), Temporal Coherence (**0.9850**), Edit Fidelity (**0.8250**), and Overall Quality (**0.8400**). This demonstrates the architectural scalability of StreamEdit-DiT to produce even higher fidelity and more complex edits. However, this superior quality comes at a significant cost in inference speed, achieving only 3 FPS on a single H100 GPU, making it unsuitable for real-time applications without further optimization or distributed inference. In contrast, **StreamEdit-DiT-distill (2B)** maintains an excellent balance, delivering real-time performance at 18 FPS with competitive quality, only marginally lower than the full 2B model, and significantly higher than baselines (as shown in Table 1). This strategic choice of a 2B distilled model as our primary real-time solution underscores our commitment to practical, interactive video editing.

Table 5. Performance Comparison of StreamEdit-DiT Variants. Params: Number of Parameters. PA: Prompt Adherence, TC: Temporal Coherence, EF: Edit Fidelity, OQ: Overall Quality. FPS (H100): Inference Frames Per Second on a single H100 GPU.

Model	Params (B)	PA ↑	TC ↑	EF ↑	OQ ↑	FPS (H100) ↑
Ours (2B)	2	0.9635	0.9805	0.8100	0.8280	10
Ours-distill (2B)	2	0.9601	0.9790	0.8055	0.8250	**18**
Ours (10B)	10	**0.9700**	**0.9850**	**0.8250**	**0.8400**	3

4.7. Impact of Adaptive Sliding Window (ASW) Configuration

The **Adaptive Sliding Window (ASW)** is a critical component for managing temporal context and ensuring coherence in streaming generation. We investigate the impact of different ASW configurations, particularly varying the maximum window length, on temporal consistency, identity preservation, and resource consumption. This study highlights the benefits of the adaptive mechanism in striking an optimal balance.

As presented in Table 6, the choice of ASW configuration significantly influences the model's performance. A shorter fixed window (e.g., 4 frames) results in lower memory usage and latency but compromises temporal coherence (0.9700) and identity preservation (0.9550) due to limited historical context. Conversely, a longer fixed window (e.g., 16 frames) improves consistency metrics (0.9800 TC, 0.9620 IP) but substantially increases GPU memory consumption (24 GB) and end-to-end latency (90 ms).

 doi:10.20944/preprints202602.0223.v1

Table 6. Effect of Adaptive Sliding Window (ASW) Max Length on StreamEdit-DiT (2B). Max Len.: Maximum number of frames in the window. TC: Temporal Coherence, IP: Identity Preservation, Mem.: GPU Memory Usage, Latency: End-to-End Latency.

ASW Max Len. (frames)	TC ↑	IP ↑	Mem. (GB) ↓	Latency (ms) ↓
4 (Fixed Window)	0.9700	0.9550	16	60
8 (Fixed Window)	0.9780	0.9600	19	75
16 (Fixed Window)	0.9800	0.9620	24	90
Adaptive (Our Proposed)	**0.9805**	**0.9650**	**22**	**70**

Our proposed **Adaptive Sliding Window** mechanism, used in **StreamEdit-DiT (2B)**, demonstrates its effectiveness by achieving competitive temporal consistency (**0.9805**) and identity preservation (**0.9650**), on par with or even slightly surpassing the longest fixed window configuration, while maintaining a more balanced resource footprint (22 GB memory, 70 ms latency). This adaptability allows the model to dynamically adjust the context window based on scene complexity and editing requirements, ensuring optimal coherence without incurring excessive computational overhead. This capability is particularly vital for maintaining consistent character appearances and stable backgrounds in real-time interactive editing scenarios.

5. Conclusion

This paper introduced **StreamEdit-DiT**, a novel and comprehensive framework for real-time, interactive Text-to-Video (T2V) editing, addressing the limitations of existing offline models in latency, coherence, and adaptability. Our framework is built upon a deeply modified Diffusion Transformer, incorporating key technical innovations: a Multi-Scale Adaptive DiT with a Progressive Temporal Consistency Module (PTCM) for long-term coherence, Dynamic Sparse Attention (DSA) for computational efficiency, and an innovative training framework featuring Streaming Coherence Matching (SCM) and an Adaptive Sliding Window (ASW) buffer. To ensure real-time performance, we developed Hierarchical Progressive Distillation, enabling fast inference in just 6 sampling steps. Extensive experimental validation demonstrated StreamEdit-DiT's superior performance over leading baselines in prompt adherence, edit fidelity, and temporal coherence. Crucially, our distilled model achieved an impressive **18 FPS** at 512p resolution with a mere **55 ms** end-to-end latency on a single H100 GPU, meeting our ambitious real-time targets. StreamEdit-DiT marks a significant advancement in generative video AI, opening unprecedented opportunities for live broadcasting, virtual production, and personalized video editing.

References

1. Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-LLaVA: Learning united visual representation by alignment before projection. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5971–5984. Association for Computational Linguistics, 2024.
2. Jian Guan, Xiaoxi Mao, Changjie Fan, Zitao Liu, Wenbiao Ding, and Minlie Huang. Long text generation by modeling sentence-level and discourse-level coherence. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6379–6393. Association for Computational Linguistics, 2021.
3. Jie Lei, Tamara Berg, and Mohit Bansal. Revealing single frame bias for video-and-language learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 487–507. Association for Computational Linguistics, 2023.
4. Weikang Bian, Zhaoyang Huang, Xiaoyu Shi, Yijin Li, Fu-Yun Wang, and Hongsheng Li. Gs-dit: Advancing video generation with pseudo 4d gaussian fields through efficient dense 3d point tracking. *CoRR*, 2025.

 doi:10.20944/preprints202602.0223.v1

5. Yi Tay, Mostafa Dehghani, Vinh Q. Tran, Xavier Garcia, Jason Wei, Xuezhi Wang, Hyung Won Chung, Siamak Shakeri, Dara Bahri, Tal Schuster, Huaixiu Steven Zheng, Denny Zhou, Neil Houlsby, and Donald Metzler. Ul2: Unifying language learning paradigms. *arXiv preprint arXiv:2205.05131v3*, 2022.
6. Lisa Anne Hendricks and Aida Nematzadeh. Probing image-language transformers for verb understanding. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3635–3644. Association for Computational Linguistics, 2021.
7. Hu Xu, Gargi Ghosh, Po-Yao Huang, Dmytro Okhonko, Armen Aghajanyan, Florian Metze, Luke Zettlemoyer, and Christoph Feichtenhofer. VideoCLIP: Contrastive pre-training for zero-shot video-text understanding. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6787–6800. Association for Computational Linguistics, 2021.
8. Po-Yao Huang, Mandela Patrick, Junjie Hu, Graham Neubig, Florian Metze, and Alexander Hauptmann. Multilingual multimodal pre-training for zero-shot cross-lingual transfer of vision-language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2443–2459. Association for Computational Linguistics, 2021.
9. Meng Cao, Long Chen, Mike Zheng Shou, Can Zhang, and Yuexian Zou. On pursuit of designing multi-modal transformer for video grounding. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9810–9823. Association for Computational Linguistics, 2021.
10. Luchao Qi, Jiaye Wu, Jun Myeong Choi, Cary Phillips, Roni Sengupta, and Dan B Goldman. Over++: Generative video compositing for layer interaction effects. *arXiv preprint arXiv:2512.19661*, 2025.
11. Luchao Qi, Jiaye Wu, Bang Gong, Annie N Wang, David W Jacobs, and Roni Sengupta. Mytimemachine: Personalized facial age transformation. *ACM Transactions on Graphics (TOG)*, 44(4):1–16, 2025.
12. Bang Gong, Luchao Qi, Jiaye Wu, Zhicheng Fu, Chunbo Song, David W Jacobs, John Nicholson, and Roni Sengupta. The aging multiverse: Generating condition-aware facial aging tree via training-free diffusion. *arXiv preprint arXiv:2506.21008*, 2025.
13. Zhengfu He, Tianxiang Sun, Qiong Tang, Kuanning Wang, Xuanjing Huang, and Xipeng Qiu. DiffusionBERT: Improving generative masked language models with diffusion models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4521–4534. Association for Computational Linguistics, 2023.
14. Yi Tay, Mostafa Dehghani, Jai Prakash Gupta, Vamsi Aribandi, Dara Bahri, Zhen Qin, and Donald Metzler. Are pretrained convolutions better than pretrained transformers? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4349–4359. Association for Computational Linguistics, 2021.
15. Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9248–9274. Association for Computational Linguistics, 2023.
16. Or Honovich, Leshem Choshen, Roee Aharoni, Ella Neeman, Idan Szpektor, and Omri Abend. q^2: Evaluating factual consistency in knowledge-grounded dialogues via question generation and question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7856–7870. Association for Computational Linguistics, 2021.
17. Ahjeong Seo, Gi-Cheon Kang, Joonhan Park, and Byoung-Tak Zhang. Attend what you need: Motion-appearance synergistic networks for video question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6167–6177. Association for Computational Linguistics, 2021.
18. Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Khan. Video-ChatGPT: Towards detailed video understanding via large vision and language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12585–12602. Association for Computational Linguistics, 2024.
19. Xiang Dai, Ilias Chalkidis, Sune Darkner, and Desmond Elliott. Revisiting transformer-based models for long document classification. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 7212–7230. Association for Computational Linguistics, 2022.
20. Zineng Tang, Jie Lei, and Mohit Bansal. DeCEMBERT: Learning from noisy instructional videos via dense captions and entropy minimization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2415–2426. Association for Computational Linguistics, 2021.

 doi:10.20944/preprints202602.0223.v1

21. Sichong Huang. Reinforcement learning with reward shaping for last-mile delivery dispatch efficiency. *European Journal of Business, Economics & Management*, 1(4):122–130, 2025.
22. Sichong Huang. Prophet with exogenous variables for procurement demand prediction under market volatility. *Journal of Computer Technology and Applied Mathematics*, 2(6):15–20, 2025.
23. Wenwen Liu. Few-shot and domain adaptation modeling for evaluating growth strategies in long-tail small and medium-sized enterprises. *Journal of Industrial Engineering and Applied Science*, 3(6):30–35, 2025.
24. Haohai Sun, Jialun Zhong, Yunpu Ma, Zhen Han, and Kun He. TimeTraveler: Reinforcement learning for temporal knowledge graph forecasting. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8306–8319. Association for Computational Linguistics, 2021.
25. Machel Reid and Victor Zhong. LEWIS: Levenshtein editing for unsupervised text style transfer. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3932–3944. Association for Computational Linguistics, 2021.
26. Reid Pryzant, Dan Iter, Jerry Li, Yin Lee, Chenguang Zhu, and Michael Zeng. Automatic prompt optimization with "gradient descent" and beam search. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7957–7968. Association for Computational Linguistics, 2023.
27. Xuanyu Zhang, Runyi Li, Jiwen Yu, Youmin Xu, Weiqi Li, and Jian Zhang. Editguard: Versatile image watermarking for tamper localization and copyright protection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11964–11974, 2024.
28. Xuanyu Zhang, Zecheng Tang, Zhipei Xu, Runyi Li, Youmin Xu, Bin Chen, Feng Gao, and Jian Zhang. Omniguard: Hybrid manipulation localization via augmented versatile deep image watermarking. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 3008–3018, 2025.
29. Zhipei Xu, Xuanyu Zhang, Runyi Li, Zecheng Tang, Qing Huang, and Jian Zhang. Fakeshield: Explainable image forgery detection and localization via multi-modal large language models. *arXiv preprint arXiv:2410.02761*, 2024.