EDITDISTILL: Is It Possible to Guide Video Editing with Image Editing?

Guojun Lei[*1], **Hong Li**[*3], **Hongbing Yang**[4], **Lixue Gong**[1], **Chi Wang**[1], **Zheng Dong**[†2]

[1] State Key Lab of CAD&CG, Zhejiang University
[2] Zhejiang Sci-Tech University [3] Beihang University [4] Tsinghua University

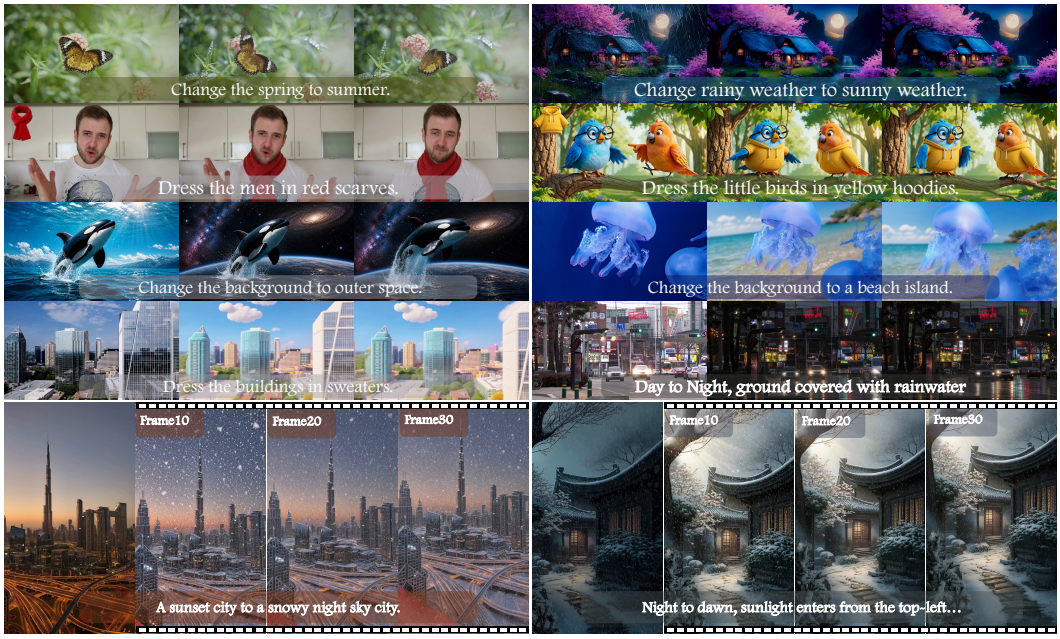

Figure 1: EDITDISTILL distills a image edit pair into a compact transformation descriptor and injects it into a video diffusion model, enabling local object edits, stylization, background replacement, and global relighting while preserving source motion and temporal coherence.

Abstract

Recent advances in image editing have enabled fine-grained and highly controllable visual modifications. However, achieving similar levels of controllability in video editing remains an open challenge, particularly for precise local edits, dynamic temporal effects, and consistent lighting across frames. In this paper, we propose EDITDISTILL, a framework that distills image-level visual transformations from edits to guide video editing. Specifically, given a source video and an edit prompt, we first generate an edited version of the initial video frame using an off-the-shelf image editing model. We then distill the visual transformation of this original-edited image pair into a compact representation termed an *edit descriptor*. This descriptor is injected into a video diffusion model through our lightweight fusion module, which steers the denoising process toward the desired edit. To align the video trajectory with the demonstrated image edit while preserving motion and layout, we train with edit-descriptor alignment and DINO structural consistency losses, and derive an equivalent inference-time guidance formulation under flow matching.

[*]Equal Contribution.
[†]Corresponding author.

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

Extensive experiments demonstrate that our method achieves strong performance on standard benchmarks and produces high-quality video editing results across diverse scenarios. For more details and videos, please refer to the anonymous webpage: `https://yu-shaonian.github.io/EditDistill-Web/`.

1 Introduction

Recent advances in AI-generated content (AIGC) have led to remarkable progress in both image and video generation. In particular, modern image editing models [Zhao et al., 2025, Wu et al., 2025a, Team et al., 2025, Team, 2025, OpenAI, 2025, 2026, G. Deepmind, 2025] have demonstrated strong capabilities in performing fine-grained and controllable edits, enabling complex local modifications with high fidelity. However, controllable video editing remains significantly more challenging, with existing methods suffering from three interrelated limitations. First, fine-grained local editing, such as modifying object details or attributes, is often unreliable and typically requires explicit masks [Hu et al., 2025], which are difficult to obtain and maintain across frames. Second, modeling dynamic background effects, such as rain, snow, or other temporally evolving phenomena, remains challenging [Wu et al., 2023] largely due to the lack of explicit temporal control signals. A similar issue was also observed in frame-wise editing methods such as I2VEdit [Ouyang et al., 2024b], which ignore the denoising process alignment during video editing. Third, existing methods struggle to jointly edit foreground and background in a consistent and controllable manner, often leading to inconsistent results [Jiang et al., 2025]. Taken together, these limitations suggest that directly controlling video editing in the video domain is inherently difficult. In contrast, we observe that such complex edits are much easier to achieve in the image domain using modern image editing models. We hypothesize that a single edited reference frame can encode the editing intent, including the visual signature of dynamic effects, while the video diffusion prior supplies the temporal unfolding. This motivates us to ask:

Can we leverage the strong controllability of modern image editing models, and distill this capability to guide fine-grained and temporally consistent video editing?

To this end, we propose EDITDISTILL, a two-stage framework that distills image-level visual transformations from edits into a compact edit descriptor and injects it into video diffusion model for controllable video editing. In the first stage, we utilize powerful image editing models (*e.g.*, Qwen-Image [Wu et al., 2025a] or GPT-Image-2 [OpenAI, 2026]) to generate an edited version of the first frame of a given video, based on an edit prompt. Given the original image and the edited image as a pair, we learn a compact edit descriptor that captures the transformation between them. Specifically, this transformation is encoded into a 1024-dimensional latent feature, which serves as guidance for the subsequent video editing process. In the second stage, we inject the edit descriptor into a video diffusion model (Wan 2.1 backbone [Wan et al., 2025]) through our lightweight fusion module. Combined with the edit descriptor alignment loss, this imposes strong constraints on the denoising trajectory, guiding the model toward the desired editing direction during the denoising process (see Fig. 2). As a result, the model can perform fine-grained and controllable video editing without requiring explicit masks. To ensure temporal consistency, we further introduce a feature alignment strategy based on the DINO [Caron et al., 2021], which aligns frame-level features similarity of the generated video with original video, thereby preserving structural coherence across frames.

Extensive experiments demonstrate that our method achieves superior performance on standard benchmarks, producing fine-grained and temporally consistent video editing results, as illustrated in Fig. 1 with both local and global editing effects. In summary, we make the following contributions:

- We introduce image-to-video edit distillation: an image pair demonstrating a visual transformation is compressed into a compact feature and reused as an explicit video editing condition.
- We design a parameter-efficient transformation injection mechanism for large video diffusion transformers, combining spatial cross-attention, global FiLM modulation, and additive residual editing while keeping the 14B backbone frozen.
- We formulate edit descriptor alignment and DINO structural similarity losses that jointly improve edit fidelity and temporal coherence, and relate them to the gradient guidance during denoising.

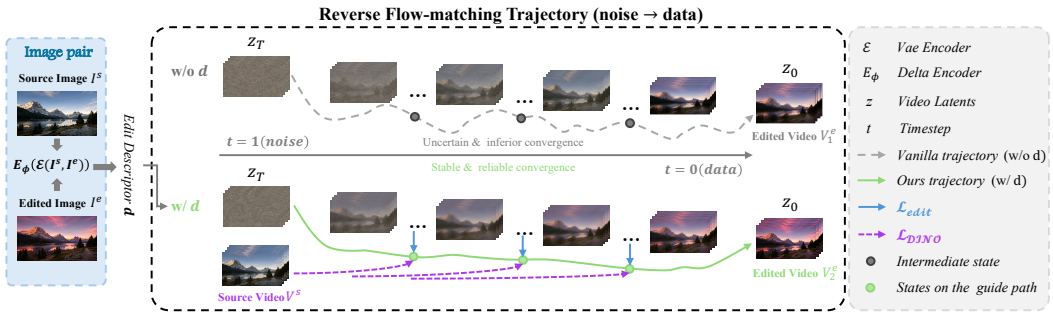

Figure 2: EDITDISTILL encodes the visual transformation between image editing pair into a compact edit descriptor, and combines the edit descriptor alignment constraint with the DINO structural similarity loss to distill the image editing capability into the denoising trajectory of video editing.

- We curate a five-family video editing dataset and show that EDITDISTILL outperforms prompt-only, inversion-based, and reference-conditioned baselines on edit fidelity, preservation, temporal consistency, and perceptual quality.

2 Related Work

Image editing with diffusion models. Recent image editing systems have demonstrated that diffusion models can follow rich editing instructions from text or images. Representative approaches include stochastic editing via noisy image trajectories [Meng et al., 2022], prompt-based editing [Hertz et al., 2023, Brooks et al., 2023], inversion-based control [Mokady et al., 2023], and latent diffusion models that enable high-resolution synthesis at scale [Rombach et al., 2022]. While these methods achieve strong performance in single-image editing, their control signals are typically defined at the image level, limiting their ability to ensure temporal consistency or to reuse learned transformations across videos. In contrast, our first stage represents an image edit as a compact feature-level transformation, enabling it to serve as a guidance signal for video editing.

Video generation and video editing. Video diffusion models provide strong priors for text-to-video and image-to-video synthesis [Blattmann et al., 2023, Wu et al., 2023, Khachatryan et al., 2023, Guo et al., 2024, Wang et al., 2025a]. Building controllable editors on top of these priors remains challenging. Existing approaches explore attention control, feature propagation, inversion, correspondence guidance, and one-shot tuning, including Video-P2P [Liu et al., 2024], TokenFlow [Geyer et al., 2024], CoDeF [Ouyang et al., 2024a], VideoSwap [Gu et al., 2024], AnyV2V [Ku et al., 2024], and RF-Solver-Edit [Wang et al., 2025b]. Recent systems broaden the control interface: HunyuanCustom [Hu et al., 2025] targets customized video generation from multimodal reference conditions, while VACE [Jiang et al., 2023] unifies video creation and editing within a single framework. UniVideo [Wei et al., 2025] combines multimodal instruction understanding with video generation and editing. Motion-control methods such as MagicMotion [Li et al., 2025], UniTransfer [Lei et al., 2025] inject trajectory conditions through a ControlNet-style branch and achieve effective object-motion control, but this design introduces a relatively heavy conditional pathway and assumes explicit trajectory or region-level controls. These methods work well when the desired change can be specified through text, inversion trajectories, reference conditions, or reliable correspondences, but they often struggle with exemplar-specific appearance changes, global relighting, or complex local edits without masks. EDITDISTILL instead replaces implicit prompt interpretation and dense control specification with an explicit learned transformation descriptor while relying on the video backbone for temporal rendering. Compared with ControlNet-style residual branches [Zhang et al., 2023], and low-rank adapters [Hu et al., 2022], our injector follows this parameter-efficient design but focus more on the diffusion process.

3 Problem Formulation

Our setting and notation. Let (I^s, I^e) be an image pair, where I^s is the source image and I^e is its edited version obtained under a given edit prompt p. Let $V^s = \{v_n^s\}_{n=1}^N$ be a source video of N frames. Our goal is to learn a video generation model F that produces an edited video $V^e = F(V^s, [I^s, I^e], p)$ by applying a visual transformation consistent with the image-level $I^s \to I^e$, while maintaining temporal coherence with respect to the source video V^s.

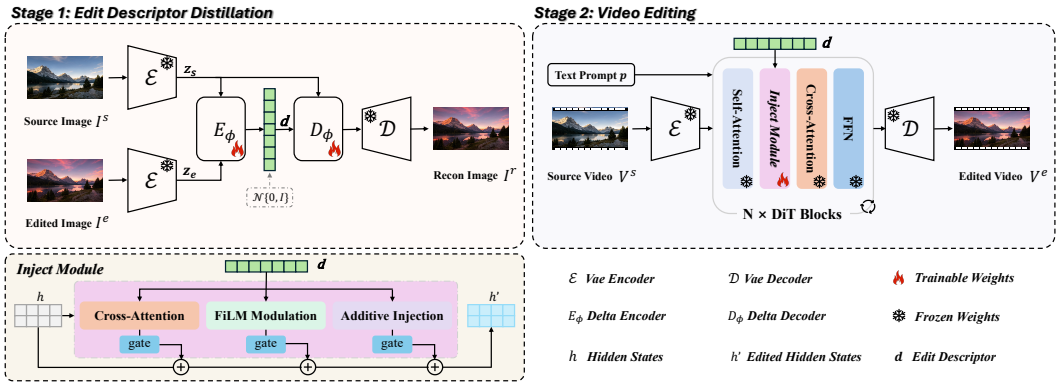

Figure 3: **Overview of EDITDISTILL. Stage I** encodes the transformation between an image pair (I^s, I^e) into the edit descriptor $\mathbf{d}$. **Stage II** keeps the *Wan2.1-I2V-14B* video diffusion frozen and trains the inject modules to inject $\mathbf{d}$ into selected transformer blocks by lightweight feature fusion modules steering the diffusion denoising toward the desired edit to produce a temporally coherent video V^e from the video V^s.

Two-stage factorization. Rather than learning a monolithic mapping from all inputs to the edited video, we introduce a compact edit descriptor $\mathbf{d} \in \mathbb{R}^m$ and factorize the problem as

$$\mathbf{d} = E_\phi(\mathcal{E}(I^s, I^e)), \quad V^e = G_\theta(V^s, p, \mathbf{d}). \tag{1}$$

Here, E_ϕ is learned from image pairs and encodes the transformation between (I^s, I^e) into a compact feature denotes edit descriptor within a additional VAE-framework network, while G_θ incorporates this feature into a pre-trained video diffusion model. This two-stage design separates transformation modeling from video generation, which enables the model to learn image-level edits and apply them consistently during video synthesis. As shown in the attention visualization in Fig. 4, which shows that the model focuses on regions relevant to the edit intent, indicating that the learned transformation effectively localizes the editing process.

4 Method : EDITDISTILL

EDITDISTILL contains two components trained sequentially. Stage I learns an image edit descriptor that captures the transformation between a before/after image pair. Stage II uses this descriptor to condition a video diffusion model. The design deliberately separates edit extraction from temporal rendering: Stage I answers what visual change should be applied, and Stage II decides where and how that change should appear in each video frame. This design further enables parameter-efficient adaptation by **training only the lightweight injection modules**, allowing the model to steer the denoising trajectory of the video diffusion toward the desired edit with a small number of trainable parameters (see pipeline in Fig. 3).

4.1 Stage I: Distilling Image Edits into Compact Descriptors

We first encode the source and edited images into latent space features using a shared frozen VAE, and the features are denoted as $z^s = \mathcal{E}_{\text{vae}}(I^s)$ and $z^e = \mathcal{E}_{\text{vae}}(I^e)$. The pair (z^s, z^e), together with their difference $(z^e - z^s)$, is fed into a multi-scale variational encoder (noted as E_ϕ) to infer the compact edit descriptor $\mathbf{d}$. Specifically, the encoder predicts a posterior distribution $q_\phi(\mathbf{d} \mid z^s, z^e) = \mathcal{N}(\mu_\phi, \text{diag}(\sigma_\phi^2))$ over a 1024-dimensional feature $\mathbf{d}$, which is obtained via reparameterized sampling during training and taken as the mean μ_ϕ at inference. Then, a conditional decoder reconstructs the edited latent as a residual transformation applied to the source $\hat{z}^e = z^s + D_\phi(z^s, \mathbf{d})$, which enforces a decomposition where z^s preserves the shared content and $\mathbf{d}$ captures the edit-specific variation. Stage I is trained with the following objective:

$$\mathcal{L}_{\text{Distill}} = \lambda_1 \left\| \hat{z}^e - z^e \right\|_1 + \lambda_2 \left\| \hat{z}^e - z^e \right\|_2^2 + \lambda_{\text{kl}} \, \text{KL}\big(q_\phi \,\|\, \mathcal{N}(0, I)\big), \tag{2}$$

which combines latent reconstruction with a KL regularizer to encourage a smooth and compact transformation space. After training, the encoder E_ϕ is frozen and used as a edit extractor in Stage II. Additional architectural details and the trade-off between representation capacity and transferability are provided in Appendix B.

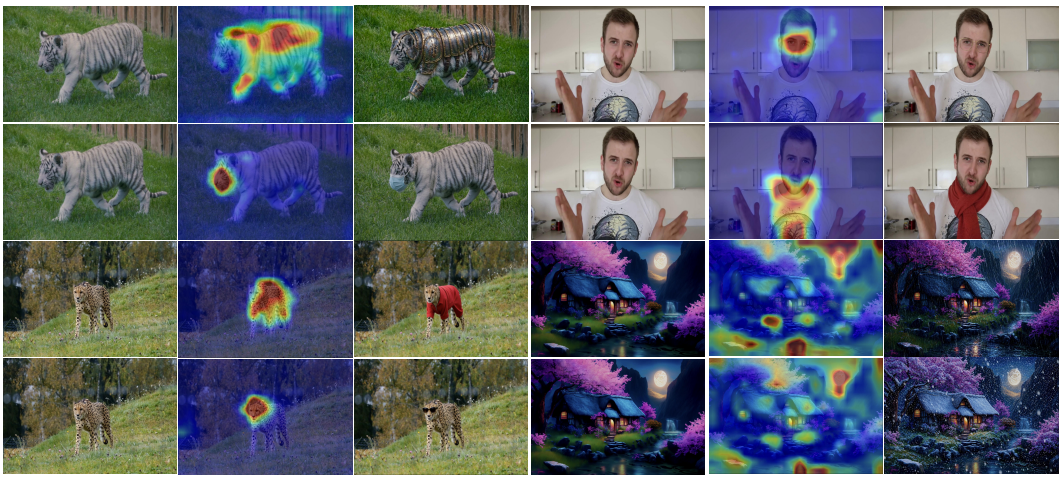

Figure 4: **Descriptor attention.** Cross-attention maps show that local edits concentrate on the regions that need to change, while global edits such as relighting or stylization produce broader but still structured responses. The descriptor behaves as a transformation operator rather than a fixed mask or copied target image.

What does the edit descriptor $\mathbf{d}$ encode? The edit descriptor together with the residual reconstruction encourages a separation of roles: z^s preserves the shared content, while $\mathbf{d}$ captures the edit-specific change. The attention maps in Fig. 4 help reveal what $\mathbf{d}$ encodes. For local edits, high responses concentrate on regions that need to change (*e.g.*, object parts or inserted attributes), while background remains largely unaffected. For global edits such as illumination or style changes, the responses become broader but still focus on meaningful regions like the sky, light sources, and main scene elements, rather than spreading uniformly. These patterns indicate that $\mathbf{d}$ does not represent a target image or a fixed mask. Instead, it specifies *what transformation to apply*, while its interaction with video features determines *where* the change is applied in each frame as illustrated by the frame-wise descriptor attention maps in Fig. 8. This separation allows the edit to transfer consistently across frames without explicit mask supervision, while preserving unaffected regions.

4.2 Stage II: Descriptor-Guided Video Diffusion

Stage II renders the distilled descriptor into a temporally coherent edited video. We adopt *Wan2.1-I2V-14B* as the base generator, trained under Flow Matching [Lipman et al., 2023] paradigm, which models a *time-dependent velocity field* $v_\theta(x_t, t, c)$ along the straight-line interpolation from data ($t = 0$) to noise ($t = 1$).

We extend this backbone with a proposed lightweight feature injection module to obtain a descriptor-conditioned velocity field $v_\Theta(x_t, t, [c_{\text{text}}, c_{\text{video}}], \mathbf{d}) \in \mathbb{R}^{N\times H\times W\times C}$, where $\mathbf{d}$ is either inferred from (I^s, I^e) or precomputed once and reused across videos. The backbone is kept entirely frozen, and only this feature fusion module is trained, enabling parameter-efficient adaptation.

Our feature injection and fusion module first adapts $\mathbf{d}$ via a shared MLP, denoted as $\bar{\mathbf{d}} = \mathrm{MLP}(\mathrm{LN}(\mathbf{d})) + \mathbf{d}$, and then injects it into the transformer blocks through three complementary branches: (1) a cross-attention branch for spatially localized edits, (2) a FiLM branch [Perez et al., 2018] for global appearance control, and (3) an additive branch for low-rank edit biases. A zero-initialized gated fusion combines the branches, so the wrapped model initially matches the frozen backbone and the edit pathway opens gradually during training. Appendix E gives details.

4.3 Training Objectives and Inference Guidance

The Stage II objective combines the standard flow-matching loss with two auxiliary losses that couple the video output to (1) the desired image-level edit and (2) the temporal structure of the source video.

Under the Rectified Flow formulation, the model predicts a velocity field $v_\Theta(x_t, t, c, \mathbf{d})$ along the linear interpolation $x_t = (1 - t)x_0 + t \cdot \varepsilon$, where the optimal target is $\varepsilon - x_0$. The flow-matching

loss is

$$\mathcal{L}_{\mathrm{FM}} = \mathbb{E}_{x_0,\varepsilon,t}\big\|v_\Theta(x_t,t,c,\mathbf{d}) - (\varepsilon - x_0)\big\|_2^2. \tag{3}$$

A one-step estimate of the clean sample is obtained as $\hat{x}_0 = x_t - t \cdot v_\Theta(x_t,t,c,\mathbf{d})$, and a reference prediction $\hat{x}_0^C$ is defined by performing the same denoising step without descriptor conditioning. To explicitly supervise the editing guidance, we introduce two losses:

$$\mathcal{L}_{\mathrm{edit}} = \tfrac{1}{N}\sum_n \big\|E_\phi(\hat{x}_{0,n}, \hat{x}_{0,n}^C) - \mathbf{d}\big\|_2^2, \qquad \mathcal{L}_{\mathrm{DINO}} = \tfrac{1}{N}\sum_n \big\|\Psi(\hat{x}_{0,n}) - \Psi(v_n^s)\big\|_2^2, \tag{4}$$

where $E_\phi(\cdot,\cdot)$ extracts the transformation between two frames, and $\Psi(\cdot)$ is a frozen DINO encoder [Caron et al., 2021]. The first term enforces *edit descriptor alignment* by matching the induced feature shift with $\mathbf{d}$, while the second enforces *temporal consistency* by preserving the structural layout of the source video. Beyond loss-based training, these objectives can be interpreted as gradient-based guidance on the denoising trajectory:

$$v_\theta^{\mathrm{edit}}(x_t^V,\, t) = \underbrace{v_\theta(x_t^V,\, t,\, c)}_{\text{base denoising direction}} - t\left(\underbrace{\alpha_{\mathrm{edit}}\, \nabla_{x_t^V}\mathcal{L}_{\mathrm{edit}}}_{\text{edit descriptor alignment}} + \underbrace{\alpha_{\mathrm{DINO}}\, \nabla_{x_t^V}\mathcal{L}_{\mathrm{DINO}}}_{\text{temporal consistency}}\right), \tag{5}$$

which steers the denoising direction toward the desired edit while preserving structural consistency. The full objective is defined with weighting factors α_{edit} and α_{DINO}, as:

$$\mathcal{L}_{\text{Stage-II}} = \mathcal{L}_{\mathrm{FM}} + \alpha_{\mathrm{edit}}\,\mathcal{L}_{\mathrm{edit}} + \alpha_{\mathrm{DINO}}\,\mathcal{L}_{\mathrm{DINO}}. \tag{6}$$

Moreover, through the score–velocity identity $\nabla_{x_t}\log p_t(x_t) = -v_\theta(x_t,t)/t$ (see Appendix 17), the same auxiliary objectives can be applied as test-time gradient guidance. This provides a unified formulation that supports both training-time conditioning and tuning-free inference-time control (details in Appendices C and D).

5 Experiments

We evaluate EDITDISTILL on different types of edits and several video benchmark datasets. Our experiments aim to answer four questions: (1) whether the image edit can be distilled to the video diffusion model, (2) whether the unedited content is preserved, (3) whether the edit is consistent over time, and (4) which parts of the method contribute most to the improvement.

5.1 Experimental Setup

Training and inference details. Stage I trains the variational transformation feature encoder E_ϕ and residual decoder D_ψ in the frozen *Wan2.1* VAE latent space at a resolution of 480×832. We use AdamW with a learning rate of 1×10^{-4}, batch size 64, and 200,000 training iterations. In Stage II, we load the frozen Stage I encoder together with the frozen *Wan2.1-I2V-14B* backbone, and train only the edit feature injection modules. Optimization is performed with AdamW using a learning rate of 1×10^{-5}, a global batch size of 16, and 30,000 iterations. Unless otherwise specified, injectors are inserted into every third transformer block of *Wan*. The loss weights are set to $\alpha_{\mathrm{edit}} = 0.5$ and $\alpha_{\mathrm{DINO}} = 0.2$. During inference, we generate 81-frame video clips at 480×832 using 50 flow-matching denoising steps. All training experiments are conducted on 16 NVIDIA H800 GPUs (80GB memory each). The trainable injection modules occupy approximately 480MB, accounting for less than 4% of the 14B backbone parameters.

Datasets. Stage I uses approximately 2M paired images aggregated from OmniGen2 and GPT-Image-Edit [Wu et al., 2025b, Wang et al., 2025c]. For stage II, We carefully constructed a large-scale video editing dataset. First, we leveraged the Wan 2.1 video generation model to synthesize preliminary video pairs from image editing pair data, while strictly enforcing identical text prompts except for the edited regions. To further improve the consistency of these coarse video-editing pairs, we use the edit loss and DINO feature loss described in the Method section, which effectively constrained both semantic and structural alignment between these video pairs. After a rigorous filtering process, we ultimately obtained 100K high-quality video editing samples. In addition, for the training data, we also constructed paired samples from the OpenVid [Nan et al., 2024] dataset by randomly injecting noise into videos and randomly shuffling video frames, using the original real videos as reconstruction targets.

Table 1: **Quantitative comparisons.** All methods run at matched 480×832 and 81-frame experimental settings. For evaluation metrics, CLIP-E and CLIP-T measure edit fidelity; DINO-sim measures preservation of unedited regions; Temp. and Warp measure temporal coherence; FVD measures perceptual video quality. Best results are in **bold** and second best results are underlined.

Method	CLIP-E↑	CLIP-T↑	DINO-sim↑	Temp.↑	Warp↓	FVD↓
InstructPix2Pix [Brooks et al., 2023]	0.221	0.288	0.534	0.892	7.15	1284.7
TokenFlow [Geyer et al., 2023]	0.196	0.275	0.668	0.948	4.21	1052.3
AnyV2V [Ku et al., 2024]	0.137	0.251	0.591	0.856	6.94	1485.2
RF-Solver-Edit [Wang et al., 2025b]	0.224	0.286	0.654	0.926	4.58	996.3
MOFA-Video [Niu et al., 2024]	0.182	0.271	0.621	0.943	4.83	1024.1
Prompt-only Wan2.1 [Wang et al., 2025a]	0.158	0.291	0.612	0.951	3.84	982.4
Omni-Video [Tan et al., 2025]	0.218	0.284	0.661	0.939	4.34	968.7
HunyuanCustom [Hu et al., 2025]	0.236	<u>0.295</u>	0.687	0.949	3.71	941.2
VACE [Jiang et al., 2023]	**0.247**	**0.302**	<u>0.701</u>	<u>0.957</u>	<u>3.42</u>	<u>914.6</u>
EDITDISTILL (ours)	<u>0.245</u>	0.282	**0.738**	**0.964**	**2.97**	**852.1**

Evaluation metrics. No single metric fully captures video editing quality, so we report complementary metrics. Edit fidelity is evaluated by CLIP-Edit Similarity (CLIP-E), defined as the similarity between the image edit $\mathrm{CLIP}(I^e) - \mathrm{CLIP}(I^s)$ and the average video edit direction $\mathrm{CLIP}(v_n^e) - \mathrm{CLIP}(v_n^s)$, and by CLIP-text alignment (CLIP-T) [Radford et al., 2021]. Preservation is measured by DINO-sim on regions outside the edit estimated by Grounded-SAM2 [Ren et al., 2024, Ravi et al., 2024, Oquab et al., 2023]. Temporal coherence is assessed using CLIP-based adjacent-frame consistency (Temp.) and optical-flow warp error (Warp) [Huang et al., 2024], while perceptual quality is measured by FVD [Unterthiner et al., 2019]. We further report VBench-style metrics, including subject consistency (SubC), background consistency (BkgC), motion smoothness (MoS), aesthetic quality (AesQ), and dynamic degree (DyaD), for the consistency of video quality. We compare with nine representative methods following similar way in VACE-benchmark covering frame-wise editing, feature propagation, inversion, motion-conditioned generation, reference conditioning, and unified video editing: prompt-only Wan2.1-I2V-14B [Wang et al., 2025a], InstructPix2Pix [Brooks et al., 2023], TokenFlow [Geyer et al., 2023], AnyV2V [Ku et al., 2024], RF-Solver-Edit [Wang et al., 2025b], MOFA-Video [Niu et al., 2024], Omni-Video [Tan et al., 2025], HunyuanCustom [Hu et al., 2025], and VACE [Jiang et al., 2023]. When a baseline supports image references, we provide the edited first frame I^e; otherwise we provide the corresponding edit caption. All methods run at matched 480×832 resolution and 81-frame inference budget using their public high-quality settings.

5.2 Main Experimental Results

Edit fidelity, preservation, and perceptual quality. Tab. 1 reports averages over the five edit families. EDITDISTILL achieves strong result on every metric, improving DINO-sim from 0.701 for the strongest baseline VACE, to 0.738. This indicates that the distilled transformation feature matches the demonstrated image-pair edit more faithfully than prompt-only or reference-image conditioning. The gains are not limited to edit strength: EDITDISTILL also reduces Warp error from 3.42 to 2.97, and lowers FVD from 914.6 to 852.1, showing better preservation, temporal coherence, and perceptual quality. These results suggest that the edit feature steers the desired transformation without forcing the video model to repaint unrelated contents.

Video consistency under dynamic motion. Tab. 2 further evaluates video-level consistency. EDITDISTILL obtains the best BkgC, AesQ, Mos and DyaD. This is important because high consistency can otherwise be achieved by producing conservative or nearly static videos. In contrast, EDITDISTILL preserves subject and background stability while maintaining the highest dynamic degree, suggesting that the edit remains coherent under non-trivial motion rather than being restricted to easy, low-motion cases.

Qualitative comparison. Fig. 5 provides a qualitative comparison that helps explain the differences behind the above metrics. Prompt-driven methods often capture the coarse caption but miss small exemplar-specific attributes, such as accessories or fine texture changes. Frame-wise and inversion-based methods can make a sharper local change in the first few frames, but the edit may jitter, weaken, or contaminate nearby regions as the clip evolves. Reference-conditioned methods better preserve the target appearance, yet they sometimes transfer irrelevant color or texture from the reference image into the background. EDITDISTILL is more stable because the image pair defines a transformation

Table 2: **Comparisons of video editing consistency.** SubC: subject consistency; BkgC: background consistency; MoS: motion smoothness; AesQ: aesthetic quality; DyaD: dynamic degree. Best results are in **bold**; second best results are underlined.

Method	SubC↑	BkgC↑	MoS↑	AesQ↑	DyaD↑
InstructPix2Pix [Brooks et al., 2023]	0.842	0.871	0.924	0.429	0.612
TokenFlow [Geyer et al., 2023]	0.946	0.951	0.972	0.514	0.524
AnyV2V [Ku et al., 2024]	0.812	0.768	0.912	0.431	0.512
RF-Solver-Edit [Wang et al., 2025b]	0.928	0.939	0.967	0.571	0.628
MOFA-Video [Niu et al., 2024]	0.932	0.945	0.974	0.568	0.718
Prompt-only Wan2.1 [Wang et al., 2025a]	0.921	0.943	0.944	0.562	0.681
Omni-Video [Tan et al., 2025]	**0.964**	0.952	0.971	0.583	0.654
HunyuanCustom [Hu et al., 2025]	0.949	0.958	0.978	0.594	0.692
VACE [Jiang et al., 2023]	0.954	0.962	0.953	0.612	0.671
EDITDISTILL (ours)	0.942	**0.969**	**0.981**	**0.645**	**0.724**

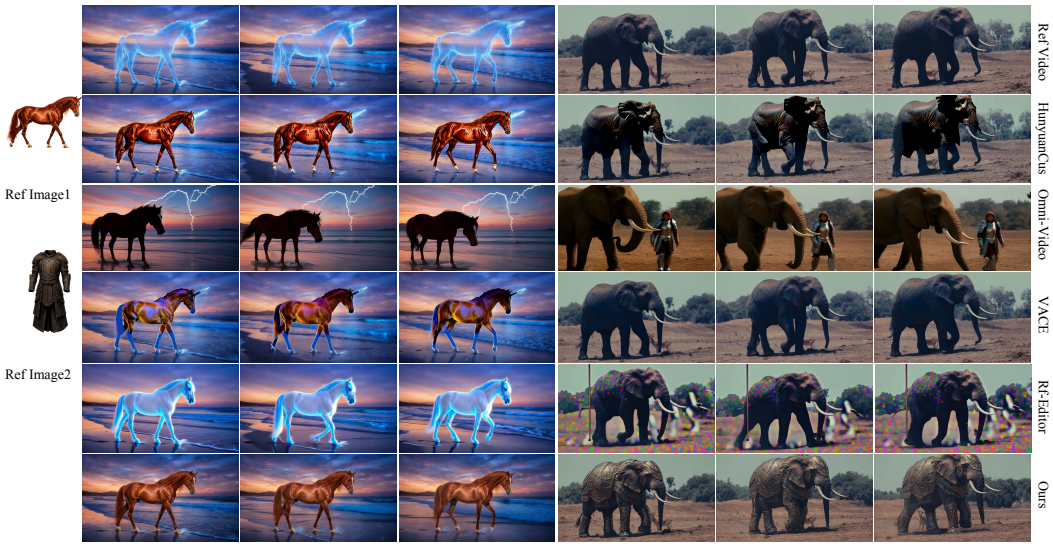

Figure 5: **Qualitative comparison.** EDITDISTILL performs both local and global edits while keeping the main subject, background, and motion stable. In contrast, other methods may miss the desired edit, unintentionally change nearby regions, or produce unstable results across frames.

rather than a target appearance: the distilled transformation feature $\mathbf{d}$ indicates what should change, and cross-attention to video tokens determines where the change should be applied.

Fig. 6 highlights the challenging setting of video relighting. Unlike local editing methods or reference-based appearance transfer approaches, relighting requires globally adjusting scene illumination while preserving object geometry, motion, and temporal consistency. Compared with existing methods, EDITDISTILL better captures the influence of global lighting changes and produces more coherent responses to dynamic indirect illumination over time.

5.3 Ablation Studies

We ablate the injection branches and auxiliary losses on the held-out benchmark. Each variant is trained under the same data, resolution, and inference frame budget as the full model.

Injection branches. Tab. 3 shows that each operation in the feature injection module contributes a distinct capability. Cross-attention is the strongest branch for edit fidelity, reaching 0.243 CLIP-E, because it can route the edit feature to spatially relevant tokens and therefore suits local edits particularly well. FiLM obtains higher DINO-sim than cross-attention but lower CLIP-E, which matches its role as a global modulator: it preserves layout and overall scene consistency well but has limited spatial selectivity. The additive branch is weakest alone, yet it provides a useful low-rank residual bias when combined with the other branches. The full gated fusion improves CLIP-E by 0.038 over cross-attention alone and lowers FVD by 69.3, demonstrating that the gains are not merely due to added parameters but arise from combining complementary conditioning routes.



Figure 6: **Relighting comparison.** The distilled descriptor captures a global lighting transformation and guides coherent relighting across frames while preserving scene geometry and motion.

Table 3: **Ablation study of the edit feature injection module.** The three branches are complementary: cross-attention is strongest for local routing, FiLM improves global appearance control, and the additive branch contributes a stable residual edit bias.

Variant	Trainable params	CLIP-E↑	DINO-sim↑	Warp↓	FVD↓
Cross-attention only	312MB	0.243	0.692	3.41	921.4
FiLM only	187MB	0.218	0.704	3.55	938.2
Additive only	154MB	0.196	0.681	3.78	967.5
All branches (ours)	480MB	**0.245**	**0.738**	**2.97**	**852.1**

Auxiliary losses. Tab. 4 also separates two failure modes. Flow matching alone gives a smooth video prior but weak pressure to realize the demonstrated edit, resulting in low CLIP-E. Adding $\mathcal{L}_{\text{edit}}$ raises CLIP-E from 0.184 to 0.273, confirming that the Stage I descriptor provides a meaningful target for video-level edit alignment. However, DINO-sim and Warp improve only modestly, so edit alignment alone does not fully prevent structural drift. Conversely, adding only $\mathcal{L}_{\text{DINO}}$ improves preservation and temporal coherence but leaves edit fidelity low, showing that structure preservation can suppress the desired transformation unless paired with an explicit edit objective. The full objective combines both terms and achieves the best score across all reported metrics. Qualitative examples in Figure 11 show the same trend: branch removal weakens edit fidelity, while removing alignment losses increases temporal or structural drift.

Summary of experimental evidence. The results show a consistent pattern. Strong video priors can be smooth but under-edit; strong image or inversion control can edit individual frames but lose temporal stability; reference-conditioned methods improve appearance transfer but may copy irrelevant details. EDITDISTILL improves this trade-off by representing the user intent as a compact transformation descriptor and injecting it directly into the denoising dynamics of a frozen video backbone, enabling controllable edits with temporal coherence and structural consistency.

6 Conclusion

In this paper, we propose EDITDISTILL, a two-stage framework for transformation-guided video editing that distills image-level editing description into a compact feature and injects it into a video diffusion model. By guiding the denoising process with this feature, together with edit-descriptor alignment and DINO structural consistency losses for temporal consistency, our method enables controllable video editing while preserving the generative prior of the backbone. Our framework provides a practical paradigm for integrating image-level editing into video generation, with potential applications in content creation, video post-production, and interactive media editing.

Table 4: **Auxiliary-loss ablation.** Removing $\mathcal{L}_{\text{edit}}$ weakens edit transfer; removing $\mathcal{L}_{\text{DINO}}$ increases structural drift and warp error.

Variant	CLIP-E↑	DINO-sim↑	Temp.↑	Warp↓	FVD↓
$\mathcal{L}_{\text{FM}}$ only	0.184	0.671	0.943	3.86	967.4
$\mathcal{L}_{\text{FM}} + \mathcal{L}_{\text{edit}}$	0.273	0.682	0.948	3.61	902.1
$\mathcal{L}_{\text{FM}} + \mathcal{L}_{\text{DINO}}$	0.198	0.731	0.961	3.04	884.6
Full objective	**0.245**	**0.738**	**0.964**	**2.97**	**852.1**

References

Andreas Blattmann, Tim Dockhorn, Suraj Kulal, Dimitrios Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, and Bjorn Ommer. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023.

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.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9650–9660, 2021.

G. Deepmind. Introducing nano banana pro, 2025. URL `https://nano-banana-pro/`.

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

Michal Geyer, Omer Bar-Tal, Shai Bagon, and Tali Dekel. Tokenflow: Consistent diffusion features for consistent video editing. In *International Conference on Learning Representations*, 2024.

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*, pages 7621–7630, 2024.

Yuwei Guo, Ceyuan Yang, Anyi Rao, Yaohui Wang, Yu Qiao, Dahua Lin, and Bo Dai. Animatediff: Animate your personalized text-to-image diffusion models without specific tuning. In *International Conference on Learning Representations*, 2024.

Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross-attention control. In *International Conference on Learning Representations*, 2023.

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

Teng Hu, Zhentao Yu, Zhengguang Zhou, Sen Liang, Yuan Zhou, Qin Lin, and Qinglin Lu. Hunyuancustom: A multimodal-driven architecture for customized video generation, 2025. URL `https://arxiv.org/abs/2505.04512`.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. VBench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21807–21818, 2024.

Zeyinzi Jiang, Zhen Han, Chaojie Mao, Jingfeng Zhang, Yulin Pan, and Yu Liu. VACE: All-in-one video creation and editing. *arXiv preprint arXiv:2503.07598*, 2023.

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.

Levon Khachatryan, Andranik Movsisyan, Vahram Tadevosyan, Roberto Henschel, Zhangyang Wang, Shant Navasardyan, and Humphrey Shi. Text2video-zero: Text-to-image diffusion models are zero-shot video generators. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15954–15964, 2023.

Max Ku, Cong Wei, Weiming Ren, Harry Yang, and Wenhu Chen. Anyv2v: A tuning-free framework for any video-to-video editing tasks. *arXiv preprint arXiv:2403.14468*, 2024.

Guojun Lei, Rong Zhang, Chi Wang, Tianhang Liu, Hong Li, Zhiyuan Ma, and Weiwei Xu. Unitransfer: Video concept transfer via progressive spatial and timestep decomposition, 2025. URL `https://arxiv.org/abs/2509.21086`.

Quanhao Li, Zhen Xing, Rui Wang, Hui Zhang, Qi Dai, and Zuxuan Wu. Magicmotion: Controllable video generation with dense-to-sparse trajectory guidance. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12112–12123, October 2025.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

Shaoteng Liu, Yuechen Zhang, Wenbo Li, Zhe Lin, and Jiaya Jia. Video-p2p: Video editing with cross-attention control. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8599–8608, 2024.

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

Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6038–6047, 2023.

Kepan Nan, Rui Xie, Penghao Zhou, Tiehan Fan, Zhenheng Yang, Zhijie Chen, Xiang Li, Jian Yang, and Ying Tai. Openvid-1m: A large-scale high-quality dataset for text-to-video generation. *arXiv preprint arXiv:2407.02371*, 2024.

Muyao Niu, Xiaodong Cun, Xintao Wang, Yong Zhang, Ying Shan, and Yinqiang Zheng. Mofa-video: Controllable image animation via generative motion field adaptions in frozen image-to-video diffusion model. *arXiv preprint arXiv:2405.20222*, 2024.

OpenAI. GPT-Image-1, 2025. URL `https://platform.openai.com/docs/models/gpt-image-1`.

OpenAI. GPT-Image-2, 2026. URL `https://platform.openai.com/docs/models/gpt-image-2`.

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.

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*, pages 8089–8099, 2024a.

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*, 2024b.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763, 2021.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. URL `https://arxiv.org/abs/2408.00714`.

Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, et al. Grounded SAM: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159*, 2024.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Bjorn 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.

Zhiyu Tan, Hao Yang, Luozheng Wu, Jiyuan Yang, Yu Tao, Jiaxin Liang, Lin Yan, and Hao Li. Omni-Video: Democratizing unified video understanding and generation. *arXiv preprint arXiv:2507.06119*, 2025.

Meituan LongCat Team, Hanghang Ma, Haoxian Tan, Jiale Huang, Junqiang Wu, Jun-Yan He, Lishuai Gao, Songlin Xiao, Xiaoming Wei, Xiaoqi Ma, Xunliang Cai, Yayong Guan, and Jie Hu. Longcat-image technical report. *arXiv preprint arXiv:2512.07584*, 2025.

Z-Image Team. Z-image: An efficient image generation foundation model with single-stream diffusion transformer. *arXiv preprint arXiv:2511.22699*, 2025.

Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. FVD: A new metric for video generation. *ICLR Workshop on Deep Generative Models for Highly Structured Data*, 2019.

Team Wan, Ang Wang, and etc Baole Ai. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

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

Jiangshan Wang, Junfu Pu, Zhongang Qi, Jiayi Guo, Yue Ma, Nisha Huang, Yuxin Chen, Xiu Li, and Ying Shan. Taming rectified flow for inversion and editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025b.

Yuhan Wang, Siwei Yang, Bingchen Zhao, Letian Zhang, Qing Liu, Yuyin Zhou, and Cihang Xie. Gpt-image-edit-1.5m: A million-scale, gpt-generated image dataset, 2025c. URL `https://arxiv.org/abs/2507.21033`.

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.

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*, 2025a.

Chenyuan Wu, Pengfei Zheng, Ruiran Yan, Shitao Xiao, Xin Luo, Yueze Wang, Wanli Li, Xiyan Jiang, Yexin Liu, Junjie Zhou, Ze Liu, Ziyi Xia, Chaofan Li, Haoge Deng, Jiahao Wang, Kun Luo, Bo Zhang, Defu Lian, Xinlong Wang, Zhongyuan Wang, Tiejun Huang, and Zheng Liu. Omnigen2: Exploration to advanced multimodal generation. *arXiv preprint arXiv:2506.18871*, 2025b.

Jay Zhangjie Wu, Yixiao Ge, Xintao Wang, Weixian Lei, Yuchao Gu, 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*, pages 7623–7633, 2023.

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.

Shanshan Zhao, Xinjie Zhang, Jintao Guo, Jiakui Hu, Lunhao Duan, Minghao Fu, Yong Xien Chng, Guo-Hua Wang, Qing-Guo Chen, Zhao Xu, et al. Unified multimodal understanding and generation models: Advances, challenges, and opportunities. *arXiv preprint arXiv:2505.02567*, 2025.

A Limitations and Impact

EDITDISTILL still has several limitations. Because the descriptor is injected implicitly into the denoising process, highly localized edits may also affect nearby regions because the conditioning is not restricted by an explicit spatial mask. The first frame is the default reference for evaluation, not a requirement of descriptor extraction: a later keyframe can specify an edit when the subject becomes visible. Nevertheless, a single reference may be unrepresentative under large motion, occlusion, or scene changes. Segment-specific descriptors are a possible extension, but are not evaluated here. Large geometric changes, dynamic object removal, and precise viewpoint or camera control remain challenging. The descriptor does not provide explicit temporal control, and performance depends on the image editor and the quality of the reference pair. Conflicting text and image conditions may degrade results. Systematic evaluation of long-tail edits, out-of-distribution transfer, prompt robustness, and long videos remains limited.

This work enables transformation-guided video editing by transferring image-level edits to videos, with applications in creative content production, video post-production, and assistive visual tools. It also supports controllable simulation of environmental changes such as lighting and weather.At the same time, improved editing capability may increase the risk of deceptive media generation, identity manipulation, and unauthorized content modification. By constraining edit specification through visual exemplars, the method may lower the barrier for targeted misuse.This work does not involve human-subject experiments. Responsible use should emphasize usage restrictions, provenance tracking, and watermarking or other disclosure mechanisms. In addition, training data and exemplars should respect privacy, copyright, and licensing constraints. We recommend treating portrait editing and safety-sensitive scenarios as restricted deployment settings.

B Stage I Architectural Details

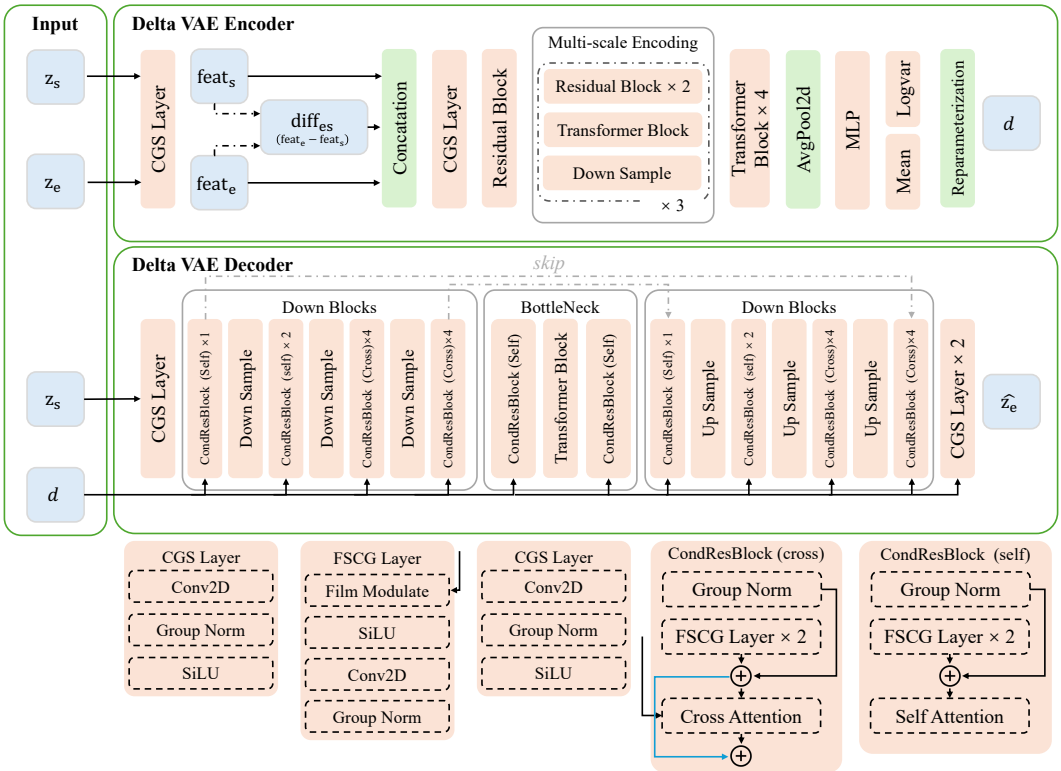

Figure 7: **Stage I edit descriptor model.** The Delta VAE encoder extracts a stochastic descriptor from the source and edited VAE latents by combining shared features, their explicit feature difference, and a multi-scale encoder. The Delta VAE decoder receives only the source latent and descriptor, and reconstructs the edited latent through descriptor-conditioned residual blocks with self- and cross-attention.

Stage I is implemented as the Delta VAE shown in Figure 7. It operates entirely in the frozen Wan VAE latent space: the source and edited images are encoded as $z^s, z^e \in \mathbb{R}^{4\times H\times W}$. The encoder observes both latents and compresses the edit into a 1024-dimensional stochastic descriptor $\mathbf{d}$, whereas the decoder observes only z^s and $\mathbf{d}$. This asymmetric design is important: the edited target is available when extracting the descriptor, but it is never passed to the reconstruction decoder through skip connections. As a result, target-specific information must pass through the compact descriptor rather than being copied spatially from z^e.

Latent-delta encoder. The encoder E_ϕ extracts the transformation from the pair (z^s, z^e). As illustrated in the upper part of Figure 7, the two latents first pass through a shared CGS stem, where each CGS layer consists of convolution, group normalization, and SiLU activation:

$$f^s = S(z^s), \qquad f^e = S(z^e), \tag{7}$$

so the source and edited features are represented in a comparable basis. We then form an explicit feature-level edit residual

$$\Delta f = f^e - f^s, \tag{8}$$

and concatenate $[f^s, f^e, \Delta f]$ before another CGS projection and a residual block. This explicit delta path makes the encoder focus on what changed, while still retaining the source and target context needed to disambiguate edits such as object insertion, material replacement, relighting, and stylization.

The projected feature is processed by a multi-scale hierarchy. Each scale contains two residual blocks, a transformer block, and a downsampling operation; this sequence is repeated three times. Local residual blocks preserve fine edit evidence, the downsampling path enlarges the receptive field, and transformer blocks allow non-local interactions between changed and unchanged regions. A final stack of four transformer blocks aggregates the coarsest representation before adaptive average pooling removes spatial coordinates. An MLP then predicts the mean and log-variance of a diagonal Gaussian descriptor distribution:

$$q_\phi(\mathbf{d} \mid z^s, z^e) = \mathcal{N}\left(\mu_\phi(z^s, z^e), \mathrm{diag}(\sigma^2_\phi(z^s, z^e))\right), \qquad \mathbf{d} \in \mathbb{R}^{1024}. \tag{9}$$

During training, $\mathbf{d}$ is sampled using the reparameterization trick. At inference time, we use the posterior mean as a deterministic descriptor.

FiLM-conditioned residual decoder. The decoder D_ψ reconstructs the edited latent from $(z^s, \mathbf{d})$ only:

$$\hat{z}^e = z^s + D_\psi(z^s, \mathbf{d}). \tag{10}$$

The source latent first passes through a CGS layer and then a U-Net-like residual decoder with down blocks, a bottleneck, up blocks, and skip connections from matching resolutions. The skip connections carry source layout and identity, but only from z^s; no feature from z^e enters the decoder.

The descriptor conditions every decoder stage through FSCG layers and conditional residual blocks. An FSCG layer applies descriptor-derived FiLM modulation, SiLU activation, convolution, and group normalization. In the down and up paths, conditional residual blocks use self-attention to preserve source structure and cross-attention to route descriptor information to spatial positions that require the edit. The bottleneck combines descriptor-conditioned residual processing with a transformer block, giving the decoder a global view before upsampling. The final CGS head predicts a four-channel residual in VAE latent space, which is added to z^s to obtain $\hat{z}^e$. This residual formulation lets the decoder leave unchanged regions close to the source while concentrating descriptor capacity on the edit itself.

Bottleneck capacity. We use a 1024-dimensional descriptor by default. Smaller descriptors lose fine edit attributes such as local insertions, material changes, and subtle lighting transformations. Larger descriptors improve reconstruction only marginally but are more prone to source-specific leakage. The chosen capacity balances reconstruction fidelity and transferability.

Stage II implication. Because Stage I forces the edit through a global descriptor and applies it through conditional modulation, $\mathbf{d}$ behaves as an edit operator rather than a spatial template. Stage II can therefore inject the same descriptor into video transformer blocks and let the video features determine where the edit should appear in each frame. This is why the same image-level descriptor can re-localize to moving regions over time, as shown in Figure 8.

C Full Flow-Matching Derivation for Stage II

C.1 Forward Interpolation Path

The forward process is defined as a straight-line interpolation between a data sample x_0 and a noise vector ε:

$$x_t = (1-t)\, x_0 + t\, \varepsilon, \qquad \varepsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \quad t \in [0,1]. \tag{11}$$

Here $t = 0$ corresponds to clean data and $t = 1$ to pure noise.

C.2 Training Objective

The model is trained to regress the *true velocity* along the interpolation path:

$$\mathcal{L}_{\mathrm{FM}} = \mathbb{E}_{t,\, x_0,\, \varepsilon}\Big[\big\| v_\theta(x_t,\, t,\, c) - (\varepsilon - x_0) \big\|^2\Big], \tag{12}$$

where c is the conditioning signal.

C.3 Reverse ODE for Inference

Given a trained vector field, new samples are generated by solving the reverse ODE:

$$\frac{dx_t}{dt} = v_\theta(x_t,\, t,\, c) = \varepsilon - x_0, \qquad x_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{13}$$

integrating from $t = 1$ to $t = 0$. In practice, this is discretized with an Euler solver or DPM-Solver.

C.4 Notation

Symbol	Meaning
I^s	Source reference image (e.g. a tiger without glasses)
I^e	Edited reference image (e.g. the same tiger wearing glasses)
$V^s = \{v_n^s\}_{n=1}^N$	Source video of N frames
$V^e = \{v_n^e\}_{n=1}^N$	Target: edited video (tiger wears glasses in every frame)
$x_t^V \in \mathbb{R}^{N\times H\times W\times C}$	Noisy video latent at time t
E_ϕ	Delta VAE Encoder
$\mathrm{DINO}(\cdot)$	Self-supervised DINO feature extractor
c	Conditional of base video model

C.5 Edit Consistency Assumption

We first define the image-level **edit descriptor**:

$$\Delta F^{\mathrm{edit}} \;\triangleq\; E_\phi(I^s, I^e) \;\in\; \mathbb{R}^{B'\times d}. \tag{14}$$

Definition C.1 (Edit Consistency)**.** *The editing transformation encoded in ΔF^{edit} should be* consistently *applied to every frame of the video. Formally, for each frame n:*

$$E_\phi(\hat{v}_n^s, \hat{v}_n^e) \;\approx\; \Delta F^{\mathrm{edit}}, \tag{15}$$

where $\hat{v}_n^s$ and $\hat{v}_n^e$ denote the n-th frame of the original and edited video predictions, respectively.

C.6 Additive Decomposition of the Vector Field

We decompose the editing vector field into an *unconstrained base prediction* and an *edit guidance correction*:

$$v_\theta^{\text{edit}}(x_t^V,\, t) = \underbrace{v_\theta(x_t^V,\, t,\, c)}_{\text{base prediction}} + \underbrace{\Delta v_t^{\text{guidance}}}_{\text{guidance correction}}. \tag{16}$$

C.7 Score Function and Vector Field Correspondence

Proposition 1 (Score–Velocity Duality)**.** *Under the linear interpolation path* (11)*, the score function of the marginal p_t satisfies:*

$$\nabla_{x_t} \log p_t(x_t) = -\frac{v_\theta(x_t,\, t)}{t}. \tag{17}$$

Proof. From (11), the conditional distribution is $p_t(x_t \mid x_0) = \mathcal{N}\big(x_t;\, (1-t)x_0,\, t^2\mathbf{I}\big)$, so

$$\nabla_{x_t} \log p_t(x_t \mid x_0) = -\frac{x_t - (1-t)x_0}{t^2} = -\frac{\varepsilon}{t}.$$

Taking the conditional expectation over x_0 given x_t yields Eq. 17. □

C.8 Bayesian Guidance Framework

Conditioning on additional edit descriptor features F_t^e, the joint score function decomposes as:

$$\nabla_{x_t^V} \log p_t\big(x_t^V,\, F_t^e\big) = \nabla_{x_t^V} \log p_t(x_t^V) + \nabla_{x_t^V} \log p_t\big(F_t^e \mid x_t^V\big)\,. \tag{18}$$

Substituting (17) into (18) and converting back to vector field language yields the **guided vector field**:

$$\boxed{v_\theta^{\text{edit}}(x_t^V,\, t) = v_\theta(x_t^V,\, t,\, c) \;-\; t \cdot \nabla_{x_t^V} \mathcal{L}_{\text{guidance}}(x_t^V,\, t)} \tag{19}$$

C.9 Edit Feature Alignment Loss $\mathcal{L}_{\text{edit}}$

Step 1. Compute the image edit residual (computed *once*, frozen):

$$\Delta F^{\text{edit}} \;\triangleq\; E_\phi(I^s, I^e) \;\in\; \mathbb{R}^{B' \times d}. \tag{20}$$

Step 2. Enforce the edit consistency assumption (15) across all N frames:

$$\mathcal{L}_{\text{edit}} = \sum_{n=1}^{N} \Big\| E_\phi(\hat{v}_n^s, \hat{v}_n^e) - \Delta F^{\text{edit}} \Big\|_2^2, \tag{21}$$

where $\hat{v}_n^s$ and $\hat{v}_n^e$ denote the n-th frame of $\mathbf{V}^s$ and $\mathbf{V}^e$, respectively.

Remark 1. *Equation* (21) *enforces that the per-frame feature shift induced by the editing process equals the image-level editing direction ΔF^{edit}. In other words, the edit semantics ("wearing glasses") are propagated consistently to every frame without requiring any frame-specific supervision.*

C.10 Temporal DINO Consistency Loss $\mathcal{L}_{\text{DINO}}$

To preserve the unedited structure of the video (body motion, background, etc.), we align the DINO features of the predicted frames to those of the original video:

$$\mathcal{L}_{\text{DINO}} = \sum_{n=1}^{N} \left\| \text{DINO}\big(\hat{v}_n^e\big) - \text{DINO}(v_n^s) \right\|_2^2, \tag{22}$$

where $\text{DINO}(v_n^s)$ is the DINO feature of the n-th original frame, computed **once** and kept frozen during inference.

Remark 2. *DINO features are sensitive to* semantic structure *(pose, layout, shape) but relatively insensitive to local appearance changes (e.g. color overlays or accessory additions). This makes them an ideal regularizer: they penalize structural drift without suppressing the desired edit.*

C.11 Total Guidance Loss

Combining both terms:

$$\boxed{\mathcal{L}_{\text{guidance}} = \alpha_{\text{edit}} \cdot \mathcal{L}_{\text{edit}} + \alpha_{\text{DINO}} \cdot \mathcal{L}_{\text{DINO}}} \tag{23}$$

where $\alpha_{\text{edit}},\ \alpha_{\text{DINO}} > 0$ are balancing hyperparameters.

C.12 Complete Guided Vector Field

Substituting (23) into (19), the full **image-guided video editing vector field** is:

$$\boxed{v_\theta^{\text{edit}}(x_t^V,\, t) = \underbrace{v_\theta(x_t^V,\, t,\, c^A)}_{\text{base denoising direction}} - t \left(\underbrace{\alpha_{\text{edit}}\, \nabla_{x_t^V} \mathcal{L}_{\text{edit}}}_{\text{edit feature alignment}} + \underbrace{\alpha_{\text{DINO}}\, \nabla_{x_t^V} \mathcal{L}_{\text{DINO}}}_{\text{temporal consistency}} \right)} \tag{24}$$

Remark 3. *The two guidance terms play complementary roles:*

- ***Edit alignment*** *steers the denoising trajectory toward the edited state, injecting the image-level edit direction $I^s \to I^e$ into every video frame.*
- ***DINO consistency*** *acts as a regularizer, preventing structural drift in parts of the video that should remain unchanged (e.g. tiger's body, motion, background).*

D Inference-Time Guidance Algorithm

Algorithm 1 describes optional inference-time guidance. The trained transformation-conditioned field v_Θ serves as the base velocity. At each step, we compute a one-step clean prediction,

$$\hat{h}_0 = h_t - t\, v_\Theta(h_t, t, c, \mathbf{d}), \tag{25}$$

measure edit and DINO losses on that prediction, and correct the velocity by the gradient in Equation 24. Gradient clipping prevents rare feature-space outliers from destabilizing the trajectory.



Figure 8: **Frame-wise descriptor attention.** The same edit descriptor re-localizes to the relevant moving regions across different video frames.

Algorithm 1 Inference-time guidance for EDITDISTILL.

Input: Source video $V^s = \{v_n^s\}_{n=1}^N$, image pair (I^s, I^e), edit descriptor feature $\mathbf{d}$, Wan conditions $c = (c_{\text{text}}, c_{\text{img}})$, Euler steps T, weights $\alpha_{\text{edit}}, \alpha_{\text{DINO}}$, gradient clip $\lambda_{\max}$.
Output: Edited video V^e.

Initialize:
$h_1 \sim \mathcal{N}(0, I)$
Precompute $\{\Psi(v_n^s)\}_{n=1}^N$ with frozen DINO
for $k = T, T-1, \ldots, 1$ **do**
  $t \leftarrow k/T$, $\Delta t \leftarrow 1/T$
  $v_{\text{base}} \leftarrow v_\Theta(h_t, t, c, \mathbf{d})$
  $\hat{h}_0 \leftarrow h_t - t\, v_{\text{base}}$
  Construct source reference $\hat{h}_0^s$ by noising V^s to time t and denoising without $\mathbf{d}$
  $\mathcal{L}_{\text{edit}} \leftarrow \frac{1}{N}\sum_n \|E_\phi(\hat{h}_{0,n}^s, \hat{h}_{0,n}) - \mathbf{d}\|_2^2$
  $\mathcal{L}_{\text{DINO}} \leftarrow \frac{1}{N}\sum_n \|\Psi(\hat{h}_{0,n}) - \Psi(v_n^s)\|_2^2$
  $g \leftarrow \nabla_{h_t}(\alpha_{\text{edit}}\mathcal{L}_{\text{edit}} + \alpha_{\text{DINO}}\mathcal{L}_{\text{DINO}})$
  $g \leftarrow g \cdot \min(1, \lambda_{\max}/\|g\|_2)$
  $v_{\text{guided}} \leftarrow v_{\text{base}} - t\, g$
  $h_{t-\Delta t} \leftarrow h_t - \Delta t\, v_{\text{guided}}$
end for
$V^e \leftarrow \mathcal{D}_{\text{vae}}(h_0)$
return V^e

E Edit Descriptor Feature Injection Modules: Per-Branch Details

This appendix expands the compressed description in Section 4.2 ("Delta injection modules") into the full per-branch design, with the rationale for each branch and the layer-selection policies that we sweep in the ablations.

Shared MLP stem. Before any injection, every injector first normalizes and adapts the latent code through a shared MLP stem,

$$\bar{d} \;=\; \mathrm{MLP}\big(\mathrm{LN}(d)\big) + d. \tag{26}$$

The shared stem isolates the basis change between the Stage I latent space (1024-dimensional, optimized for image-pair reconstruction) and the per-block residual stream of the video DiT (4096–

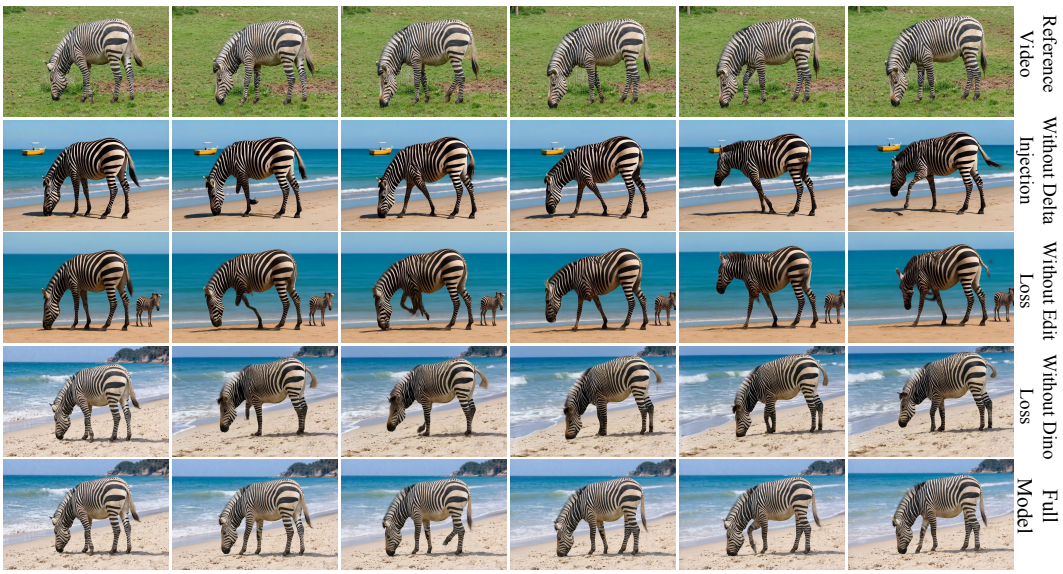

Figure 9: **Qualitative ablation.** Removing individual branches or alignment losses weakens either edit fidelity or temporal preservation, matching the quantitative trends in Section 5.3.

6144-dimensional depending on the block, optimized for spatiotemporal denoising). Sharing this stem across branches keeps the trainable parameter count small and acts as a soft regularizer that prevents each branch from learning a private, conflicting interpretation of d.

Branch 1: cross-attention. The cross-attention branch lets the hidden sequence $h \in \mathbb{R}^{T\times D}$ attend to the adapted code $\bar{d}$ as a global edit token,

$$h_{\text{cross}} \;=\; h \;+\; \text{CrossAttn}\big(Q(h),\, K(\bar{d}),\, V(\bar{d})\big). \tag{27}$$

Because the spatial query side decides where to attend on a per-token basis, this branch is the natural fit for *spatially selective* edits (object-local color changes, identity swaps in a face region, accessory addition). It also inherits the spatial re-localization property of the Stage I attention: the same $\bar{d}$ produces a different attention mask on each frame, so the edit follows the moving content.

Branch 2: FiLM. The FiLM branch [Perez et al., 2018] uses $\bar{d}$ to predict per-channel scale and shift parameters that modulate h globally,

$$h_{\text{film}} \;=\; \gamma(\bar{d}) \odot h \;+\; \beta(\bar{d}), \qquad \gamma, \beta : \mathbb{R}^{|d|} \to \mathbb{R}^{D}. \tag{28}$$

FiLM has no spatial selectivity (the same γ, β are applied at every token), which is precisely what we want for *global appearance transforms* such as global color grade, time-of-day shift, or stylization. We initialize γ to one and β to zero, so the FiLM branch is identity at the start of training.

Branch 3: additive. The additive branch projects $\bar{d}$ into a small set of L_a learned token slots and fuses them into the residual stream through a low-rank residual path,

$$h_{\text{add}} \;=\; h \;+\; W_{\text{up}}\, \sigma\big(W_{\text{down}}[\,h\,;\,\bar{d}\,]\big), \tag{29}$$

with $W_{\text{down}} \in \mathbb{R}^{r\times(D+|d|)}$, $W_{\text{up}} \in \mathbb{R}^{D\times r}$, $r \ll D$, and σ a SiLU activation. This is the most expressive branch but also the most prone to overfitting; it acts as a learned low-rank edit bias and is the most useful one when the demonstrated edit cannot be cleanly described as either a spatial overlay (cross-attention) or a global recolor (FiLM).

Gated fusion. Branch outputs are combined by a learned, scalar-gated projection,

$$h_{\text{out}} \;=\; h \;+\; \tanh(\xi) \cdot \big[\, h_{\text{cross}} - h \;+\; h_{\text{film}} - h \;+\; h_{\text{add}} - h \,\big], \tag{30}$$

with ξ a learnable scalar initialized to zero. At initialization $\tanh(\xi) = 0$, so $h_{\text{out}} = h$ and the wrapped backbone is bit-identical to the pre-trained Wan2.1-I2V-14B; the gate then opens monotonically during training. This is the standard zero-init trick [Zhang et al., 2023] that makes fine-tuning of large pre-trained generators stable.

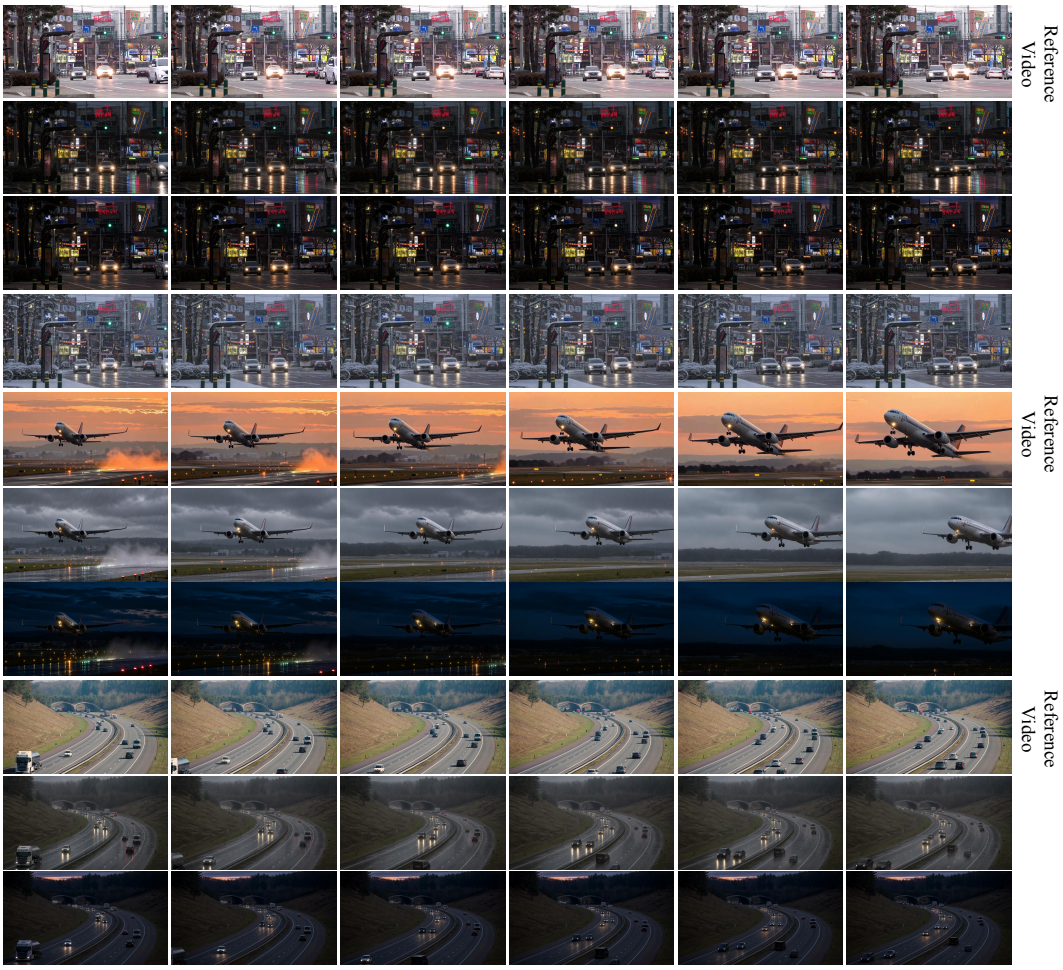

Figure 10: **More results in Traffic environment.**

F Human Preference Evaluation

We conducted a human preference study with 20 participants and 20 videos spanning relighting, stylization, object addition/removal, and restoration-type edits. Outputs from the compared methods were anonymized and randomly shuffled. Participants rated each result on a 1–5 scale for edit correctness, content preservation, temporal consistency, visual quality, and motion naturalness. Table 5 reports the mean scores under the corresponding category labels. These are subjective ratings, distinct from the automatic metrics reported in Section 5.

Table 5: **Human preference evaluation.** Mean ratings on a 1–5 scale; higher is better. No confidence intervals or significance tests are reported, so differences are descriptive.

Method	Edit correctness	Edit preservation	Temporal consistency	Aesthetic quality	Dynamic degree
HunyuanCustom[Hu et al., 2025]	3.91	3.27	4.02	3.82	2.92
VACE[Jiang et al., 2025]	3.73	2.85	3.87	3.62	3.46
UniVideo[Wei et al., 2025]	3.84	3.52	4.15	4.11	3.29
EDITDISTILL	**4.12**	**3.86**	**4.21**	**4.19**	**3.57**

EDITDISTILL receives the highest mean rating in each category in this study. The aggregate scores do not establish statistical significance or characterize participant-level variation.

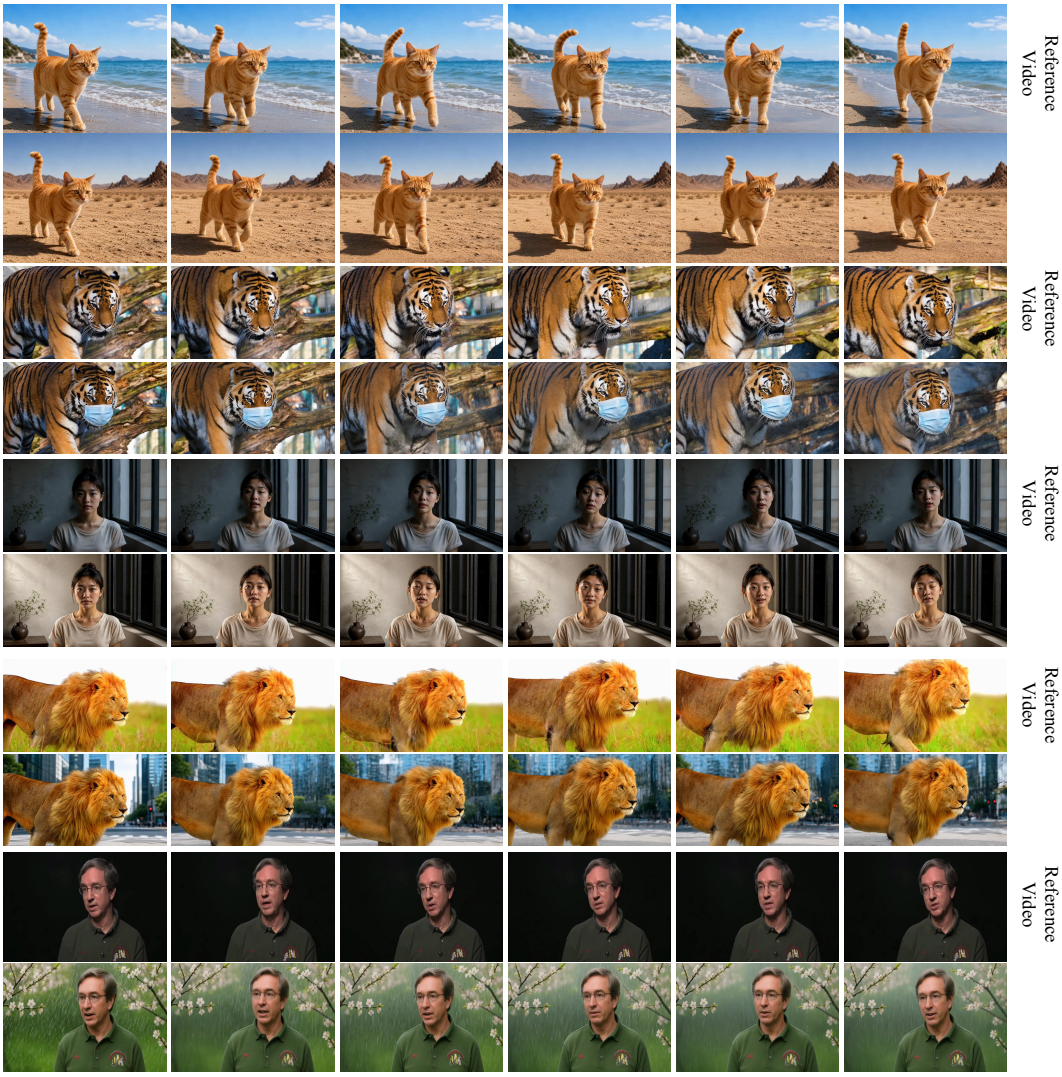

Figure 11: **Various video Editing.**

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 provide a clear and consistent description of the paper's contributions and scope (refer to sec. 1).

   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 of the proposed method are discussed in the appendix (see sec. A), including that local edits may affect surrounding background regions and that the method is less effective under large motion.

   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: [N/A]

Justification: The paper is a systems and methodological work and does not focus on formal theoretical analysis. Its effectiveness is instead demonstrated through experiments (see sec. 5) and observations (fig. 2) on benchmark datasets.

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 provides experimental results in the experiments section, along with details of the method and evaluation protocol to support the main claims. See sec 4 and sec. 5.

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: Public access to the code and data is not included in the current anonymized version. A full release will be provided in the final submission. In addition, we provide an anonymous webpage showcasing the video results presented in the paper.

   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 provides the model architecture, training setup, and evaluation protocol, including key implementation details and hyperparameters necessary to understand the results.

   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: [Yes]

   Justification: The paper reports quantitative results with appropriate evaluation metrics, and statistical variation (e.g., across different scenes) is reflected in the reported performance.

   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 experimental setup section and the appendix specify the computing environment, including hardware configuration and other implementation details.

   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]

   The work complies with the NeurIPS Code of Ethics. It does not involve human-subject experiments and includes a discussion of potential risks and responsible use. See sec. A.

   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 includes a broader impact section that discusses both positive applications (e.g., creative and assistive uses) and potential risks (e.g., deception, unauthorized manipulation, and misuse). See sec. A.

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: [N/A]

    Justification: The paper does not release any high-risk models or datasets. Potential misuse risks and possible safeguards (e.g., provenance tracking and watermarking) are discussed in the broader impact section.

    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: [No]

    Justification: The paper cites the Wan backbone used in our pipeline and the datasets (not original assets of ours) used for training. However, explicit license details and terms of use for these assets are not fully specified in the current version.

    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: [N/A]

    The paper does not release new assets in the current submission. The proposed model and implementation are described in the paper.

    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: [N/A]

    Justification: The submission does not include crowdsourcing or human-subject studies.

    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: [N/A]

    Justification: The work does not involve human-subject data collection or intervention studies requiring IRB review.

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 proposed method does not use LLMs as part of the core model, training procedure, or evaluation pipeline.

    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.