BEYOND THE FINAL REWARD: DEPTH-AWARE POST-TRAINING FOR INSTRUCTIONAL VIDEO EDITING

Anonymous authors
Paper under double-blind review

ABSTRACT

Post-training with reinforcement learning is the standard route to aligning instruction-guided video editing models with human preferences. Current pipelines realize it by rolling out a full denoising trajectory and scoring only the final decoded video, typically with several specialized reward models whose scores are combined by hand. Such outcome-only supervision only observes the final output of each diffusion-transformer evaluation, treating the evaluation as a single atomic decision. In reality, nevertheless, the prediction is progressively assembled across dozens of instruction-conditioned blocks. Thus, this supervision cannot reveal at which depth editing quality is formed, nor which part of the instruction is responsible for it. To address this issue, we introduce a post-training method for **V**ideo editing through **I**ntermediate **S**elf-evaluation and **T**oken **A**ttribution (**VISTA**). At selected transformer layers, VISTA first decodes groups of intermediate edit candidates, and evaluates them with a single reward model. A return-to-go objective then assigns each layer credit using its own reward and those observed at subsequent layers. The winning candidate is distilled into a token-importance memory that conditions later layers and re-weights a layer-wise GRPO update. By doing this, VISTA converts outcome-only GRPO into depth-aware self-evaluation without composing multiple specialized reward models. We further bound the bias caused by reusing the token-importance memory within a layer update, and give a sufficient condition under which this bias is dominated by genuine descent on the training objective. Experiments on 5B and 14B video-editing backbones demonstrate consistent improvements on OpenVE-Bench and FiVE-Bench, while ablations verify the complementary effects of intermediate self-evaluation and token attribution. Video samples are available at `https://videoediting-rl.github.io`.

1 INTRODUCTION

Instruction-guided video editing, producing a version of a source video that faithfully reflects a natural-language editing instruction while preserving the untargeted content, has advanced rapidly in recent years (Wu et al., 2025; Bai et al., 2025; Zhang et al., 2026c; Lin et al., 2026b). Driven by large-scale pretraining of video diffusion transformers on web-scale video–text corpora (Zi et al., 2026; He et al., 2025), modern editing backbones can now handle a broad spectrum of instructions while maintaining strong temporal consistency and visual fidelity (Li et al., 2025a; Ju et al., 2025). This pretraining stage endows the models with rich generative priors and general-purpose editing competence, and has been the primary source of the field's recent progress. However, pretraining alone is not sufficient: a pretrained editing model captures *plausible* edits but is not explicitly optimized for *instruction adherence* or *human-perceived quality*, and consequently still suffers from instruction misalignment, over-editing of untargeted regions, and temporal artifacts (Cong et al., 2025; Liu et al., 2026b). Closing this gap requires a post-training stage, and reinforcement learning (RL) fine-tuning, most notably Group Relative Policy Optimization (GRPO) (Shao et al., 2024), has emerged as the standard tool, using multiple reward models to align the backbone with human preferences along instruction alignment, temporal consistency, and visual fidelity (Park et al., 2026; Liu et al., 2026a; Xue et al., 2025).

Yet, reinforcement-learning-based post-training for video editing remains considerably underexplored compared with pretraining. Representative RL fine-tuning methods for diffusion models treat the denoising process as a multi-step trajectory but evaluate only the final generated output: a complete denoising trajectory is sampled, and a scalar reward is assigned to the decoded image or video (Su et al., 2024; Fan et al., 2023; Xue et al., 2025). This outcome-only supervision has two key limitations. First, it is *sparse*: the reward indicates whether the final edit is successful but provides no direct information about which intermediate decisions contributed to the outcome (Hu et al., 2025). Such delayed feedback makes temporal credit assignment difficult, and provides limited guidance for improving the model's editing behavior (Arjona-Medina et al., 2019). Second, it is *coarse*: although the diffusion backbone produces a rich sequence of intermediate representations, these representations are not explicitly evaluated or leveraged as training signals (Liang et al., 2025; Zhang et al., 2026d; Liao et al., 2025a). Together, these limitations can make RL fine-tuning

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

for video editing sample-inefficient and offer little insight into why an edit succeeds or fails, which is particularly problematic given the computational cost of generating and evaluating complete video rollouts (Xue et al., 2025).

We begin from the observation that the backbone itself is able to provide a denser source of supervision. Within a single rollout, a diffusion transformer (DiT) (Peebles & Xie, 2023) progressively transforms the visual representation under instruction conditioning in its successive blocks. Figure 1 traces one forward pass through the backbone, decoding each block's hidden state into an intermediate edit and measuring both its reward and how concentrated its dependence on the instruction is. This probe reveals two complementary forms of structure along backbone depth. First, the decoded states are already recognizable edits, and their rewards increase progressively with depth. They can therefore support meaningful layer-wise evaluation, consistent with evidence that intermediate diffusion features encode preferences over the eventual sample (Wu et al., 2026). Second, the instruction is read selectively but differently at each depth: attributing every block's prediction to individual instruction words yields an importance distribution whose importance concentration (1 when one word carries all of it, 0 when it is spread uniformly (Appendix C)) stays high throughout, while the words it singles out change from block to block.

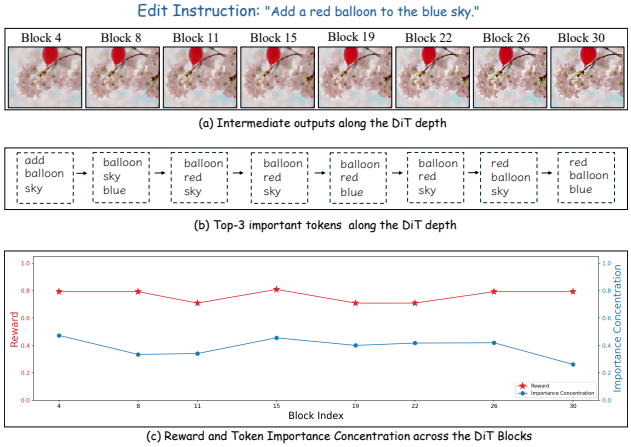


Figure 1: **(a)** The decoded intermediate edits: already recognizable and scoreable. **(b)** The three words carrying the most importance at each block. **(c)** Reward and importance concentration against block index, where the concentration summarizes how peaked the importance in (b) is across the whole instruction (Appendix C).

Building on these observations, we propose a post-training method for **V**ideo editing through **I**ntermediate **S**elf-evaluation and **T**oken **A**ttribution (**VISTA**). At a small set of selected layers, VISTA decodes a group of candidate intermediate edits, self-evaluates them with a reward model, and assigns each layer a return-to-go (RTG) (Son et al., 2025) that incorporates the rewards obtained at subsequent layers. It then distills the winning candidate into a token-importance signal that conditions the remaining layers and reweights a layer-wise GRPO update. This turns group-relative policy optimization from a terminal-reward objective into a depth-aware one that follows the instruction tokens actually driving reward. Because the learning signal comes from evaluating the model's own intermediate outputs, one reward model suffices, recasting RL fine-tuning as *self-enhancement* rather than reward engineering (Yuan et al., 2024; He et al., 2026a) and eliminating specialized reward ensembles and their hand-tuned weighting.

Our contributions are summarized shortly as follows:

- We reformulate RL fine-tuning for instruction-guided video editing as a *sequential self-evaluation process*, in which the diffusion transformer rollout exposes and rewards its own intermediate outputs rather than relying solely on a terminal reward. We consolidate this idea in VISTA, a reinforcement learning framework that turns a standard rollout into a closed layer-wise refinement loop.
- We instantiate the framework with two key components: (i) *intermediate self-evaluation* with return-to-go credit assignment across selected transformer layers; and (ii) an *additive token-importance memory* that distills each self-evaluated winner into a signal that steers and reweights a layer-wise GRPO update.
- We validate VISTA on 5B and 14B video editing backbones, showing consistent gains over outcome-only RL fine-tuning on OpenVE-Bench and FiVE-Bench against baselines of comparable scale, with ablations that isolate the contribution of each component.

2 Related Work

Instruction-guided video editing. Instruction-driven editing was first established in the image domain, either by manipulating cross-attention maps of a pretrained diffusion model (Hertz et al., 2022) or by supervised training on synthetically generated instruction–image pairs (Brooks et al., 2023). Extending this to video adds a temporal-coherence requirement, and the first generation of video methods met it without task-specific supervision: by one-shot tuning on the source video (Wu et al., 2023a), by fusing or controlling attention during inversion to preserve unedited structure (Qi et al., 2023; Liu et al., 2024; Ceylan et al., 2023), by propagating the features of a single edited keyframe across frames (Geyer et al., 2024; Ku et al., 2024; Yang et al., 2023), or by merging tokens and structural controls

across frames (Li et al., 2024; Zhang et al., 2024). These training-free adaptations have since been largely superseded. Riding on large-scale video diffusion transformers (Peebles & Xie, 2023; Blattmann et al., 2023; Yang et al., 2025; Kong et al., 2024; Wan et al., 2025), the field has converged on feed-forward editors trained on million-scale instruction–video corpora synthesized by expert pipelines and filtered by vision–language models (Cheng et al., 2024; Zi et al., 2026; Wu et al., 2025; Zhang et al., 2025a; Bai et al., 2025; He et al., 2025; Liang et al., 2026b). Around this data-centric core, three directions have emerged. In-context formulations unify image and video editing, or dispense with paired clips altogether, by conditioning the generator on interleaved exemplars (Ju et al., 2025; Liao et al., 2025b; Zhang et al., 2025b). Architectural work targets the failure modes that persist after supervised training, disentangling semantic anchoring from motion alignment (Zhang et al., 2026c), transferring image-manipulation priors to video (Wang et al., 2026), or extending instructions with reference images (Lin et al., 2026b; Mai et al., 2026). A third line makes the editor *deliberate* before it generates, through explicit chain-of-thought planning (Liang et al., 2026a), implicit reasoning inside the diffusion transformer (Li et al., 2026), self-reflective revision in unified models (Liu et al., 2025), or reasoning over digital-twin representations with RL (Shen et al., 2026). Progress is tracked by benchmarks that score instruction adherence and content preservation at increasingly fine granularity (Li et al., 2025a; Chen et al., 2026b; Miao et al., 2026). However, most of these models are fit with likelihood-based objectives on (pseudo-)paired data, and are therefore never explicitly trained to satisfy instruction adherence or human-perceived quality.

Self-evaluation and self-enhancement. In language modeling this appears as verbal self-feedback and revision (Madaan et al., 2023; Shinn et al., 2023), deliberate search over intermediate states (Yao et al., 2023), a model acting as its own judge and reward source (Zheng et al., 2023; Yuan et al., 2024), or an internalized critic folded into the RL objective so that critique and generation are learned jointly (Lin et al., 2026a). The same idea is now reaching visual generation: diffusion alignment can be bootstrapped from the model's own preferred samples with minimal human feedback (He et al., 2026a; Tewel et al., 2026), self-reward RL and self-distillation turn out to be two views of sharpening a model's own output distribution (Tomihari & Sato, 2026), and, most relevant to us, the intermediate states of a video diffusion model already carry preference information about the sample that will eventually be produced (Wu et al., 2026). Video editing itself has begun to adopt the pattern, with unified models that critique and revise a completed edit before returning it (Liu et al., 2025).

VISTA pushes this observation inward. Where prior self-improvement loops operate over discrete text tokens or across *completed* generations, ours runs inside a single forward pass of a video diffusion transformer: the backbone decodes candidates at intermediate layers, self-evaluates them with one reward model, records why the winner succeeded as a token-importance mask, and reinforces that behavior at the next layer.

Due to space constraints, related work from other perspectives relevant to VISTA is provided in Appendix A.

3 METHOD

In this section, we present VISTA, a depth-wise self-evaluation framework that extends terminal supervision to selected transformer layers (Fig. 2). At each selected layer, the model decodes and evaluates a group of intermediate edits; a return-to-go assigns credit across subsequent layers, and the winning edit is distilled into a token-importance memory that guides the remaining computation and the layer-wise GRPO update. We first formulate the layer-wise rollout in Sec. 3.1 and establish its optimization guarantee in Appendix. B. We then detail the intermediate self-evaluation and memory-guided policy-refinement components in Secs. 3.2 and 3.3, respectively.

3.1 PROBLEM FORMULATION

Given a source video $\mathbf{V}^{\text{src}} \in \mathbb{R}^{F\times H\times W\times 3}$ and a textual editing instruction $\mathbf{c} = (c_1, \ldots, c_N) \in \mathcal{V}^N$, *i.e.*, a sequence of N tokens drawn from a vocabulary $\mathcal{V}$, an instruction-guided editing model is a mapping $f_\theta : (\mathbf{V}^{\text{src}}, \mathbf{c}) \mapsto \mathbf{V}^{\text{edit}}$ parameterized by θ, which should faithfully reflect the semantics of $\mathbf{c}$ while preserving the content of $\mathbf{V}^{\text{src}}$ that is not targeted by the instruction. We instantiate f_θ as an L-layer transformer-based diffusion backbone Φ_θ with hidden states $\mathcal{H}^{(\ell)} = \Phi_\theta^{(\ell)}(\mathcal{H}^{(\ell-1)}, \mathbf{c})$. Rather than optimizing a single terminal reward, we select a subset of self-evaluation layers $\mathcal{L} = \{\ell_1, \ell_2, \ldots, \ell_K\} \subseteq \{1, \ldots, L\}$, and at each ℓ_k a lightweight decoding head g_ϕ maps the hidden state to an intermediate output

$$\hat{\mathbf{V}}^{(\ell_k)} = g_\phi\Big(\mathcal{H}^{(\ell_k)}\Big), \tag{1}$$

where the model can evaluate before generation completes. Collecting these outputs across all selected layers yields the self-evaluation trajectory $\tau_\theta(\mathbf{V}^{\text{src}}, \mathbf{c}) = (\hat{\mathbf{V}}^{(\ell_1)}, \ldots, \hat{\mathbf{V}}^{(\ell_K)})$, the ordered sequence of intermediate outputs produced by f_θ at the layers in $\mathcal{L}$.

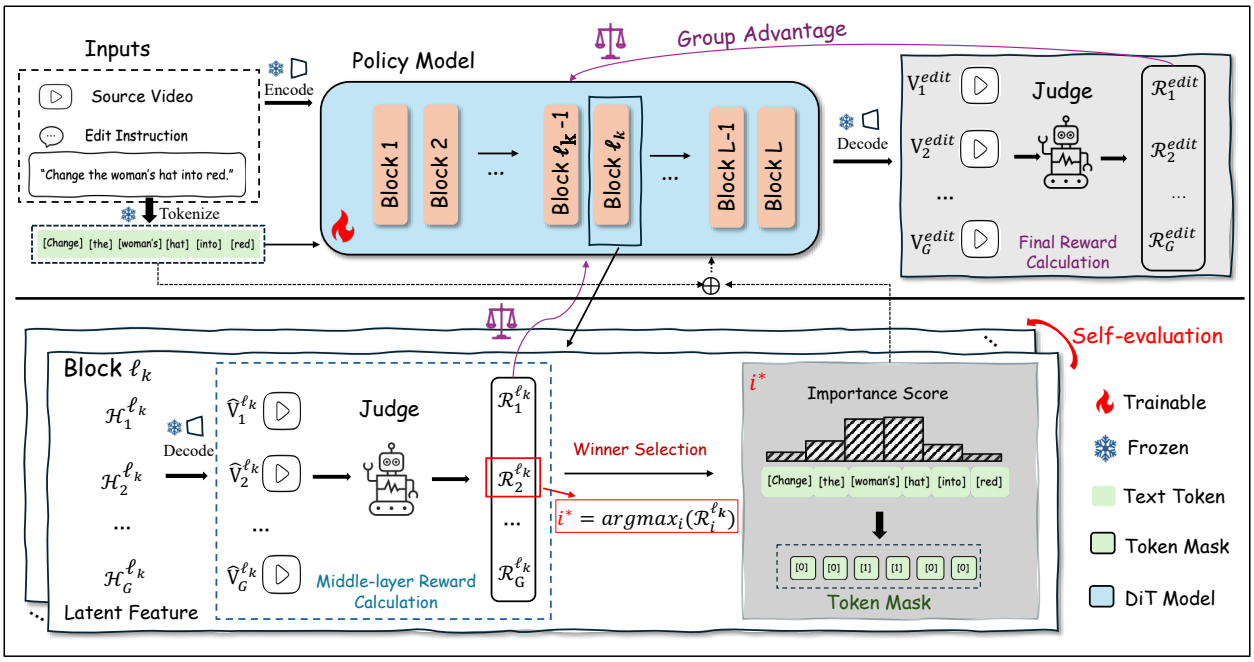


Figure 2: Overview of the proposed VISTA framework for video editing. At selected transformer layers, the model decodes intermediate outputs and *self-evaluates* them; the best self-evaluated output is then distilled into a token-importance memory that drives a GRPO update of the later layers, forming a closed self-evaluation refinement loop.

Let $R(\cdot \mid \mathbf{V}^{\text{src}}, \mathbf{c})$ be a reward model scoring editing quality along instruction alignment, temporal consistency, and visual fidelity. The standard outcome-driven objective $\mathcal{J}_{\text{outcome}}(\theta) = \mathbb{E}_{(\mathbf{V}^{\text{src}},\mathbf{c})}[R(\mathbf{V}^{\text{edit}} \mid \mathbf{V}^{\text{src}}, \mathbf{c})]$ supervises only the final output. VISTA instead updates the model once at each selected layer ℓ_k, and the objective at that layer is a *return-to-go* that sees only the rewards from ℓ_k onward (Sec. 3.2):

$$\mathcal{J}^{(k)}_{\text{VISTA}}(\theta) \;=\; \mathbb{E}_{(\mathbf{V}^{\text{src}},\mathbf{c})}\left[\sum_{j=k}^{K} \gamma\, R\Big(\hat{\mathbf{V}}^{(\ell_j)} \,\Big|\, \mathbf{V}^{\text{src}}, \mathbf{c}\Big)\right], \tag{2}$$

where $\gamma \in [0,1]$ is a discount. The objective is along depth: $\mathcal{J}^{(k)}_{\text{VISTA}}$ doesn't credit layers before ℓ_k. After the update at ℓ_k, a token-importance mask $\mathbf{m}(\mathbf{c}) \in [0,1]^N$ is distilled from the best self-evaluated output and conditions the remaining layers (Sec. 3.3).

3.2 Token-Importance Memory Distillation

This component instantiates the self-evaluation stage of VISTA: it lifts the otherwise opaque forward pass into a set of observable intermediate outputs that the model can score and compare against one another. Following the self-evaluation layers $\mathcal{L} = \{\ell_1, \ldots, \ell_K\}$ defined in Sec. 3.1, we treat each ℓ_k as a *self-evaluation checkpoint* at which the model decodes an intermediate output. Applying the decoding head g_ϕ from Eq. (1) to a group of G stochastic samples yields, at each checkpoint, a group of candidate intermediate outputs

$$\mathcal{G}^{(\ell_k)} = \left\{\hat{\mathbf{V}}_g^{(\ell_k)}\right\}_{g=1}^{G}, \tag{3}$$

which collectively form the output groups $\{\mathcal{G}^{(\ell_k)}\}_{k=1}^{K}$. Exposing intermediate hidden states as observable videos lifts the vanilla generation pipeline into a well-defined space over which self-evaluation and comparison become tractable.

Each candidate $\hat{\mathbf{V}}_g^{(\ell_k)}$ is self-evaluated by the reward model $R(\cdot \mid \mathbf{V}^{\text{src}}, \mathbf{c})$ introduced in Sec. 3.1, which measures editing quality along instruction alignment, temporal consistency, and visual fidelity. Because an earlier self-evaluation checkpoint shapes all subsequent generation, its quality should be judged not only by its immediate reward but also by the rewards it enables at later checkpoints. We therefore adopt a *return-to-go* formulation that credits each layer with

a discounted sum of the rewards attained from that layer onward,

$$\mathcal{R}_g^{(\ell_k)} \;=\; \sum_{j=k}^{K} \gamma\, R\Big(\hat{\mathbf{V}}_g^{(\ell_j)} \,\Big|\, \mathbf{V}^{\text{src}}, \mathbf{c}\Big), \tag{4}$$

where $\gamma \in [0,1]$ is a discount factor controlling how strongly later checkpoints' rewards propagate back to earlier layers: $\gamma = 0$ recovers the purely local reward, whereas $\gamma \to 1$ shares credit almost equally across all downstream checkpoints. Within each group we then identify the *best-evaluated output* as the candidate:

$$\hat{\mathbf{V}}_\star^{(\ell_k)} \;=\; \arg\max_{g \in \{1,\dots,G\}} \mathcal{R}_g^{(\ell_k)}, \tag{5}$$

i.e., the self-evaluated winner representing the most successful editing behavior observed at stage ℓ_k once its downstream effect is accounted for.

Given the self-evaluated winners discovered in Eq. 5, it (i) distills each winner into a low-dimensional token-level statistic that captures the cause of its success, and (ii) uses that statistic to bias a group-relative policy update. Conceptually, this replaces scalar-reward reinforcement with a structured, causally-informed policy improvement operator. To distill the winner into a memory that can guide later layers, let $\mathbf{c}_n \in \mathbb{R}^d$ be the context embedding of the n-th instruction token and $\mathcal{L}_\star^{(\ell_k)}$ the term of Eq. (11) contributed by the winner $\hat{\mathbf{V}}_\star^{(\ell_k)}$ of Eq. (5). We attribute this loss to each instruction position by grad$\times$input,

$$a_n^{(\ell_k)} \;=\; \Big|\Big\langle \nabla_{\mathbf{c}_n} \mathcal{L}_\star^{(\ell_k)},\; \mathbf{c}_n \Big\rangle\Big|, \qquad n = \{1,\dots,N\}, \tag{6}$$

where $\langle\cdot,\cdot\rangle$ is the inner product over the d embedding dimensions and the absolute value keeps tokens that decrease the loss as well as those that increase it, and normalize across the instruction to obtain the *token-importance memory*

$$m_n^{(\ell_k)} = \frac{a_n^{(\ell_k)}}{\max\Big(\sum_{n'=1}^{N} a_{n'}^{(\ell_k)},\; \epsilon_0\Big)}, \qquad \mathbf{m}^{(\ell_k)}(\mathbf{c}) = \big[m_1^{(\ell_k)}, \dots, m_N^{(\ell_k)}\big] \in [0,1]^N, \tag{7}$$

a distribution over the N instruction tokens, written $\mathbf{m}(\mathbf{c})$ when the layer is clear. The floor ϵ_0 guards against a vanishing attribution mass and is inactive otherwise. The memory is computed from detached quantities, hence constant in the update it weights, and enters the backbone through Eqs. (8) and (10).

3.3 Self-Evaluation-Guided Policy Refinement

Having distilled the self-evaluated winner into a token importance mask, we now use this structured signal to refine the policy. The mask plays two complementary roles: it propagates information about successful editing behavior to subsequent transformer layers, and it reshapes the conditioning of the layer-wise GRPO update so that the visual policy is scored under the instruction tokens most responsible for that behavior. The $\mathbf{m}(\mathbf{c})$ is injected into the subsequent transformer layers in an additive manner: the mask-weighted instruction tokens are added back to the original token representations, so that the conditioning of later layers becomes $\mathbf{c} + \mathbf{m}(\mathbf{c}) \odot \mathbf{c}$,

$$\tilde{\mathcal{H}}^{(\ell)} \;=\; \Phi_\theta^{(\ell)}\Big(\mathcal{H}^{(\ell-1)},\; \mathbf{c} + \mathbf{m}(\mathbf{c}) \odot \mathbf{c}\Big), \qquad \ell > \ell_k, \tag{8}$$

which emphasizes the tokens distilled from earlier self-evaluated successes without discarding the original instruction. Unlike a scalar reward, this token-level statistic retains the structural information of why an editing strategy succeeded, enabling fine-grained credit transfer across stages.

Finally, we optimize θ by performing a model update at each self-evaluation layer. At layer ℓ_k, for each candidate $\hat{\mathbf{V}}_g^{(\ell_k)}$ in the group $\mathcal{G}^{(\ell_k)}$, we compute a group-relative advantage over the return-to-go of Eq. (4),

$$A_g^{(\ell_k)} \;=\; \frac{\mathcal{R}_g^{(\ell_k)} - \mu^{(\ell_k)}}{\sigma^{(\ell_k)}}, \qquad \mu^{(\ell_k)} = \tfrac{1}{G}\sum_g \mathcal{R}_g^{(\ell_k)}, \tag{9}$$

with $\sigma^{(\ell_k)}$ the group standard deviation of $\{\mathcal{R}_g^{(\ell_k)}\}_{g=1}^{G}$.

Since the diffusion backbone generates each candidate through a T-step denoising rollout with latents $\{\mathbf{z}_{t,g}^{(\ell_k)}\}_{t=0}^{T}$, the policy factorizes over the sampling steps, and we evaluate its log-probability per step. Under the additively-augmented



Figure 3: **Qualitative comparison on instruction-guided video editing.** Each block shows frames sampled from a source video together with its editing instruction, followed by the results of the size-matched baselines and our method. All methods share identical source videos, instructions, and sampling settings.

conditioning of Eq. (8), the current policy assigns each transition the probability $\pi_\theta(\mathbf{z}_{t-1,g}^{(\ell_k)} \mid \mathbf{z}_{t,g}^{(\ell_k)}, \mathbf{c} + \mathbf{m}(\mathbf{c}) \odot \mathbf{c})$, giving the per-step importance ratio

$$\rho_{g,t}^{(\ell_k)} = \frac{\pi_\theta\left(\mathbf{z}_{t-1,g}^{(\ell_k)} \mid \mathbf{z}_{t,g}^{(\ell_k)}, \mathbf{c} + \mathbf{m}(\mathbf{c}) \odot \mathbf{c}\right)}{\pi_{\theta_{\text{old}}}\left(\mathbf{z}_{t-1,g}^{(\ell_k)} \mid \mathbf{z}_{t,g}^{(\ell_k)}, \mathbf{c}\right)}, \tag{10}$$

in which the added mask changes the current policy's velocity prediction, and therefore its visual-step log-probability, at every sampling step. The per-step GRPO loss at layer ℓ_k is

$$\mathcal{L}^{(\ell_k)}(\theta) = -\mathbb{E}\left[\frac{1}{GT}\sum_{g=1}^{G}\sum_{t=1}^{T}\min\left(\rho_{g,t}^{(\ell_k)} A_g^{(\ell_k)}, \text{clip}\left(\rho_{g,t}^{(\ell_k)}, 1-\epsilon, 1+\epsilon\right) A_g^{(\ell_k)}\right)\right] + \beta\,\text{KL}(\pi_\theta \,\|\, \pi_{\text{ref}}), \tag{11}$$

where ϵ is the clipping threshold and β controls the KL regularization against a reference policy π_{ref}. Note that the return-to-go advantage $A_g^{(\ell_k)}$ is a single scalar shared across all sampling steps of candidate g.

The resulting update cycle proceeds as follows. At checkpoint ℓ_k, the model scores the candidates decoded from the rollout (Eqs. 3 and 5), distills the winner into the token-importance memory $\mathbf{m}(\mathbf{c})$ (Eqs. 6–8), and performs a single gradient step on $\mathcal{L}^{(\ell_k)}$ under that memory; the memory is subsequently propagated to ℓ_{k+1}, where the same operations are repeated. Two properties bound the cost of this procedure: the intermediate outputs $\{\hat{\mathbf{V}}^{(\ell_j)}\}_{j=1}^{K}$ are obtained from a single rollout of $\pi_{\theta_{\text{old}}}$ and reused at every checkpoint, and $\mathbf{m}(\mathbf{c})$ is the only quantity propagated across depth, re-scoring this fixed trajectory through the conditioning of Eq. (10) rather than requiring additional generation. A single

rollout therefore yields K supervision signals at distinct depths, in place of the single terminal reward employed by outcome-driven RL. Appendix B bounds the error this reuse introduces and gives the condition under which each update still improves the fresh-rollout objective.

Trajectory surrogate. Evaluating the per-step ratio in Eq. (10) over the full T-step denoising trajectory is computationally expensive, as it requires a policy forward pass at every sampling step and for every candidate in the group. To make the per-layer update tractable, we utilize *LeapAlign* (Liang et al., 2026c), which replaces the entire T-step sampling trajectory of Eq. (11) with a compact two-step surrogate. We defer the detailed formulation and implementation to Appendix D.

Table 1: **Comparison on the OpenVE-Bench video-editing benchmark, judged by GEMINI-2.5-PRO and QWEN3.5-397B-A17B.** We report the average score across all edit categories (**Avg. Score**) together with per-category scores along three complementary axes: Instruction Compliance (IC), Consistency & Detail Fidelity (CF), and Visual Quality & Stability (VQ). All scores are on a 1–5 scale; higher is better. Our method attains the highest average score while using substantially fewer parameters than competing baselines.

Method	#Params	Avg. Score	Local Remove			Global Style			Background Change			Local Change			Local Add		
			IC	CF	VQ	IC	CF	VQ	IC	CF	VQ	IC	CF	VQ	IC	CF	VQ
GEMINI-2.5-PRO																	
Small-scale models (≤5B)																	
Omni-Video	1.3B	1.59	1.37	1.10	1.20	2.02	1.79	1.91	1.42	1.36	1.36	2.09	1.65	1.58	1.81	1.42	1.70
Lucy-Edit	5B	2.13	1.17	1.08	1.10	1.62	1.57	1.59	2.71	2.24	2.20	3.85	3.52	3.40	2.37	1.49	2.01
Kiwi-Edit	5B	3.16	3.31	2.66	2.73	3.74	3.52	3.64	3.02	2.76	**2.63**	4.06	3.80	3.77	2.93	2.22	2.64
Large-scale models (≥13B)																	
Ditto	14B	1.80	1.00	1.00	1.00	3.83	3.21	3.71	1.59	1.54	1.49	1.58	1.49	1.46	1.48	1.22	1.39
UniVideo	20B	1.39	1.14	1.02	1.08	2.31	1.84	2.28	1.36	1.31	1.24	1.35	1.18	1.15	1.25	1.12	1.21
VINO	13B	2.48	2.31	1.98	2.10	3.62	3.48	3.57	2.07	1.73	1.53	3.11	2.85	2.74	2.42	1.61	2.13
SAMA	14B	3.27	**3.61**	2.76	2.92	**3.91**	**3.71**	3.83	3.03	**2.81**	2.46	4.28	4.02	3.98	2.93	2.18	2.66
Ours	5B	**3.31**	3.59	**3.05**	**3.10**	3.64	3.52	3.57	**3.12**	2.71	2.51	**4.40**	**4.14**	**4.11**	**3.06**	**2.40**	**2.81**
QWEN3.5-397B-A17B																	
Small-scale models (≤5B)																	
Omni-Video	1.3B	2.70	1.42	1.56	1.63	2.81	3.28	3.86	2.88	2.76	2.93	2.72	2.92	3.09	2.39	2.81	3.37
Lucy-Edit	5B	3.20	1.05	1.90	1.90	1.62	4.19	4.26	3.97	3.69	3.76	4.11	4.25	4.23	2.51	2.96	3.58
Kiwi-Edit	5B	3.87	3.22	3.12	3.17	3.95	4.48	4.60	**4.14**	3.93	**4.00**	4.51	4.34	4.35	3.21	3.25	3.81
Large-scale models (≥13B)																	
Ditto	14B	2.71	1.07	1.38	1.48	3.79	3.78	4.41	3.45	3.39	3.45	2.24	2.39	2.71	2.00	2.28	2.81
UniVideo	20B	2.53	1.15	1.17	1.24	3.00	3.19	4.16	2.66	2.69	2.71	2.26	2.48	2.57	2.33	2.85	3.43
VINO	13B	3.39	2.48	2.58	2.62	3.97	4.16	4.55	3.34	3.41	3.33	3.48	3.67	3.69	2.61	3.15	3.82
SAMA	14B	3.99	3.64	**3.63**	**3.69**	3.97	**4.72**	**4.78**	4.08	**4.03**	3.98	4.46	**4.46**	4.45	2.82	3.22	3.91
Ours	5B	**4.05**	**3.85**	**3.63**	**3.69**	**4.02**	4.48	4.53	4.03	4.00	3.93	**4.55**	**4.46**	**4.48**	**3.43**	**3.60**	**4.04**

4 EXPERIMENTS

4.1 EXPERIMENTAL SETTINGS

Training Settings We evaluate VISTA at two scales, both post-trained on the same self-collected editing corpus. Rather than instruction-tuning from scratch, each model starts from a released instruction-tuned editing checkpoint: the 5B variant builds on WAN2.2-TI2V-5B initialized from Kiwi-Edit (Lin et al., 2026b) and is fully fine-tuned, whereas the 14B variant builds on WAN2.1-T2V-14B initialized from SAMA (Zhang et al., 2026c) and is adapted with LoRA for memory efficiency. Since both initializations also appear as baselines, the reported gains isolate the effect of our RL post-training relative to the exact starting policy. All experiments run on 16 NVIDIA H200 GPUs; the complete hyperparameter listing for both configurations is deferred to Appendix E.

Benchmarks We evaluate on two complementary benchmarks. **OpenVE-Bench** (431 video-edit pairs, eight editing categories) targets general instruction following; we use its local add, local remove, local replace, global style, and background change categories. The main text reports two independent judges, GEMINI-2.5-PRO and QWEN3.5-397B-A17B; the benchmark's official judges, GEMINI-3.1-PRO and QWEN3.5-35B-A3B, together with a cross-judge analysis over all four evaluators, are deferred to Appendix G.4. **FiVE-Bench** (100 source videos, 420 prompt pairs, six editing types) targets fine-grained, object-level accuracy and content preservation, scoring six reference-free metrics instead of a VLM judge: CLIP similarity to the target caption (CLIPS) and along the intended edit direction (CLIPS_e),

motion fidelity (MF), structural distance (Struct), and PSNR and LPIPS on the non-edited regions. Since these measure embedding and pixel agreement rather than instruction compliance, and can therefore favor methods that leave the source nearly unchanged, we treat them as a secondary check and defer the full results to Appendix G.3.

Baselines Since editing performance is strongly correlated with model scale, we group the baselines by capacity: three baselines with $\leq$5B parameters (Omni-Video (Yang et al., 2026), Lucy-Edit (Team, 2025), Kiwi-Edit (Lin et al., 2026b)) and four with $\geq$13B parameters (Ditto (Bai et al., 2025), UniVideo (Wei et al., 2025), VINO (Chen et al., 2026a), SAMA (Zhang et al., 2026c)). The main text reports our 5B model only, and it shows that VISTA at 5B already matches or surpasses baselines with two to four times as many parameters. Our 14B model, together with quantitative comparisons is reported in Appendix G.4 and Appendix G.3, the corresponding qualitative comparisons on both benchmarks and at both scales are given in Appendix F.

4.2 Comparisons with State-of-the-Art Methods

Quantitative comparison. Table 1 reports OpenVE-Bench scores under two independent judges. VISTA attains the best average under both, 3.31 with GEMINI-2.5-PRO and 4.05 with QWEN3.5-397B-A17B. The most informative margin is the one over Kiwi-Edit, the checkpoint our 5B model is initialized from: with backbone, training corpus, and instruction tuning all held fixed, it isolates the contribution of layer-wise self-evaluation. The same 5B model also edges out SAMA, the strongest $\geq$13B baseline, at roughly a third of its parameters. The two judges differ in absolute calibration but agree on the ordering at the top, so the ranking is not an artifact of a single evaluator. Scale alone does not determine editing quality, Ditto and UniVideo both fall below Lucy-Edit (5B), which supports our premise that the binding constraint in this regime is supervision signal density rather than model capacity.

Table 2: **Component ablation on OpenVE-Bench.** Results are reported on the 5B backbone, and using GEMINI-2.5-PRO (1–5; higher is better).

Training strategy	Avg.	IC	CF	VQ
Base model	3.16	3.41	2.99	3.08
+ outcome-only GRPO	2.98	3.24	2.83	2.87
w/o. Self-eval.	2.47	2.89	2.24	2.29
w/o. TI memory	1.43	1.55	1.33	1.40
VISTA (full)	**3.31**	**3.56**	**3.16**	**3.22**

Qualitative comparison. The category breakdown localizes where the gain originates. VISTA improves over SAMA on all three axes for Local Change and Local Add, and it lifts both preservation axes at essentially matched instruction compliance on Local Remove. That the improvement shows up mainly on the preservation axes is the expected signature of this mechanism, since a sharply concentrated mask should reduce over-editing of untargeted regions. Global Style is the sole exception, trailing SAMA under both judges; a global restyling instruction spreads relevance across the entire prompt, leaving the mask little structure to concentrate on. This points to a limitation of the token-importance memory rather than of the layer-wise reward. Figure 3 shows the same behavior visually: VISTA applies the requested change while leaving untargeted regions and the source motion largely intact, whereas the baselines more often under-apply the edit or alter content the instruction does not mention.



Figure 4: Visual comparison of the ablated variants and our full VISTA.

4.3 Ablation Study on VISTA

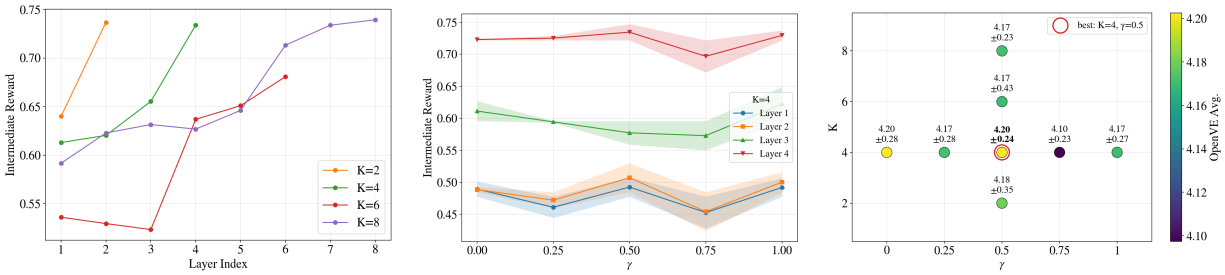
We ablate the two components of Secs. 3.2 and 3.3: *w/o.* token importance (TI) memory removes token-importance memory distillation, and *w/o.* Self-eval removes self-evaluation-guided policy refinement. The base model and outcome-only GRPO are included as controls. All variants share the same backbone, optimizer, data, inference protocol, and random seeds, and differ only in the training strategy. We report the 5B backbone with $G = 4$ and 300 steps, judged by GEMINI-2.5-PRO.

Outcome-only GRPO lowers the base score from 3.16 to 2.98 (Table 2). A single terminal reward is therefore not merely weak supervision for video editing but also actively harmful. Our layer-wise formulation is designed to address that failure mode, and the full method recovers this loss and improves all three axes over the base model instead. The two newly-minted components removal fail for different reasons. Without token-importance memory distillation

there are no intermediate scores, so the return-to-go of Eq. (4) degenerates to the terminal reward broadcast to every checkpoint, and the average falls to 2.47. Without self-evaluation, the collapse is far larger (1.43), which we attribute to the memory being the only quantity passed between checkpoints: in its absence the K updates are computed on one rollout under identical conditioning, so they reinforce the same signal repeatedly instead of acting on progressively refined evidence. Dense intermediate feedback and cross-layer propagation are thus not independent additions but two halves of one loop. Fig. 4 shows the corresponding outputs under identical inputs, which validates the discussions.

4.4 Method Analysis on VISTA

Figure 5 sweeps the number of self-evaluation checkpoints $K \in \{2, 4, 6, 8\}$, kept approximately uniform over depth, and the return-to-go discount $\gamma \in \{0, 0.25, 0.5, 0.75, 1.0\}$. The question is whether the gains come from adding more auxiliary predictions or from the depth-aware credit structure itself. OpenVE-Bench scores in this analysis are produced by Qwen3.6-35B-A3B, the reward model used during training.



(a) Intermediate rewards across the K selected layers with $\gamma = 0.5$.

(b) Per-checkpoint rewards under different γ with $K = 4$.

(c) Editing quality for different (K, γ) configurations.

Figure 5: **Sensitivity to self-evaluation density K and return-to-go discount γ. (a)** Intermediate reward as a function of checkpoint depth for $K \in \{2, 4, 6, 8\}$. **(b)** Intermediate reward at each of the four checkpoints as γ varies; shaded regions indicate variability. **(c)** OpenVE-Bench average score for the evaluated (K, γ) configurations, annotated with mean $\pm$ standard deviation. The red circle marks the selected setting, $K = 4, \gamma = 0.5$.

Across every configuration, rewards rise toward the final checkpoint (Figures 5a and 5b), confirming the premise that intermediate states are progressively better edits and that depth is therefore a meaningful axis for credit assignment. The two sweeps separate the two ways of exploiting that axis. Increasing K adds supervision without changing its structure, and it does not help: the terminal reward saturates at roughly the same value regardless of density, and $K = 6$ and $K = 8$ both score below $K = 4$ (Figure 5c) at a cost that grows approximately linearly in K (Appendix G.1). Varying γ changes the structure of the credit rather than its amount, and here the setting does matter: quality peaks at an intermediate discount, with $\gamma = 0.5$ matching $\gamma = 0$ on mean score at lower variability while larger discounts degrade. At $K = 4$, $\gamma = 0.5$ (our best selection) weights the next one or two checkpoints appreciably and the remainder negligibly, so the useful signal spans a short downstream window rather than the entire remaining stack.

5 Conclusions

We have presented VISTA, which replaces the terminal reward of standard RL post-training for instruction-guided video editing with supervision distributed along the depth of the diffusion transformer. VISTA treats the rollout as a sequential self-evaluation process over a small set of selected layers, extending group-relative policy optimization from a single terminal objective to per-layer objectives. The principle is causal along depth: each layer is credited by a return-to-go over the rewards from that layer onward. At each layer, the model decodes a set of intermediate edits, ranks them with a single reward model, and distills the winning candidate into a token-importance memory. This memory is injected additively into the conditioning, thereby shifting the log-density that the policy assigns to the same visual transition and steering the layer-wise update toward the instruction tokens that self-evaluation identifies as responsible for the successful edit. We further bound the combined approximation error incurred by reusing an old-policy rollout and a detached memory, and give a sufficient condition under which improvement of this surrogate transfers to the ideal fresh objective. Across 5B and 14B backbones, VISTA improves over its own initialization and over size-matched baselines on OpenVE-Bench and FiVE-Bench, with agreement across four independent judges. Ablations show that the layer-wise reward and the memory are two halves of one loop, and the hyperparameter analysis indicates that the benefit lies in how credit is routed across depth rather than in the amount of auxiliary supervision.

6 AI USE STATEMENT

We used generative AI tools to assist with language editing, improving readability, brainstorming, and refining the presentation of research ideas and experimental descriptions. AI tools were also occasionally used to discuss implementation details and provide feedback on writing and experimental design. All core research ideas, methodologies, implementations, experiments, analyses, and conclusions were developed and verified by the authors. Any AI-assisted content was carefully reviewed and validated by the authors before inclusion in the manuscript. The authors take full responsibility for the content of this work, including all technical claims, experimental results, and conclusions.

7 REPRODUCIBILITY STATEMENT

We provide the complete formulation of VISTA, including its layer-wise objective, token-importance memory, and LeapAlign surrogate. Appendix E reports the backbone, rollout, optimization, regularization, and hardware settings for both model scales. We also document the benchmark protocols, evaluator prompts, and ablation settings needed to reproduce the reported comparisons.

REFERENCES

Donghoon Ahn, Hyoungwon Cho, Jaewon Min, Wooseok Jang, Jungwoo Kim, SeonHwa Kim, Hyun Hee Park, Kyong Hwan Jin, and Seungryong Kim. Self-rectifying diffusion sampling with perturbed-attention guidance. In *European Conference on Computer Vision*, pp. 1–17. Springer, 2024.

Jose A Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. Rudder: Return decomposition for delayed rewards. *Advances in Neural Information Processing Systems*, 32, 2019.

Qingyan Bai, Qiuyu Wang, Hao Ouyang, Yue Yu, Hanlin Wang, Wen Wang, Ka Leong Cheng, Shuailei Ma, Yanhong Zeng, Zichen Liu, et al. Scaling instruction-based video editing with a high-quality synthetic dataset. *arXiv preprint arXiv:2510.15742*, 2025.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *International Conference on Learning Representations*, volume 2024, pp. 4965–4987, 2024.

Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. 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 *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18392–18402. IEEE, 2023.

Duygu Ceylan, Chun-Hao P Huang, and Niloy J Mitra. Pix2video: Video editing using image diffusion. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 23149–23160. IEEE, 2023.

Hila Chefer, Yuval Alaluf, Yael Vinker, Lior Wolf, and Daniel Cohen-Or. Attend-and-excite: Attention-based semantic guidance for text-to-image diffusion models. *ACM transactions on Graphics (TOG)*, 42(4):1–10, 2023.

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

Yinan Chen, Jiangning Zhang, Teng Hu, Yuxiang Zeng, Zhucun Xue, Qingdong He, Chengjie Wang, Yong Liu, Xiaobin Hu, and Shuicheng Yan. Ivebench: Modern benchmark suite for instruction-guided video editing assessment. In *International Conference on Learning Representations*, volume 2026, pp. 137363–137383, 2026b.

Jiaxin Cheng, Tianjun Xiao, and Tong He. Consistent video-to-video transfer using synthetic dataset. In *International Conference on Learning Representations*, volume 2024, pp. 16867–16879, 2024.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

Kevin Clark, Paul Vicol, Kevin Swersky, and David Fleet. Directly fine-tuning diffusion models on differentiable rewards. In *International Conference on Learning Representations*, volume 2024, pp. 4793–4822, 2024.

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

Ruiyi Ding, Jie Li, He Kang, Ziyan Liu, Chengru Song, et al. Agg: Jacobian-aggregated group gradient for efficient grpo training of diffusion models. *arXiv preprint arXiv:2607.17572*, 2026.

Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. Dpok: Reinforcement learning for fine-tuning text-to-image diffusion models. *Advances in Neural Information Processing Systems*, 36:79858–79885, 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*, volume 2024, pp. 1608–1620, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

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

Xiaoxuan He, Siming Fu, Wanli Li, Zhiyuan Li, Dacheng Yin, Kang Rong, Fengyun Rao, and Bo Zhang. Sail: Self-amplified iterative learning for diffusion model alignment with minimal human feedback. *arXiv preprint arXiv:2602.05380*, 2026a.

Xiaoxuan He, Siming Fu, Zeyue Xue, Weijie Wang, Ruizhe He, Yuming Li, Dacheng Yin, Shuai Dong, Haoyang Huang, Hongfa Wang, et al. Flash-grpo: Efficient alignment for video diffusion via one-step policy optimization. *arXiv preprint arXiv:2605.15980*, 2026b.

Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Susung Hong, Gyuseong Lee, Wooseok Jang, and Seungryong Kim. Improving sample quality of diffusion models using self-attention guidance. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7428–7437. IEEE, 2023.

Zijing Hu, Fengda Zhang, Long Chen, Kun Kuang, Jiahui Li, Kaifeng Gao, Jun Xiao, Xin Wang, and Wenwu Zhu. Towards better alignment: Training diffusion models with reinforcement learning against sparse rewards. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23604–23614. IEEE, 2025.

Xuan Ju, Tianyu Wang, Yuqian Zhou, He Zhang, Qing Liu, Nanxuan Zhao, Zhifei Zhang, Yijun Li, Yuanhao Cai, Shaoteng Liu, et al. Editverse: Unifying image and video editing and generation with in-context learning. *arXiv preprint arXiv:2509.20360*, 2025.

Tero Karras, Miika Aittala, Tuomas Kynkäänniemi, Jaakko Lehtinen, Timo Aila, and Samuli Laine. Guiding a diffusion model with a bad version of itself. *Advances in Neural Information Processing Systems*, 37:52996–53021, 2024.

Yuval Kirstain, Adam Polyak, Uriel Singer, Shahbuland Matiana, Joe Penna, and Omer Levy. Pick-a-pic: An open dataset of user preferences for text-to-image generation. *Advances in neural information processing systems*, 36: 36652–36663, 2023.

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

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.

Mingi Kwon, Jaeseok Jeong, and Youngjung Uh. Diffusion models already have a semantic latent space. *arXiv preprint arXiv:2210.10960*, 2022.

Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. In *Artificial intelligence and statistics*, pp. 562–570. Pmlr, 2015.

Kimin Lee, Hao Liu, Moonkyung Ryu, Olivia Watkins, Yuqing Du, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, and Shixiang Shane Gu. Aligning text-to-image models using human feedback. *arXiv preprint arXiv:2302.12192*, 2023.

Minghan Li, Chenxi Xie, Yichen Wu, Lei Zhang, and Mengyu Wang. Five-bench: A fine-grained video editing benchmark for evaluating emerging diffusion and rectified flow models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16672–16681, 2025a.

Rui Li, Yuanzhi Liang, Ziqi Ni, Haibing Huang, Chi Zhang, and Xuelong Li. Rethinking reward signals in video grpo: When scores become targets. *arXiv preprint arXiv:2511.19356*, 2025b.

Xirui Li, Chao Ma, Xiaokang Yang, and Ming-Hsuan Yang. Vidtome: Video token merging for zero-shot video editing. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7486–7495. IEEE, 2024.

Yan Li, Lin Liu, Xiaopeng Zhang, and Qi Tian. Reasoning to align: Implicit reasoning in diffusion transformers for video editing. *arXiv preprint arXiv:2605.24674*, 2026.

Sen Liang, Fengbin Guan, Youliang Zhang, Xin Li, and Zhibo Chen. Cot-edit: Let cot guide instruction video editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 37960–37970, 2026a.

Sen Liang, Cong Wang, Zhentao Yu, Fengbin Guan, Zhengguang Zhou, Teng Hu, Youliang Zhang, Yuan Zhou, Xin Li, Qinglin Lu, et al. Goku: A million-scale universal dataset and benchmark for instruction-based video editing. *arXiv preprint arXiv:2606.30599*, 2026b.

Zhanhao Liang, Yuhui Yuan, Shuyang Gu, Bohan Chen, Tiankai Hang, Mingxi Cheng, Ji Li, and Liang Zheng. Aesthetic post-training diffusion models from generic preferences with step-by-step preference optimization. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13199–13208. IEEE, 2025.

Zhanhao Liang, Tao Yang, Jie Wu, Chengjian Feng, and Liang Zheng. Leapalign: Post-training flow matching models at any generation step by building two-step trajectories. *arXiv preprint arXiv:2604.15311*, 2026c.

Xinyao Liao, Wei Wei, Xiaoye Qu, and Yu Cheng. Step-level reward for free in rl-based t2i diffusion model fine-tuning. *arXiv preprint arXiv:2505.19196*, 2025a.

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

Jianbo Lin, Xiaomin Yu, Yi Xin, Yifu Guo, Zhuosong Jiang, Zhongqi Yue, Weishi Wang, Heqing Zou, Chengwei Qin, and Hui Xiong. Icrl: Learning to internalize self-critique with reinforcement learning. *arXiv preprint arXiv:2605.15224*, 2026a.

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

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-grpo: Training flow matching models via online rl. *Advances in neural information processing systems*, 38:40783–40818, 2026a.

Jie Liu, Gongye Liu, Jiajun Liang, Ziyang Yuan, Xiaokun Liu, Mingwu Zheng, Xiele Wu, Qiulin Wang, Menghan Xia, Xintao Wang, et al. Improving video generation with human feedback. *Advances in Neural Information Processing Systems*, 38:82155–82192, 2026b.

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

Xinyu Liu, Hangjie Yuan, Yujie Wei, Jiazheng Xing, Yujin Han, Jiahao Pan, Yanbiao Ma, Chi-Min Chan, Kang Zhao, Shiwei Zhang, et al. Towards reason-informed video editing in unified models with self-reflective learning. *arXiv preprint arXiv:2512.09924*, 2025.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in neural information processing systems*, 36:46534–46594, 2023.

Jinjie Mai, Chaoyang Wang, Gordon Guocheng Qian, Willi Menapace, Sergey Tulyakov, Bernard Ghanem, Peter Wonka, and Ashkan Mirzaei. Easyv2v: A high-quality instruction-based video editing framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 30435–30445, 2026.

Chenxuan Miao, Yutong Feng, Yi Lu, Yunfeng Yan, Donglian Qi, Shiwei Zhang, Yu Liu, Xi Chen, and Hengshuang Zhao. Omniedit-bench: A comprehensive benchmark for instruction-based video editing. *arXiv preprint arXiv:2608.05049*, 2026.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Zhuohan Ouyang, Zhe Qian, Wenhuo Cui, and Chaoqun Wang. Region-constrained group relative policy optimization for flow-based image editing. *arXiv preprint arXiv:2604.09386*, 2026.

Jinyoung Park, Jeehye Na, Jinyoung Kim, and Hyunwoo J Kim. Deepvideo-r1: Video reinforcement fine-tuning via difficulty-aware regressive grpo. *Advances in Neural Information Processing Systems*, 38:138605–138632, 2026.

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

Mihir Prabhudesai, Anirudh Goyal, Deepak Pathak, and Katerina Fragkiadaki. Aligning text-to-image diffusion models with reward backpropagation. 2023.

Mihir Prabhudesai, Russell Mendonca, Zheyang Qin, Katerina Fragkiadaki, and Deepak Pathak. Video diffusion alignment via reward gradients. *arXiv preprint arXiv:2407.08737*, 2024.

Chenyang Qi, Xiaodong Cun, Yong Zhang, Chenyang Lei, Xintao Wang, Ying Shan, and Qifeng Chen. Fatezero: Fusing attentions for zero-shot text-based video editing. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15886–15896. IEEE, 2023.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Yiqing Shen, Chenjia Li, and Mathias Unberath. Text-driven reasoning video editing via reinforcement learning on digital twin representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3945–3954, 2026.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

Jaehyeon Son, Soochan Lee, and Gunhee Kim. Distilling reinforcement learning algorithms for in-context model-based planning. In *International Conference on Learning Representations*, volume 2025, pp. 37293–37310, 2025.

Hongzu Su, Lichao Meng, Lei Zhu, Ke Lu, and Jingjing Li. Ddpo: Direct dual propensity optimization for post-click conversion rate estimation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1179–1188, 2024.

DecartAI Team. Lucy edit: Open-weight text-guided video editing, 2025.

Yoad Tewel, Yuval Atzmon, Gal Chechik, and Lior Wolf. Bootstrap your generator: Unpaired visual editing with flow matching. *arXiv preprint arXiv:2606.03911*, 2026.

Akiyoshi Tomihari and Issei Sato. Power distribution bridges sampling, self-reward rl, and self-distillation. *arXiv preprint arXiv:2605.04542*, 2026.

Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam, Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion model alignment using direct preference optimization. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8228–8238. IEEE, 2024.

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

Weicheng Wang, Zhicheng Zhang, Zhongqi Zhang, Juncheng Zhou, Yongjie Zhu, Wenyu Qin, Meng Wang, Pengfei Wan, and Jufeng Yang. Live: Leveraging image manipulation priors for instruction-based video editing. *arXiv preprint arXiv:2604.17021*, 2026.

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

Haoxuan Wu, Lai Man Po, Mengyang Liu, Kun Li, Hongzheng Yang, and Wei Liu. Through the prism: Preference representation in intermediate states of video diffusion models. *arXiv preprint arXiv:2606.20310*, 2026.

Jay Zhangjie Wu, Yixiao Ge, Xintao Wang, Stan Weixian Lei, Yuchao Gu, Yufei Shi, Wynne Hsu, Ying Shan, Xiaohu Qie, and Mike Zheng Shou. Tune-a-video: One-shot tuning of image diffusion models for text-to-video generation. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7589–7599. IEEE, 2023a.

Xiaoshi Wu, Yiming Hao, Keqiang Sun, Yixiong Chen, Feng Zhu, Rui Zhao, and Hongsheng Li. Human preference score v2: A solid benchmark for evaluating human preferences of text-to-image synthesis. *arXiv preprint arXiv:2306.09341*, 2023b.

Yuhui Wu, Liyi Chen, Ruibin Li, Shihao Wang, Chenxi Xie, and Lei Zhang. Insvie-1m: Effective instruction-based video editing with elaborate dataset construction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16692–16701, 2025.

Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. Imagereward: Learning and evaluating human preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36:15903–15935, 2023.

Jiazheng Xu, Yu Huang, Jiale Cheng, Yuanming Yang, Jiajun Xu, Yuan Wang, Wenbo Duan, Shen Yang, Qunlin Jin, Shurun Li, et al. Visionreward: Fine-grained multi-dimensional human preference learning for image and video generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 11269–11277, 2026.

Zeyue Xue, Jie Wu, Yu Gao, Fangyuan Kong, Lingting Zhu, Mengzhao Chen, Zhiheng Liu, Wei Liu, Qiushan Guo, Weilin Huang, et al. Dancegrpo: Unleashing grpo on visual generation. *arXiv preprint arXiv:2505.07818*, 2025.

Hao Yang, Zhiyu Tan, Jia Gong, Luozheng Qin, Hesen Chen, Xiaomeng Yang, Yuqing Sun, Yuetan Lin, Mengping Yang, and Hao Li. Omni-video 2: Scaling mllm-conditioned diffusion for unified video generation and editing. *arXiv preprint arXiv:2602.08820*, 2026.

Shuai Yang, Yifan Zhou, Ziwei Liu, and Chen Change Loy. Rerender a video: Zero-shot text-guided video-to-video translation. In *SIGGRAPH Asia 2023 Conference Papers*, pp. 1–11, 2023.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. In *International Conference on Learning Representations*, volume 2025, pp. 83048–83077, 2025.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*, 2024.

Chi Zhang, Chengjian Feng, Feng Yan, Qiming Zhang, Mingjin Zhang, Yujie Zhong, Jing Zhang, and Lin Ma. Instructvedit: A holistic approach for instructional video editing. *arXiv preprint arXiv:2503.17641*, 2025a.

Jaxon Zhang, Binxin Yang, Hubery Yin, Chen Li, and Jing Lyu. Drm: Diffusion-based reward model with step-wise guidance. *arXiv preprint arXiv:2605.25661*, 2026a.

Qi Zhang, Dawei Wang, and Shaofeng Zou. Step-level denoising-time diffusion alignment with multiple objectives. *arXiv preprint arXiv:2604.14379*, 2026b.

Xinyao Zhang, Wenkai Dong, Yuxin Song, Bo Fang, Qi Zhang, Jing Wang, Fan Chen, Hui Zhang, Haocheng Feng, Yu Lu, et al. Sama: Factorized semantic anchoring and motion alignment for instruction-guided video editing. *arXiv preprint arXiv:2603.19228*, 2026c.

Yabo Zhang, Yuxiang Wei, XIAOPENG ZHANG, Wangmeng Zuo, Qi Tian, et al. Controlvideo: Training-free controllable text-to-video generation. In *International Conference on Learning Representations*, volume 2024, pp. 54441–54461, 2024.

Zhongwei Zhang, Fuchen Long, Wei Li, Zhaofan Qiu, Wu Liu, Ting Yao, and Tao Mei. Region-constraint in-context generation for instructional video editing. *arXiv preprint arXiv:2512.17650*, 2025b.

Ziyi Zhang, Li Shen, Sen Zhang, Deheng Ye, Yong Luo, Miaojing Shi, Dongjing Shan, Bo Du, and Dacheng Tao. Aligning few-step diffusion models with dense reward difference learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026d.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023.

Xingyu Zhou, Qifan Li, Xiaobin Hu, Hai Chen, and Shuhang Gu. Guiding a diffusion transformer with the internal dynamics of itself. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11536–11545, 2026.

Bojia Zi, Penghui Ruan, Marco Chen, Xianbiao Qi, Shaozhe Hao, Shihao Zhao, Youze Huang, Bin Liang, Rong Xiao, and Kam-Fai Wong. Señorita-2m: A high-quality instruction-based dataset for general video editing by video specialists. *Advances in Neural Information Processing Systems*, 38, 2026.

A Related Work

RL fine-tuning of visual generative models. Aligning generative models with non-differentiable, preference-based objectives via RL from human feedback (Christiano et al., 2017; Ouyang et al., 2022; Schulman et al., 2017) has become standard in language modeling, and Group Relative Policy Optimization (GRPO) (Shao et al., 2024; Guo et al., 2025) further removes the learned value network by forming advantages from relative comparisons within a sampled group. In visual generation, the same recipe is realized either by casting denoising as a multi-step decision process and optimizing a reward on the generated sample (Lee et al., 2023; Black et al., 2024; Fan et al., 2023), by backpropagating differentiable reward gradients through the sampler (Prabhudesai et al., 2023; Clark et al., 2024; Prabhudesai et al., 2024), by preference-based objectives that bypass explicit RL (Rafailov et al., 2023; Wallace et al., 2024), or by scaling GRPO to image, flow, and video generation (Xue et al., 2025; Liu et al., 2026a; Park et al., 2026), with the reward signal supplied by learned preference models for images (Xu et al., 2023; Kirstain et al., 2023; Wu et al., 2023b) or videos (Xu et al., 2026; Liu et al., 2026b). As this recipe has moved to video, the dominant concerns have become the cost of a rollout, addressed by one-step or aggregated-gradient policy updates (He et al., 2026b; Ding et al., 2026); the spatial locality of the update, addressed by restricting the objective to edited regions (Ouyang et al., 2026); and the reliability of the score being optimized, since a reward that becomes a target is readily gamed (Li et al., 2025b).

These pipelines share a common structure: a full denoising trajectory is rolled out, and a single scalar reward is assigned to the final decoded sample, usually obtained by composing several specialized reward models with hand-tuned weights. A recent line of work relieves this sparsity by supervising the denoising chain more densely—by reward shaping along the chain (Hu et al., 2025), by comparing samples at individual denoising steps rather than only at the end (Liang et al., 2025; Zhang et al., 2026b), by having the reward model emit step-wise guidance in place of a terminal score (Zhang et al., 2026a), or by reading step-level rewards off the model's own intermediate predictions (Liao et al., 2025a). All of these densify along the *sampling-time* axis, i.e., across denoising steps, while treating each network evaluation as an atomic black box. VISTA densifies along the *architectural* axis instead, in the spirit of deep supervision (Lee et al., 2015) but with a reward rather than a label: it decodes and self-evaluates hidden states at selected transformer layers within a single evaluation. The two axes are complementary, and our layer-wise updates are combined with a two-step trajectory surrogate that keeps the sampling-time cost tractable.

Exploiting internal representations of diffusion transformers. Beyond classifier-free guidance (Ho & Salimans, 2022), a growing body of work shows that a diffusion model's own internals carry semantically meaningful and directly exploitable structure: intermediate feature spaces admit semantic directions (Kwon et al., 2022), attention maps can be perturbed or re-excited to steer sampling (Hong et al., 2023; Ahn et al., 2024; Chefer et al., 2023), and a degraded or shallower version of the model can serve as its own guidance signal (Karras et al., 2024). Most relevant to us, the layer-wise internal dynamics of a diffusion transformer can be projected into partial predictions and used to guide generation at inference time without any external model (Zhou et al., 2026). We build on this observation, but use intermediate states as a *training* signal rather than an inference-time guidance term: each selected layer decodes a group of candidates, self-evaluates them, and distills the winner into a token-level importance mask that both conditions subsequent layers and reweights a layer-wise GRPO update.

B Theory Analysis

The potential bias arises because VISTA reuses one old-policy trajectory and detached incoming memories across the K layer-wise updates. We therefore analyze memory staleness together with the other two sources of discrepancy introduced by this reuse: distribution shift from the behavior policy and the LeapAlign approximation. The resulting bound gives explicit conditions under which every fixed-rollout update also improves its corresponding fresh-rollout objective.

Let θ_0 be the behavior-policy parameters used to generate the stored trajectory $\tau \sim \mathcal{P}_0$. Before update k, the incoming memory $\bar{\mathbf{m}}_{k-1} = \mathbf{m}_{k-1}(\theta_{k-1})$ has been computed at the preceding checkpoint and detached. The ideal objective would instead regenerate a trajectory from the current policy and recompute the memory at the current parameters.

For a parameter–memory pair $(\theta, \mathbf{m})$, define the on-policy checkpoint loss

$$F_k(\theta, \mathbf{m}) := \mathbb{E}_{\tau \sim \mathcal{P}_{\theta,\mathbf{m}}} \left[\ell_k(\theta, \mathbf{m}; \tau) \right], \tag{12}$$

where $\mathcal{P}_{\theta,\mathbf{m}}$ is the trajectory distribution under the memory-augmented conditioning and ℓ_k is the per-trajectory loss at checkpoint k. We abbreviate the fresh-memory objective as $F_k(\theta) := F_k(\theta, \mathbf{m}_{k-1}(\theta))$. The implemented objective $\widetilde{F}_k(\theta)$ evaluates stored trajectories from $\mathcal{P}_0$ using the detached memory $\bar{\mathbf{m}}_{k-1}$, a clipped approximate importance

weight, and LeapAlign. When $\mathcal{P}_{\theta,\bar{\mathbf{m}}_{k-1}}$ is absolutely continuous with respect to $\mathcal{P}_0$, let

$$w_k(\tau) := \frac{\mathrm{d}\mathcal{P}_{\theta,\bar{\mathbf{m}}_{k-1}}}{\mathrm{d}\mathcal{P}_0}(\tau) \tag{13}$$

be the exact importance weight and let $\bar{w}_k = \mathrm{clip}(w_k, 1-\epsilon, 1+\epsilon)$.

For each checkpoint k, let Ω_k be a local parameter region containing the parameters immediately before and after its update. We assume:

1. $\mathcal{P}_{\theta,\bar{\mathbf{m}}_{k-1}} \ll \mathcal{P}_0$ and the per-trajectory loss is bounded, $|\ell_k(\theta, \mathbf{m}; \tau)| \le B$, for every $\theta \in \Omega_k$;
2. $F_k(\theta, \mathbf{m})$ is L_m-Lipschitz in $\mathbf{m}$, and the memory map is locally L_θ-Lipschitz in θ;
3. there exist finite r_k and κ_k such that, for every $\theta \in \Omega_k$,
$$\|\theta - \theta_{k-1}\| \le r_k, \qquad D_{\mathrm{KL}}\big(\mathcal{P}_{\theta,\bar{\mathbf{m}}_{k-1}} \,\|\, \mathcal{P}_0\big) \le \kappa_k; \tag{14}$$
4. the residual between the exactly clipped fixed-rollout objective and the implemented LeapAlign objective is uniformly bounded on Ω_k:
$$\Big|\mathbb{E}_{\tau\sim\mathcal{P}_0}[\bar{w}_k(\tau)\ell_k(\theta, \bar{\mathbf{m}}_{k-1}; \tau)] - \widetilde{F}_k(\theta)\Big| \le \bar{\delta}_k. \tag{15}$$

The Lipschitz assumption is intended locally, between changes of the argmax-selected winner in Eq. (5); it does not assert global smoothness across winner switches.

Proposition 1 (Fixed-Rollout Surrogate Gap)**.** *Under the assumptions above, the discrepancy between the ideal fresh objective and the implemented fixed-rollout surrogate satisfies*

$$\Big|F_k(\theta) - \widetilde{F}_k(\theta)\Big| \le L_m L_\theta \,\|\theta - \theta_{k-1}\| + B\sqrt{2D_{\mathrm{KL}}\big(\mathcal{P}_{\theta,\bar{\mathbf{m}}_{k-1}} \,\|\, \mathcal{P}_0\big)} + \bar{\delta}_k. \tag{16}$$

Consequently, throughout Ω_k,

$$\Big|F_k(\theta) - \widetilde{F}_k(\theta)\Big| \le \varepsilon_k := L_m L_\theta r_k + B\sqrt{2\kappa_k} + \bar{\delta}_k. \tag{17}$$

Proof of Proposition 1. Insert the stale-memory on-policy objective $F_k(\theta, \bar{\mathbf{m}}_{k-1})$ and apply the triangle inequality:

$$\begin{aligned} |F_k(\theta) - \widetilde{F}_k(\theta)| \le\ & |F_k(\theta, \mathbf{m}_{k-1}(\theta)) - F_k(\theta, \bar{\mathbf{m}}_{k-1})| \\ & + \Big|\mathbb{E}_{\mathcal{P}_0}[w_k\ell_k] - \widetilde{F}_k(\theta)\Big| \end{aligned} \tag{18}$$

The first term is bounded by

$$L_m\|\mathbf{m}_{k-1}(\theta) - \bar{\mathbf{m}}_{k-1}\| \le L_m L_\theta \|\theta - \theta_{k-1}\|. \tag{19}$$

For the second term, insert the clipped exact weight $\bar{w}_k$:

$$\begin{aligned} \Big|\mathbb{E}_{\mathcal{P}_0}[w_k\ell_k] - \widetilde{F}_k(\theta)\Big| &\le B\,\mathbb{E}_{\mathcal{P}_0}|w_k - \bar{w}_k| \\ &\quad + \Big|\mathbb{E}_{\mathcal{P}_0}[\bar{w}_k\ell_k] - \widetilde{F}_k(\theta)\Big| \\ &\le B\sqrt{2D_{\mathrm{KL}}\big(\mathcal{P}_{\theta,\bar{\mathbf{m}}_{k-1}} \,\|\, \mathcal{P}_0\big)} + \bar{\delta}_k. \end{aligned} \tag{20}$$

The final inequality uses $|w_k - \bar{w}_k| \le |w_k - 1|$, $\mathbb{E}_{\mathcal{P}_0}|w_k - 1| = 2D_{\mathrm{TV}}(\mathcal{P}_{\theta,\bar{\mathbf{m}}_{k-1}}, \mathcal{P}_0)$, Pinsker's inequality, and Eq. (15). Combining this with the memory term yields Eq. (16); applying Eq. (14) yields Eq. (17). □

Corollary B.0.1 (Validity across all checkpoints)**.** *Suppose the update at every checkpoint $k \in \{1, \dots, K\}$ remains in Ω_k and achieves a surrogate decrease*

$$\widetilde{F}_k(\theta_{k+1}) \le \widetilde{F}_k(\theta_k) - D_k. \tag{21}$$

If, for every checkpoint,

$$D_k > 2\varepsilon_k = 2\big(L_m L_\theta r_k + B\sqrt{2\kappa_k} + \bar{\delta}_k\big), \tag{22}$$

then every update decreases its corresponding ideal fresh-rollout, fresh-memory objective:

$$F_k(\theta_{k+1}) < F_k(\theta_k), \qquad k = 1, \dots, K. \tag{23}$$

Thus the same behavior-policy rollout can be reused across all K checkpoints without changing the direction of improvement relative to the checkpoint-wise fresh objectives.

Proof. Applying Eq. (17) at both endpoints gives

$$F_k(\theta_{k+1}) \le F_k(\theta_k) - D_k + 2\varepsilon_k. \tag{24}$$

Equation (22) makes the final two terms strictly negative for every k. □

Interpretation and scope. This result supports the main efficiency choice of VISTA: one rollout can supervise all K layer-wise updates without regeneration when each update satisfies the explicit condition in Eq. (22). The three terms have direct operational meanings. The radius r_k measures parameter and memory drift, κ_k measures departure from the behavior rollout, and $\bar{\delta}_k$ measures the error introduced by LeapAlign. The update norm is directly computable, the behavior-policy KL can be estimated from stored log-probabilities, and $\bar{\delta}_k$ can be calibrated by periodically comparing LeapAlign with the full-trajectory objective on held-out minibatches. When the observed surrogate decrease exceeds twice their combined bound, the corollary guarantees genuine improvement of the corresponding fresh objective while reducing K rollouts to one. This statement does not rely on GRPO clipping alone implying a small KL; it requires the trust-region bounds to hold. The guarantee is also local to regions where the selected winner is unchanged and the attribution mass remains nonzero. If either the trust-region or descent condition fails, a fresh rollout restores the reference point for the next sequence of updates.

C Importance Concentration

This section defines the per-position importance and the Importance Concentration reported in Fig. 1. Let $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$ denote the context embeddings that condition the backbone, with $\mathbf{x}_n \in \mathbb{R}^d$ at context position n, and let $\mathbf{p}^{(\ell)}$ be the prediction read out of block ℓ. We do not differentiate the reward, which is produced by a non-differentiable VLM judge; instead we differentiate the energy of the block's own prediction,

$$s^{(\ell)} \;=\; \frac{1}{|\mathbf{p}^{(\ell)}|}\,\big\|\mathbf{p}^{(\ell)}\big\|_2^2, \tag{25}$$

which measures how strongly block ℓ is driving the edit at that depth. Back-propagating $s^{(\ell)}$ to the context embeddings and contracting the gradient with the input gives the grad×input saliency at each position,

$$a_n^{(\ell)} \;=\; \left|\left\langle \frac{\partial s^{(\ell)}}{\partial \mathbf{x}_n},\, \mathbf{x}_n \right\rangle\right|, \qquad p_n^{(\ell)} \;=\; \frac{a_n^{(\ell)}}{\sum_{n'=1}^{N} a_{n'}^{(\ell)}}, \tag{26}$$

where the normalization makes $\mathbf{p}^{(\ell)}$ a distribution over the N context positions. The first-order term $\langle \partial s/\partial \mathbf{x}_n, \mathbf{x}_n \rangle$ is the standard grad×input estimate of the contribution of position n, and the absolute value keeps contributions that suppress the prediction as well as those that amplify it. Finally, we summarize this distribution by its normalized inverse entropy,

$$c^{(\ell)} \;=\; 1 \;-\; \frac{H\big(\mathbf{p}^{(\ell)}\big)}{\log N}, \qquad H\Big(\mathbf{p}^{(\ell)}\Big) = -\sum_{n=1}^{N} p_n^{(\ell)} \log p_n^{(\ell)}. \tag{27}$$

Despite being built from an entropy, $c^{(\ell)}$ runs in the opposite direction: $c^{(\ell)} = 1$ when all importance sits on a single context position and $c^{(\ell)} = 0$ when it is spread uniformly, so larger values indicate a more focused read of the instruction. Dividing by $\log N$ makes the score comparable across prompts of different length. In Fig. 1 we evaluate Eqs. (25)–(27) at every block of a single rollout, retaining the graph between blocks so that one backward pass per block suffices. Panel (c) of that figure plots $c^{(\ell)}$ against depth alongside the reward of the corresponding intermediate edit, while panel (b) lists, for each block, the three context positions with the largest $p_n^{(\ell)}$, which is how we read off the instruction words the block attends to most.

D LeapAlign: Two-Step Trajectory Surrogate

This section details the *LeapAlign* surrogate introduced in Sec. 3.3, which approximates the full T-step trajectory objective of Eq. (11) with only two policy evaluations per candidate while producing one log-probability per self-evaluation layer (*pyramid head*).

Two-step leap. Let $\{\sigma_t\}_{t=0}^{T}$ denote the noise schedule and let t_1 be the current denoising index. Instead of scoring every transition, LeapAlign samples a leap target

$$t_2 \;=\; t_1 + \Delta, \qquad \Delta \in \big[\Delta_{\min},\; T - 2 - t_1\big], \tag{28}$$

where $\Delta_{\min}$ is a minimum gap; if no valid t_2 fits, LeapAlign falls back to the single-step estimator and broadcasts its log-probability to all heads. A first forward pass at t_1 yields the velocity prediction $v_\theta(\mathbf{z}_{t_1}, t_1)$, and an Euler leap estimates the latent at t_2,

$$\hat{\mathbf{z}}_{t_2} \;=\; \mathbf{z}_{t_1} + (\sigma_{t_2} - \sigma_{t_1})\, v_\theta(\mathbf{z}_{t_1}, t_1). \tag{29}$$

To keep the surrogate anchored to the actually stored rollout latent $\mathbf{z}_{t_2}^{\text{act}}$ while preserving gradients through the leap, we apply a straight-through re-anchoring

$$\tilde{\mathbf{z}}_{t_2} \;=\; \hat{\mathbf{z}}_{t_2} + \mathrm{sg}\big(\mathbf{z}_{t_2}^{\text{act}} - \hat{\mathbf{z}}_{t_2}\big)\,, \tag{30}$$

where $\mathrm{sg}(\cdot)$ is the stop-gradient operator.

Per-layer log-probabilities. A second forward pass at t_2, evaluated with a pyramid block split of the L backbone blocks into the K self-evaluation heads, returns per-head velocity predictions $\{v_\theta^{(k)}(\tilde{\mathbf{z}}_{t_2}, t_2)\}_{k=1}^{K}$. For each head we compute a single log-probability at index t_2 via the group-relative denoising step, giving the per-layer log-probabilities that replace the full-trajectory sum in the ratio of Eq. (10). The noise scale and step size depend only on the schedule and index and are therefore shared across heads. Thus a single leap ($t_1 \to t_2$) produces the two-step surrogate used by every layer, reducing the per-candidate cost from T policy passes to 2.

E Detailed Training Settings

E.1 Training Parameters

This section lists the full training configurations for the 5B and 14B models summarized in Sec. 4.

Backbone and adaptation. The 5B model is built on Wan2.2-TI2V-5B and fully fine-tuned. The 14B model is built on Wan2.1-T2V-14B and adapted with LoRA (rank 64, $\alpha = 64$), applied to the q, k, v, o projections and the first/third feed-forward sublayers.

Rollout. The 5B model samples $G = 4$ candidates per prompt with a 30-step denoising trajectory, stochasticity $\eta = 0.05$, flow-matching timestep shift 5.0, and classifier-free guidance scale 4.5 at inference. The 14B model uses the same $G = 4$ but a shorter 20-step trajectory, $\eta = 0.1$, and guidance scale 5.0, together with a reduced resolution (33 frames at 480×832) to fit the larger backbone in memory.

GRPO objective and KL regularization. The 5B model clips the importance ratio at $\epsilon = 0.1$ while the 14B model uses $\epsilon = 0.2$; both clip the advantage magnitude at 5.0. The KL penalty toward the reference policy uses $\beta = 0.4$ with an adaptive schedule (target KL 0.01, horizon 50 steps) for the 5B model, whose reference policy is instantiated by disabling the current adapter; the 14B model instead uses a small fixed $\beta = 0.05$ with no explicit reference policy, relying on the LoRA update's limited capacity for regularization.

Optimization. We optimize the 5B model with learning rate $5e^{-7}$, 1 inner GRPO epoch, and gradient accumulation of 2, and the 14B model with a larger LoRA learning rate of $5e^{-5}$ and no gradient accumulation; both train for 300 steps with checkpoints saved every 25 steps.

Layer-wise self-evaluation and return-to-go. We instantiate $K = 4$ self-evaluation layers for the 5B model and $K = 3$ layer snapshots for the 14B model, scoring each layer's intermediate output with a role-agnostic pointwise reward judged by a Qwen3-VL-based model. Return-to-go credit assignment uses a discount factor $\gamma = 0$ for the 5B model (each layer credited only by its own reward) and $\gamma = 0.5$ for the 14B model (credit shared more evenly across downstream layers), both with uniform layer weighting. The additive token-importance memory of Eq. (8) is carried from layer ℓ to $\ell + 1$ with strength 0.5 in the 14B run, and its normalization floor is set to $\epsilon_0 = 10^{-12}$ (Eq. 7). The 14B configuration additionally applies a layer-wise correction to the advantage with mixing coefficient $\eta_{\text{lw}} = 0.3$, weight $\lambda_{\text{lw}} = 0.5$,

LeapAlign and infrastructure. For the LeapAlign surrogate, both models sample the leap gap Δ between 1 and 4 denoising steps with gradient checkpointing through the leap; the nested-gradient coefficient is 1.0 for the 5B model and 0.3 for the 14B model. The 5B model is trained with DeepSpeed ZeRO stage 1 under bf16 mixed precision, while the 14B model uses LoRA with gradient checkpointing under a single-node multi-GPU setup.

E.2 Reward Calculation

We use the following prompt to obtain the reward scores. The placeholders {`n_src`}, {`n_gen`}, and {`instruction`} are replaced with the number of sampled source frames, the number of sampled edited frames, and the corresponding editing instruction, respectively.

```
You are a strict, detail-oriented evaluator for video-editing quality.
You will be shown frames from a SOURCE video and an EDITED video that
was generated by applying an editing instruction.

Inputs (in order):
- {n_src} frames from the SOURCE video (before editing)
- {n_gen} frames from the EDITED video (after applying the instruction)

Editing instruction: {instruction}

Score the edit on three dimensions, each on a continuous 1-5 scale.
Use FLOAT scores with one decimal place whenever possible (e.g. 2.3,
3.7, 4.2), not just integers or .5 values. Avoid saturated extreme
scores: 1.0 and 5.0 should be VERY rare. Use 1.0 only when the edit
is completely failed / destructive, and use 5.0 only when the result
is truly flawless across every frame. For almost all real outputs,
prefer nuanced scores in the open interval (1.0, 5.0), such as
1.3, 2.6, 3.4, or 4.7.

Instruction Compliance anchors (interpolate with decimals):
  ~1.2 = ignored / wrong target;
  ~2.0 = attempted but wrong object or action;
  ~3.0 = right target with clear errors (wrong colour / count / partial);
  ~4.0 = correct with minor flaws;
  ~4.8 = near-flawless across all frames. Reserve 5.0 for truly perfect only.

Consistency & Detail Fidelity anchors (interpolate with decimals):
  ~1.2 = scene nearly unrecognisable;
  ~2.0 = obvious distortions outside the edit region;
  ~3.0 = structure ok, local warping or jerkiness;
  ~4.0 = near-perfect, only subtle issues;
  ~4.8 = non-edited details almost identical. Reserve 5.0 for exact
  preservation only.

Visual Quality & Stability anchors (interpolate with decimals):
  ~1.2 = severe artifacts / flicker;
  ~2.0 = distracting seams or colour mismatch;
  ~3.0 = tolerable flicker or blur;
  ~4.0 = minor edge softness;
  ~4.8 = visually excellent. Reserve 5.0 for indistinguishable real
  footage only.

OUTPUT RULES (read carefully -- the response is parsed by a script):
- Your reply MUST consist of EXACTLY two blocks, in this order:
    1) <think>...</think>  -- AT MOST 50 words of internal reasoning.
    2) <answer>...</answer> -- the three score lines, nothing else.
- Inside <answer>, output the three lines BELOW, one per line, no
  bullets, no markdown, no extra commentary:
    Instruction Compliance: <score>
    Consistency & Detail Fidelity: <score>
    Visual Quality & Stability: <score>
- Each <score> is a number in [1, 5] with preferably ONE decimal
  place (e.g. 2.3, 3.7, 4.2). Optionally suffix with '/5'
  (e.g. '3.7/5'). Avoid exact 1.0 or 5.0 unless the result is
  catastrophic or flawless.
- Keep <think> SHORT. If you run out of room, STOP thinking and
  emit <answer> immediately -- a missing score block is treated as a
  failed evaluation. NEVER skip <answer>.

Example of a valid response (do not copy the scores, just the
structure):
<think>Target object recoloured correctly but background flickers;
minor warping near edge.</think>
<answer>
```

```
Instruction Compliance: 4.1
Consistency & Detail Fidelity: 3.2
Visual Quality & Stability: 2.6
</answer>
```

The complete reward-calculation pipeline is illustrated in Figure 6. For each candidate, uniformly sampled frames from the source and edited videos are presented to the reward model together with the editing instruction. The model independently scores instruction compliance, consistency and detail fidelity, and visual quality and stability on a continuous scale from 1 to 5. Their mean is used as the scalar reward for group-relative advantage estimation.

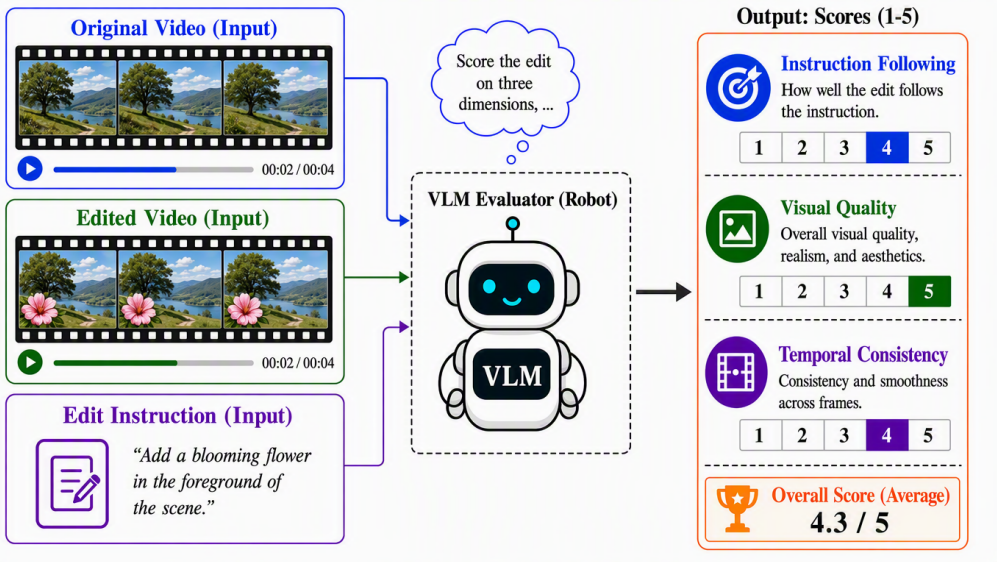


Figure 6: **Reward-calculation pipeline.** Sampled frames from the source video and an edited candidate, together with the editing instruction, are provided to a vision-language reward model. The model evaluates the candidate along three complementary dimensions: instruction compliance, consistency and detail fidelity, and visual quality and stability. The three scores are averaged to produce the scalar reward used by VISTA.

F More Visualization Results

The main text compares our 5B model against all baselines at once (Fig. 3). Here we adopt the capacity-matched view used by the appendix tables: our 5B model is compared against the three $\leq$5B baselines (Appendix F.2) and our 14B model against the four $\geq$13B baselines (Appendix F.1). Examples are drawn from both benchmarks and cover the OpenVE-Bench editing categories as well as the six FiVE-Bench editing types. In every figure, all methods receive identical source videos and editing instructions and are sampled with identical inference settings, and the two scale groups share the same examples, so that the two groups can be read side by side.

Across the examples, VISTA more reliably applies the requested edit while preserving untargeted objects, scene structure, and source motion. Competing methods more frequently under-apply the instruction, modify regions unrelated to the requested edit, or introduce appearance and temporal inconsistencies. These examples complement the quantitative results in Sec. 4 and illustrate the practical benefits of intermediate self-evaluation and token-importance memory.

F.1 $\geq$13B Scale

Figures 7–10 repeat the comparison at the larger scale, pitting our 14B model against Ditto, UniVideo, VINO, and SAMA on the same source videos and instructions used above. The same trends hold: the capacity gain of the $\geq$13B baselines mainly improves visual quality, while instruction compliance and preservation of untargeted content remain the limiting factors that VISTA addresses. Object-level localization remains the main differentiator, as the larger baselines render the edited region more sharply than their $\leq$5B counterparts but still tend to modify neighboring content or miss the referenced attribute.

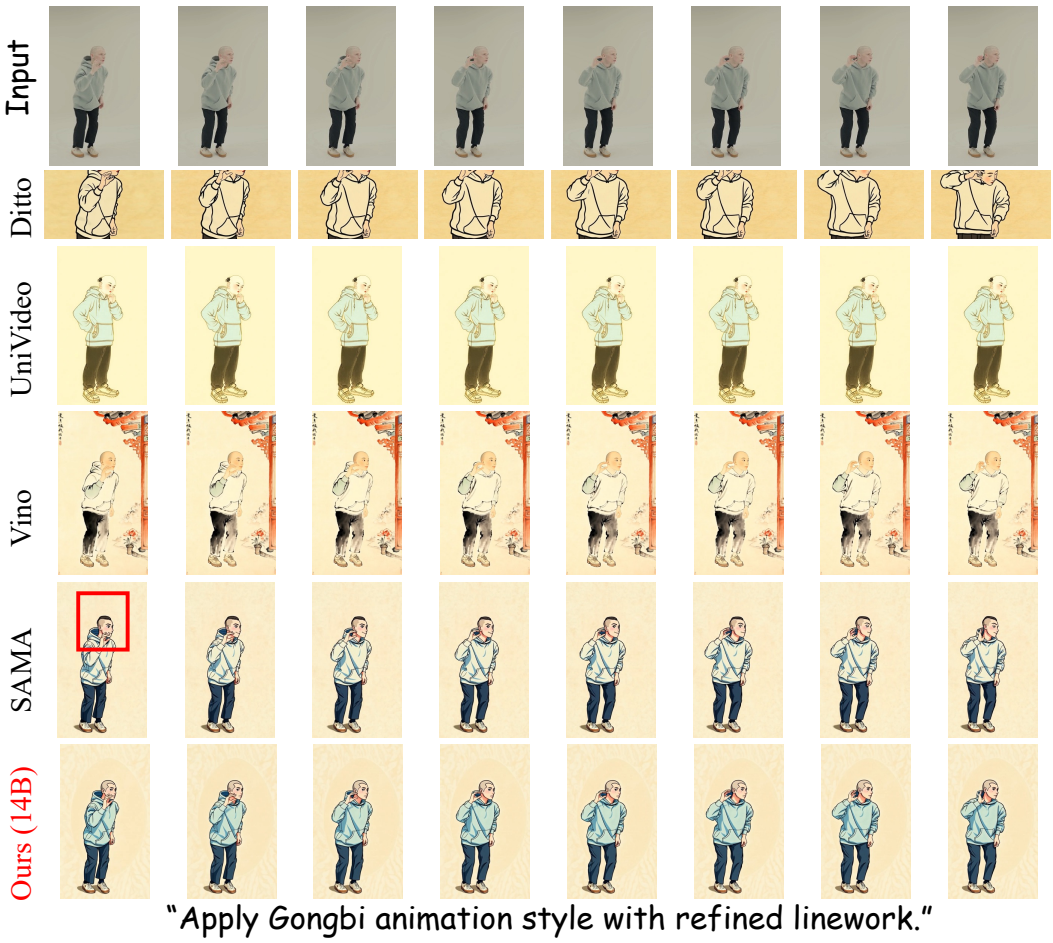


Figure 7: **Qualitative comparison at the ≥13B scale (I).** Each example shows the source video and editing instruction together with outputs from the four ≥13B baselines and our 14B model.

F.2 ≤5B Scale

Figures 11–14 compare our 5B model with Omni-Video, Lucy-Edit, and Kiwi-Edit. They also provide the visual evidence for the under-editing behavior of Lucy-Edit discussed in Appendix G.3: its outputs remain close to the source video in most examples, which explains its strong preservation metrics despite weak instruction compliance.

G Experiment Results

G.1 Computation Cost

Increasing the number of self-evaluation checkpoints K introduces additional intermediate decoding and reward-model evaluations. We profile this overhead on the 5B backbone, varying K at a fixed discount (Table 3) and the return-to-go discount γ at a fixed checkpoint density (Table 4). Both tables report the measured wall-clock time, reward-evaluation workload, and peak memory usage.

The computational cost grows approximately linearly with K (Table 3). Mean iteration time rises from 519.8 seconds for $K = 2$ to 836.9, 1132.5, and 1416.8 seconds for $K = 4$, 6, and 8, respectively, while reward-model calls increase from 48 to 144 per iteration and reward evaluation accounts for roughly half of the iteration time. Peak memory remains within 131.7–139.8 GiB rather than growing systematically with K, because intermediate outputs are decoded and scored sequentially.

The discount γ has no comparable effect on cost (Table 4), since it only reweights rewards that have already been computed. The number of reward evaluations is constant at 80 per iteration, and iteration time varies within 822.1–



Figure 8: **Qualitative comparison at the ≥13B scale (II).** Each example shows the source video and editing instruction together with outputs from the four ≥13B baselines and our 14B model.



Figure 9: **Qualitative comparison at the ≥13B scale (III).** Each example shows the source video and editing instruction together with outputs from the four ≥13B baselines and our 14B model.

894.9 seconds without a monotone trend, a spread consistent with run-to-run scheduling variation rather than with γ itself.

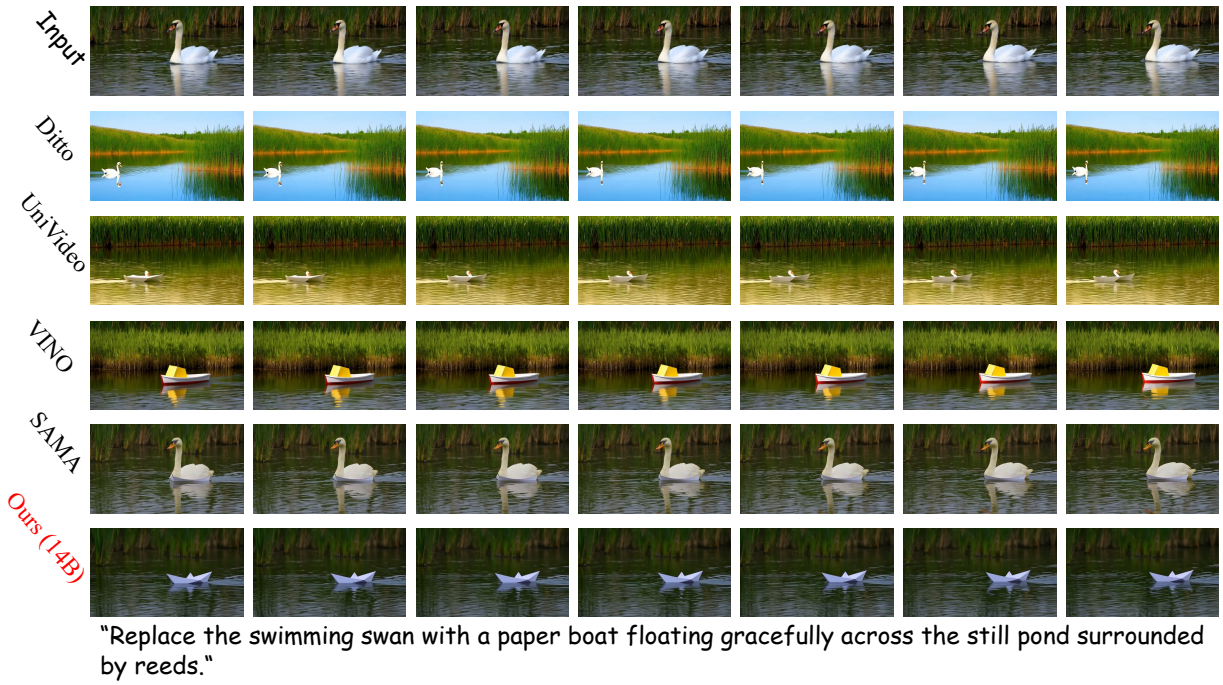


Figure 10: **Qualitative comparison at the ≥13B scale (IV).** Each example shows the source video and editing instruction together with outputs from the four ≥13B baselines and our 14B model.



Figure 11: **Qualitative comparison at the ≤5B scale (I).** Each example shows the source video and editing instruction together with outputs from the three ≤5B baselines and our 5B model.

G.2 Effect of Self-Evaluation Hyperparameters

We further analyze how checkpoint density K (Table 5) and return-to-go discount γ (Table 6) affect editing quality and layer-wise refinement. Each configuration uses one seed; consequently, the zero seed-level standard deviations in the raw logs do not constitute uncertainty estimates.



Figure 12: **Qualitative comparison at the ≤5B scale (II).** Each example shows the source video and editing instruction together with outputs from the three ≤5B baselines and our 5B model.



Figure 13: **Qualitative comparison at the ≤5B scale (III).** Each example shows the source video and editing instruction together with outputs from the three ≤5B baselines and our 5B model.

Increasing K beyond four does not improve the average OpenVE-Bench score (Table 5): $K = 4$ attains the highest mean of 4.20, whereas $K = 6$ and $K = 8$ both fall to 4.17. The two denser settings are also less balanced, with $K = 6$ trading the lowest IC (3.55) for the highest CF and VQ and thus the largest across-dimension spread (0.43). Combined with the approximately linear cost growth of Table 3, this indicates diminishing returns beyond four checkpoints.

For the discount (Table 6), $\gamma = 0.00$ and $\gamma = 0.50$ tie at the highest mean score of 4.20 after rounding. We select $\gamma = 0.50$ because it attains the highest IC (3.86), the strongest late-layer reward (0.73), and a smaller across-dimension



Figure 14: **Qualitative comparison at the ≤5B scale (IV).** Lucy-Edit leaves the target object largely unedited, illustrating the under-editing outlier behavior behind its favorable preservation metrics.

Table 3: **Training cost as a function of the checkpoint density** K, with $\gamma = 0.5$ fixed. Step and reward times are averages per training iteration. Total time is the observed wall-clock time; the $K = 8$ run contains 320 steps and all others contain 300. Peak memory is the maximum allocated GPU memory.

K	Steps	Step time (s)	Reward time (s)	Reward share (%)	Reward evals/iter.	Peak mem. (GiB)	Total time (h)
2	300	519.8	240.1	46.2	48	139.8	43.3
4	300	836.9	413.5	49.4	80	131.7	69.7
6	300	1132.5	587.5	51.9	112	131.7	94.4
8	300	1416.8	658.8	46.5	144	139.8	125.9

variation (0.24 versus 0.28), while still propagating credit from downstream checkpoints. Larger discounts do not help: $\gamma = 0.75$ drops to 4.10 and $\gamma = 1.00$ recovers only to 4.17. Across every configuration in both tables, late rewards exceed early rewards by approximately 0.10–0.16, confirming progressive refinement through backbone depth. Overall, $K = 4, \gamma = 0.5$ offers the best balance between editing quality, stable layer-wise refinement, and computational cost.

G.3 Additional Results on FiVE-Bench

In Sec. 4, we evaluate instruction-level editing quality on OpenVE-Bench using VLM judges. We complement that evaluation here with the reference-free automatic protocol of FiVE-Bench (Li et al., 2025a), which focuses on fine-grained object-level edits. The benchmark covers six edit types: rigid and non-rigid object replacement, color alteration, material modification, object addition, and object removal.

We report six complementary metrics. **CLIPS** measures semantic alignment between the edited video and the target caption, while **CLIPS**$_e$ measures whether the visual change follows the intended edit direction. **MF** measures motion fidelity between the source and edited videos. We additionally report **Struct**, the structural distance between source and edited content, together with **PSNR** and **LPIPS** on the non-edited regions. Higher CLIPS, CLIPS$_e$, MF, and PSNR indicate better performance, whereas lower Struct and LPIPS are preferred. Tables 7 and 8 group three edit types each for compact presentation. Within each model-scale group (≤5B and ≥13B), the best result among non-outlier methods is shown in bold. Lucy-Edit is marked with a dagger because qualitative inspection reveals a degenerate under-editing behavior: it frequently leaves the source nearly unchanged rather than successfully applying the instruction. Its preservation-oriented scores must therefore not be interpreted as evidence of superior editing quality.

Table 4: **Training cost as a function of the return-to-go discount** γ, with $K = 4$ fixed. Columns follow Table 3.

γ	Steps	Step time (s)	Reward time (s)	Reward share (%)	Reward evals/iter.	Peak mem. (GiB)	Total time (h)
0.00	300	822.1	414.3	50.4	80	139.8	68.5
0.25	300	894.9	444.6	49.7	80	131.7	74.6
0.50	300	836.9	413.5	49.4	80	131.7	69.7
0.75	300	892.3	459.3	51.5	80	136.4	74.4
1.00	300	862.5	420.6	48.8	80	135.6	71.9

Table 5: **Quality and layer-reward diagnostics as a function of the checkpoint density** K, with $\gamma = 0.5$ fixed. Avg., IC, CF, and VQ are evaluated on OpenVE-Bench. Dim. std. is the standard deviation across the three quality dimensions, not across random seeds. Early, middle, and late denote rewards aggregated over the corresponding backbone-depth regions. Final gain is the final reward minus the mean intermediate reward.

K	Avg.	IC	CF	VQ	Dim. std.	Early	Middle	Late	Layer avg.	Layer spread	Final gain
2	4.18	3.69	4.44	4.41	0.35	0.64	0.69	**0.74**	**0.69**	0.05	0.10
4	**4.20**	**3.86**	4.39	4.36	0.24	0.61	0.64	0.73	0.66	0.05	0.10
6	4.17	3.55	**4.49**	**4.47**	0.43	0.54	0.59	0.68	0.59	0.06	0.11
8	4.17	3.85	4.35	4.31	**0.23**	0.57	0.62	0.72	0.63	0.06	0.11

Lucy-Edit as an under-editing outlier. Lucy-Edit obtains unusually strong values on several preservation-oriented metrics, including MF, Struct, PSNR, and LPIPS. However, qualitative inspection shows that these scores largely VISTAbecause the method often fails to perform the requested edit and instead produces an output close to the source video. This is a known failure mode of reference-free preservation metrics: doing little or nothing minimizes structural and perceptual change and can therefore appear favorable despite poor instruction compliance. We retain Lucy-Edit in the tables for completeness, but exclude it from best-result highlighting and from consideration as the strongest successfully editing baseline. At the 5B scale, we therefore use Kiwi-Edit as the primary matched comparison.

Comparison at the 5B scale. Compared with the size-matched Kiwi-Edit baseline, VISTA improves 32 of the 36 reported task–metric combinations. On both rigid and non-rigid object replacement, as well as color alteration, it improves all six metrics simultaneously. The gain is particularly clear for non-rigid replacement, where CLIPS_e increases from 18.94 to 19.50, MF increases from 77.14 to 77.85, and LPIPS decreases from 213.75 to 194.86. For material modification, object addition, and object removal, VISTA consistently improves motion and structural preservation, including an MF gain of $+1.28$ for object addition and $+1.10$ for object removal. The few regressions are small and occur mainly in caption-alignment metrics, illustrating the expected trade-off between applying a stronger edit and preserving the source video.

Comparison at the 14B scale. Against SAMA, the strongest size-matched large-scale baseline, VISTA improves 28 of the 36 task–metric combinations. It obtains lower Struct, higher PSNR, and lower LPIPS on all six edit types, and improves MF on five of the six. For example, on object addition it reduces Struct from 23.79 to 20.68 and LPIPS from 137.58 to 133.50, while increasing MF from 96.36 to 96.59. Semantic alignment is more mixed: VISTA improves CLIPS_e for rigid replacement, non-rigid replacement, and object addition, but trails SAMA slightly on several CLIPS measurements. These results indicate that the principal advantage at the larger scale is stronger preservation of scene structure and motion without a systematic loss of editing alignment.

Overall trade-off. No individual automatic metric is sufficient on its own. A method can obtain high preservation scores by under-applying the requested edit, while strong caption alignment may coexist with severe motion degradation. This behavior is visible in the baselines: Lucy-Edit's near-source outputs produce artificially strong preservation scores despite frequent editing failures, whereas UniVideo achieves competitive CLIPS values in some categories while exhibiting substantially lower MF. VISTA provides a more balanced operating point, maintaining competitive semantic alignment while consistently improving structural, perceptual, and motion fidelity. This trend agrees with the VLM-based results in Sec. 4, where the method more faithfully edits the target while preserving untargeted content.

Table 6: **Quality and layer-reward diagnostics as a function of the return-to-go discount** γ, with $K = 4$ fixed. Columns follow Table 5.

γ	Avg.	IC	CF	VQ	Dim. std.	Early	Middle	Late	Layer avg.	Layer spread	Final gain
0.00	**4.20**	3.81	**4.40**	**4.40**	0.28	**0.61**	**0.64**	0.72	0.65	0.05	0.10
0.25	4.17	3.77	4.39	4.35	0.28	0.59	0.63	0.72	0.64	0.05	0.11
0.50	**4.20**	**3.86**	4.39	4.36	0.24	**0.61**	**0.64**	**0.73**	**0.66**	0.05	0.10
0.75	4.10	3.78	4.28	4.24	**0.23**	0.57	0.60	0.69	0.62	0.05	0.10
1.00	4.17	3.79	4.38	4.34	0.27	**0.61**	**0.64**	**0.73**	**0.66**	0.05	0.10

Table 7: **Automatic-metric results on rigid object replacement, non-rigid object replacement, and color alteration.** We compare semantic alignment (CLIPS and CLIPS_e), motion fidelity (MF), and content preservation (Struct, PSNR, and LPIPS). Arrows indicate the preferred direction, and bold denotes the best result among non-outlier methods within each model-scale group. Lucy-Edit† is excluded from best-result highlighting because it is an under-editing outlier, as discussed below.

Method	#P	Rigid replacement						Non-rigid replacement						Color alteration					
		CLIPS ↑	CLIPS_e ↑	MF ↑	Struct ↓	PSNR ↑	LPIPS ↓	CLIPS ↑	CLIPS_e ↑	MF ↑	Struct ↓	PSNR ↑	LPIPS ↓	CLIPS ↑	CLIPS_e ↑	MF ↑	Struct ↓	PSNR ↑	LPIPS ↓
Omni-Video	1.3B	**26.96**	**21.06**	65.14	97.71	13.36	334.97	**26.70**	**20.67**	64.51	101.36	13.35	334.58	27.06	21.88	66.05	90.54	13.81	321.70
Lucy-Edit†	5B	25.03	20.19	81.72	29.77	21.47	152.04	23.53	18.55	80.65	40.34	20.41	165.53	27.39	21.99	83.66	31.85	22.15	154.21
Kiwi-Edit	5B	25.63	20.28	77.86	50.97	17.36	200.75	25.20	18.94	77.14	58.70	16.79	213.75	27.71	22.20	78.92	46.61	18.14	188.12
Ours (5B)	5B	25.75	20.60	**78.56**	**47.88**	**17.94**	**182.89**	25.27	19.50	**77.85**	**56.66**	**17.28**	**194.86**	**27.80**	**22.27**	**79.59**	**43.66**	**18.61**	**172.73**
Ditto	14B	23.52	19.77	82.27	80.34	15.38	248.25	23.09	18.03	79.06	87.45	14.64	258.86	25.34	21.14	80.77	70.43	15.77	235.62
UniVideo	20B	26.77	21.47	43.80	116.20	11.35	448.87	25.50	20.49	45.66	120.34	11.32	446.36	26.63	22.18	44.37	123.27	11.36	444.36
VINO	13B	26.48	21.13	67.04	91.98	14.24	312.47	26.12	19.73	67.64	111.51	13.69	324.61	27.88	**22.54**	69.95	71.16	16.21	258.54
SAMA	14B	**27.39**	22.07	87.57	37.91	19.20	149.04	27.30	21.33	**87.27**	46.00	18.89	147.21	**28.00**	22.33	87.52	34.20	19.76	148.51
Ours (14B)	14B	27.32	**22.23**	**87.58**	**37.13**	**19.54**	**144.16**	**27.39**	**21.36**	86.64	**45.62**	**19.08**	**143.92**	27.96	22.30	**87.84**	**32.90**	**20.05**	**144.17**

G.4 Additional Results on OpenVE-Bench

Cross-judge comparison. The four VLM judges differ substantially in score calibration, making a direct average of their raw scores potentially misleading. We therefore visualize both the reported scores and each method's rank within each judge in Figure 15. The two VISTA variants obtain the lowest mean rank of 1.88, ahead of SAMA (2.25) and Kiwi-Edit (4.00). The 5B variant ranks first under Gemini-2.5-Pro and Qwen3.5-397B-A17B, while the 14B variant ranks first under Gemini-3.1-Pro and Qwen3.5-35B-A3B. Thus, the overall conclusion is stable even though the judges assign different absolute scores.

Category-level gains. Figure 16 compares each VISTA variant with its size-matched baseline: the 5B model against Kiwi-Edit and the 14B model against SAMA. Each category score is the mean of IC, CF, and VQ, and all differences are computed within the same judge. The 5B model shows especially consistent gains on Local Remove and Local Change, including Local Remove improvements of $+0.35$, $+0.55$, and $+0.53$ under the first three judges. Global Style is less consistent, which agrees with our observation that token-localized guidance is most effective when the instruction targets a specific object or region. The 14B comparison is more evaluator-dependent: it improves Local

Table 8: **Automatic-metric results on material modification, object addition, and object removal.** We compare semantic alignment (CLIPS and CLIPS_e), motion fidelity (MF), and content preservation (Struct, PSNR, and LPIPS). Arrows indicate the preferred direction, and bold denotes the best result among non-outlier methods within each model-scale group. Lucy-Edit† is excluded from best-result highlighting because it is an under-editing outlier, as discussed below.

Method	#P	Material modification						Object addition						Object removal					
		CLIPS ↑	CLIPS_e ↑	MF ↑	Struct ↓	PSNR ↑	LPIPS ↓	CLIPS ↑	CLIPS_e ↑	MF ↑	Struct ↓	PSNR ↑	LPIPS ↓	CLIPS ↑	CLIPS_e ↑	MF ↑	Struct ↓	PSNR ↑	LPIPS ↓
Omni-Video	1.3B	26.77	21.29	66.32	90.94	13.63	330.94	25.44	21.69	68.19	74.90	14.26	306.46	**23.93**	19.73	53.82	82.59	14.17	279.65
Lucy-Edit†	5B	26.46	21.48	82.66	30.13	22.09	145.27	23.26	20.95	94.93	17.68	22.42	132.62	23.85	20.21	80.26	14.73	25.01	99.27
Kiwi-Edit	5B	**26.94**	21.54	78.08	47.48	17.71	195.82	**25.64**	**21.82**	91.04	33.37	18.69	154.63	23.54	**20.89**	77.03	38.91	19.80	149.79
Ours (5B)	5B	26.80	**21.58**	**79.09**	**43.57**	**18.09**	**177.39**	25.51	21.79	**92.32**	**32.13**	**18.83**	**151.20**	23.79	20.76	**78.13**	**38.12**	**19.94**	**142.60**
Ditto	14B	25.55	21.28	83.81	61.20	16.61	224.58	22.19	19.34	92.89	61.79	16.50	235.72	21.26	18.48	76.08	76.00	18.27	188.10
UniVideo	20B	26.24	21.31	45.65	113.86	11.41	446.39	25.60	**22.98**	43.66	119.14	11.54	433.43	**24.08**	20.33	45.78	109.42	12.13	397.92
VINO	13B	26.39	20.67	63.73	103.64	13.05	352.48	**25.71**	21.91	80.65	55.01	16.19	251.78	23.99	20.20	67.95	46.17	18.89	189.10
SAMA	14B	27.11	**21.60**	87.01	35.03	19.52	149.37	25.03	20.94	96.36	23.79	19.80	137.58	24.01	**20.89**	83.49	23.00	20.69	120.97
Ours (14B)	14B	**27.17**	21.55	**87.44**	**34.04**	**19.77**	**145.21**	24.82	21.13	**96.59**	**20.68**	**20.07**	**133.50**	23.87	20.41	**83.74**	**22.09**	**20.85**	**120.23**

Table 9: **Comparison on the OpenVE video-editing benchmark, judged by GEMINI-2.5-PRO.** We report the average score across all edit categories (**Avg. Score**) together with per-category scores along three complementary axes: Instruction Compliance (IC), Consistency & Detail Fidelity (CF), and Visual Quality & Stability (VQ). All scores are on a 1–5 scale; higher is better. Our method attains the highest average score while using substantially fewer parameters than competing baselines.

Method	#Params	Avg. Score	Local Remove			Global Style			Background Change			Local Change			Local Add		
			IC	CF	VQ	IC	CF	VQ	IC	CF	VQ	IC	CF	VQ	IC	CF	VQ
Small-scale models (≤5B)																	
Omni-Video	1.3B	1.59	1.37	1.10	1.20	2.02	1.79	1.91	1.42	1.36	1.36	2.09	1.65	1.58	1.81	1.42	1.70
Lucy-Edit	5B	2.13	1.17	1.08	1.10	1.62	1.57	1.59	2.71	2.24	2.20	3.85	3.52	3.40	2.37	1.49	2.01
Kiwi-Edit	5B	3.16	3.31	2.66	2.73	3.74	3.52	3.64	3.02	2.76	2.63	4.06	3.80	3.77	2.93	2.22	2.64
Ours	5B	**3.31**	3.59	3.05	3.10	3.64	3.52	3.57	3.12	2.71	2.51	4.40	4.14	4.11	3.06	2.40	2.81
Large-scale models (≥13B)																	
Ditto	14B	1.80	1.00	1.00	1.00	3.83	3.21	3.71	1.59	1.54	1.49	1.58	1.49	1.46	1.48	1.22	1.39
UniVideo	20B	1.39	1.14	1.02	1.08	2.31	1.84	2.28	1.36	1.31	1.24	1.35	1.18	1.15	1.25	1.12	1.21
VINO	13B	2.48	2.31	1.98	2.10	3.62	3.48	3.57	2.07	1.73	1.53	3.11	2.85	2.74	2.42	1.61	2.13
SAMA	14B	3.27	3.61	2.76	2.92	3.91	3.71	3.83	3.03	2.81	2.46	4.28	4.02	3.98	2.93	2.18	2.66
Ours (14B)	14B	3.22	3.44	2.63	2.81	3.93	3.76	3.90	3.14	2.81	2.51	4.14	3.78	3.71	2.87	2.12	2.69

Table 10: **Comparison on the OpenVE video-editing benchmark, judged by QWEN3.5-397B-A17B.** We report the average score across all edit categories (**Avg. Score**) together with per-category scores along three complementary axes: Instruction Compliance (IC), Consistency & Detail Fidelity (CF), and Visual Quality & Stability (VQ). All scores are on a 1–5 scale; higher is better. Our method attains the highest average score while using substantially fewer parameters than competing baselines.

Method	#Params	Avg. Score	Local Remove			Global Style			Background Change			Local Change			Local Add		
			IC	CF	VQ	IC	CF	VQ	IC	CF	VQ	IC	CF	VQ	IC	CF	VQ
Small-scale models (≤5B)																	
Omni-Video	1.3B	2.70	1.42	1.56	1.63	2.81	3.28	3.86	2.88	2.76	2.93	2.72	2.92	3.09	2.39	2.81	3.37
Lucy-Edit	5B	3.20	1.05	1.90	1.90	1.62	4.19	4.26	3.97	3.69	3.76	4.11	4.25	4.23	2.51	2.96	3.58
Kiwi-Edit	5B	3.87	3.22	3.12	3.17	3.95	4.48	4.60	4.14	3.93	4.00	4.51	4.34	4.35	3.21	3.25	3.81
Ours	5B	**4.05**	3.85	3.63	3.69	4.02	4.48	4.53	4.03	4.00	3.93	4.55	4.46	4.48	3.43	3.60	4.04
Large-scale models (≥13B)																	
Ditto	14B	2.71	1.07	1.38	1.48	3.79	3.78	4.41	3.45	3.39	3.45	2.24	2.39	2.71	2.00	2.28	2.81
UniVideo	20B	2.53	1.15	1.17	1.24	3.00	3.19	4.16	2.66	2.69	2.71	2.26	2.48	2.57	2.33	2.85	3.43
VINO	13B	3.39	2.48	2.58	2.62	3.97	4.16	4.55	3.34	3.41	3.33	3.48	3.67	3.69	2.61	3.15	3.82
SAMA	14B	3.99	3.64	3.63	3.69	3.97	4.72	4.78	4.08	4.03	3.98	4.46	4.46	4.45	2.82	3.22	3.91
Ours (14B)	14B	3.99	3.75	3.86	3.88	3.81	4.50	4.72	4.05	4.10	4.02	4.31	4.49	4.49	2.78	3.15	3.87

Table 11: **Comparison on the OpenVE-Bench video-editing benchmark, judged by GEMINI-3.1-PRO.** We report the average score (**Avg. Score**) and per-category scores along Instruction Compliance (IC), Consistency & Detail Fidelity (CF), and Visual Quality & Stability (VQ). All scores are on a 1–5 scale; higher is better.

Method	#Params	Avg. Score	Local Remove			Global Style			Background Change			Local Change			Local Add		
			IC	CF	VQ	IC	CF	VQ	IC	CF	VQ	IC	CF	VQ	IC	CF	VQ
Small-scale models (≤5B)																	
Omni-Video	1.3B	1.66	1.29	1.08	1.14	2.88	1.98	2.47	1.12	1.10	1.08	2.42	1.63	1.60	2.10	1.36	1.70
Lucy-Edit	5B	1.88	1.02	1.00	1.00	1.59	1.52	1.59	2.37	1.88	1.76	3.58	2.77	2.62	2.31	1.45	1.76
Kiwi-Edit	5B	2.94	2.92	2.12	2.27	4.09	3.76	3.97	2.90	2.68	2.47	4.09	3.17	3.28	2.66	1.69	2.00
Ours (5B)	5B	**3.09**	3.63	2.53	2.73	3.90	3.53	3.64	3.00	2.63	2.54	4.20	3.31	3.35	2.87	2.09	2.34
Large-scale models (≥13B)																	
Ditto	14B	1.86	1.00	1.00	1.00	4.40	3.00	4.07	1.39	1.31	1.34	1.77	1.43	1.55	1.70	1.34	1.60
UniVideo	20B	1.71	1.15	1.07	1.12	3.81	2.14	3.71	1.03	1.03	1.03	1.72	1.35	1.51	1.79	1.37	1.78
VINO	13B	2.51	2.29	2.02	2.14	4.29	3.52	4.17	1.64	1.51	1.39	3.09	2.80	2.77	2.42	1.63	2.04
SAMA	14B	3.31	3.90	2.97	3.20	4.12	3.72	3.93	3.12	3.07	2.59	4.25	3.74	3.66	2.93	1.97	2.51
Ours (14B)	14B	**3.33**	3.93	3.20	3.34	4.21	3.83	4.07	3.08	2.93	2.54	4.17	3.52	3.40	3.07	2.09	2.54

Table 12: **Comparison on the OpenVE-Bench video-editing benchmark, judged by QWEN3.5-35B-A3B.** As an independent-judge counterpart to Table 1, we report the average score (**Avg. Score**) and per-category scores along Instruction Compliance (IC), Consistency & Detail Fidelity (CF), and Visual Quality & Stability (VQ). All scores are on a 1–5 scale; higher is better. The ranking is largely consistent across judges, confirming that our method is competitive with substantially larger baselines.

Method	#Params	Avg. Score	Local Remove			Global Style			Background Change			Local Change			Local Add		
			IC	CF	VQ	IC	CF	VQ	IC	CF	VQ	IC	CF	VQ	IC	CF	VQ
Small-scale models (≤5B)																	
Omni-Video	1.3B	3.34	3.18	3.63	3.68	3.57	3.71	3.91	3.29	3.58	3.51	3.28	3.42	3.48	3.39	3.37	3.93
Lucy-Edit	5B	3.74	3.80	4.76	4.80	3.73	4.72	4.59	3.55	3.66	3.32	4.26	4.25	4.32	3.38	3.24	4.07
Kiwi-Edit	5B	3.93	3.77	3.86	3.90	4.32	4.59	4.40	3.74	3.98	3.68	4.21	4.09	4.23	3.61	3.46	4.12
Ours (5B)	5B	4.00	3.88	3.80	3.83	4.28	4.47	4.29	3.89	4.17	3.80	4.27	4.22	4.26	3.68	3.49	4.21
Large-scale models (≥13B)																	
Ditto	14B	3.33	3.47	4.36	4.47	4.03	3.91	4.28	3.36	3.61	3.54	2.94	3.02	3.31	2.85	2.61	3.43
UniVideo	20B	3.37	3.44	4.07	4.24	3.78	3.52	4.19	3.04	3.19	3.17	3.48	3.57	3.86	3.13	2.91	3.63
VINO	13B	3.81	4.00	4.44	4.47	4.44	4.43	4.36	3.47	3.63	3.56	3.81	3.82	4.05	3.35	3.06	3.97
SAMA	14B	4.00	4.14	4.15	4.15	4.31	4.59	4.31	3.69	3.90	3.59	4.31	4.20	4.29	3.58	3.36	3.96
Ours (14B)	14B	4.06	4.20	4.12	4.08	4.10	4.59	4.43	3.49	4.00	3.64	4.46	4.46	4.55	3.25	3.37	4.18

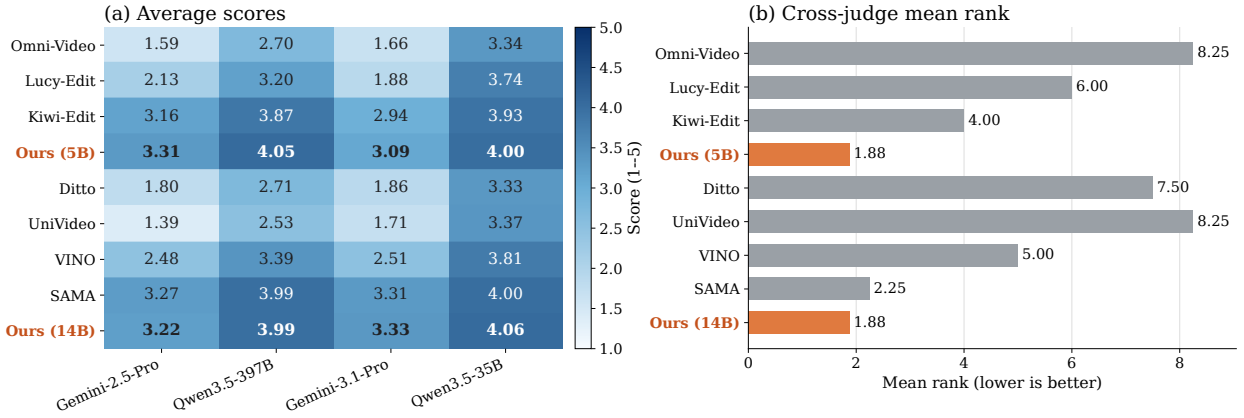


Figure 15: **Cross-model robustness across four VLM judges. Left:** Average OpenVE-Bench scores; darker cells indicate higher scores. **Right:** Mean rank across the four judges, where lower is better. Ranking within each judge avoids conflating their different score calibrations. Both the 5B and 14B VISTA variants achieve a mean rank of 1.88.

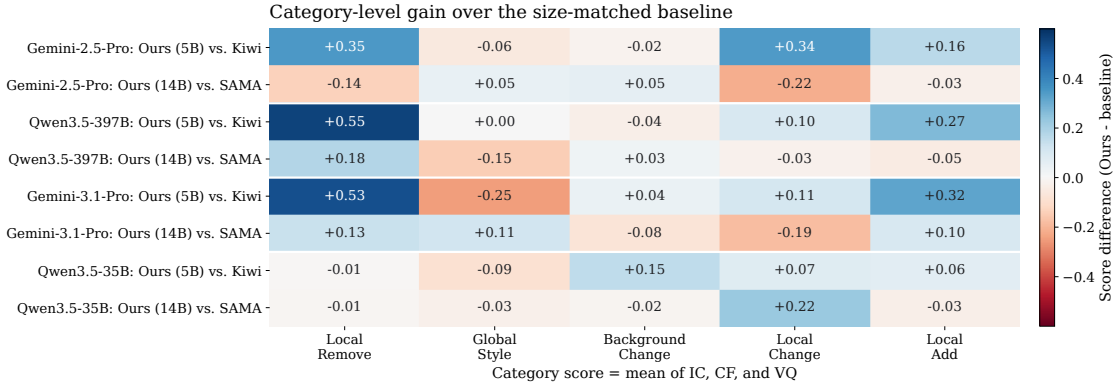


Figure 16: **Category-level score differences against size-matched baselines.** Each cell reports the difference between VISTA and the corresponding baseline after averaging IC, CF, and VQ within a category. The 5B model is compared with Kiwi-Edit and the 14B model with SAMA. Blue cells indicate gains and red cells indicate regressions.

Change by +0.22 under QWEN3.5-35B-A3B, while some judges prefer SAMA on that category. This variation motivates reporting multiple independent judges rather than relying on a single evaluator.

G.5 HUMAN PREFERENCE STUDY

The evaluations above rely on VLM judges, which share architectural and training biases with the reward model used during post-training. To check that the reported ordering also holds for human viewers, we conduct a preference study on a subset of OpenVE-Bench.

Protocol. We sample 100 evaluation cases, 20 from each of the five OpenVE-Bench categories used in Sec. 4, and collect outputs for our 5B model and five baselines spanning both capacity groups: Omni-Video, Lucy-Edit, Kiwi-Edit, VINO, and SAMA. All methods use the identical source videos, instructions, and sampling settings as in Table 1. Twenty-four participants took part, twelve of whom reported prior video editing or VFX experience. Method identities were hidden, and the presentation order of both clips and methods was randomized per participant.

Each participant completed two tasks. In the *rating* task, a single edited video is shown next to its source video and instruction, and the participant scores it on the same three axes used by the VLM judges, Instruction Compliance (IC), Consistency & Detail Fidelity (CF), and Visual Quality & Stability (VQ), on a 1–5 scale with the anchor descriptions of Appendix **??** rewritten in plain language. In the *pairwise* task, our output and one baseline output are shown side by side in random left–right order, and the participant selects the better edit or declares a tie. Every clip–method pair received three independent ratings and every clip–baseline pair three independent comparisons, giving 1,800 ratings and 1,500 comparisons in total. We interleaved attention checks in which the unedited source was presented as an edit; two participants who passed fewer than half of these checks were replaced before analysis.

Table 13: **Human ratings on a 100-clip OpenVE-Bench subset.** Mean participant scores on the same three axes used by the VLM judges (1–5; higher is better), with 95% confidence intervals on the average over the three axes. Human raters are stricter in absolute terms than GEMINI-2.5-PRO, but produce the same ordering.

Method	#Params	IC	CF	VQ	Avg.
Omni-Video	1.3B	1.61	1.39	1.44	1.48 ±0.07
Lucy-Edit	5B	2.18	1.96	2.07	2.07 ±0.09
VINO	13B	2.51	2.22	2.29	2.34 ±0.09
Kiwi-Edit	5B	3.22	2.83	2.92	2.99 ±0.08
SAMA	14B	3.39	2.94	3.00	3.11 ±0.08
Ours	5B	**3.44**	**3.14**	**3.17**	**3.25** ±0.08

Table 14: **Pairwise human preference for our 5B model against each baseline.** Percentages over 300 comparisons per pair. **Net** is the win rate minus the loss rate; p-values are from a two-sided binomial test on non-tied comparisons.

Ours vs.	#Params	Win (%)	Tie (%)	Loss (%)	Net
Omni-Video	1.3B	86.4	7.1	6.5	+79.9
Lucy-Edit	5B	74.9	10.8	14.3	+60.6
VINO	13B	68.1	13.2	18.7	+49.4
Kiwi-Edit	5B	52.7	21.4	25.9	+26.8
SAMA	14B	45.2	23.1	31.7	+13.5

Results. Table 13 shows that participants reproduce the method ordering of the VLM judges, with our 5B model rated highest on all three axes. Absolute scores are lower than those of GEMINI-2.5-PRO across the board, so the study is informative about ordering rather than calibration. The margin over Kiwi-Edit, the checkpoint our model is initialized from, is $+0.26$ average, and the margin over SAMA is $+0.14$ at roughly a third of its parameters. As in the VLM evaluation, the gain is larger on the two preservation axes (CF $+0.31$, VQ $+0.25$ over Kiwi-Edit) than on instruction compliance (IC $+0.22$), consistent with the interpretation in Sec. 4 that conditioning later layers on the named target mainly reduces over-editing of untargeted content.

Table 14 reports the pairwise task. Our model is preferred over every baseline, with net preference decreasing monotonically with baseline strength. Restricted to non-tied comparisons, the preference is statistically significant against both Kiwi-Edit (67.1%, $p < 0.001$) and SAMA (58.8%, $p < 0.01$). The tie rate grows from 7.1% against Omni-Video to 23.1% against SAMA, indicating that the remaining differences at the top of the ranking are genuinely narrow rather than being obscured by rater noise.

H FULL VISUAL COMPARISONS OF MAIN RESULTS



Figure 17: **Additional qualitative comparison (I).** Under identical inputs and sampling settings, VISTA maintains stronger instruction compliance and temporal consistency, whereas competing methods more often under-edit the target or introduce unintended changes to the surrounding scene.

Figures 17–19 provide additional qualitative comparisons across diverse instruction-guided video editing examples. For each example, all methods receive the same source video and editing instruction and are evaluated under identical inference settings. Across the comparisons, VISTA more reliably applies the requested edit while preserving untargeted objects, scene structure, and source motion. Competing methods more frequently under-apply the instruction, modify regions unrelated to the requested edit, or introduce appearance and temporal inconsistencies. These examples complement the quantitative results in Sec. 4 and illustrate the practical benefits of intermediate self-evaluation and token-importance memory.



Figure 18: **Additional qualitative comparison (II).** Further examples demonstrate that VISTA achieves a better balance between applying the desired edit and preserving the source video's identity, fine-grained details, and visual stability.

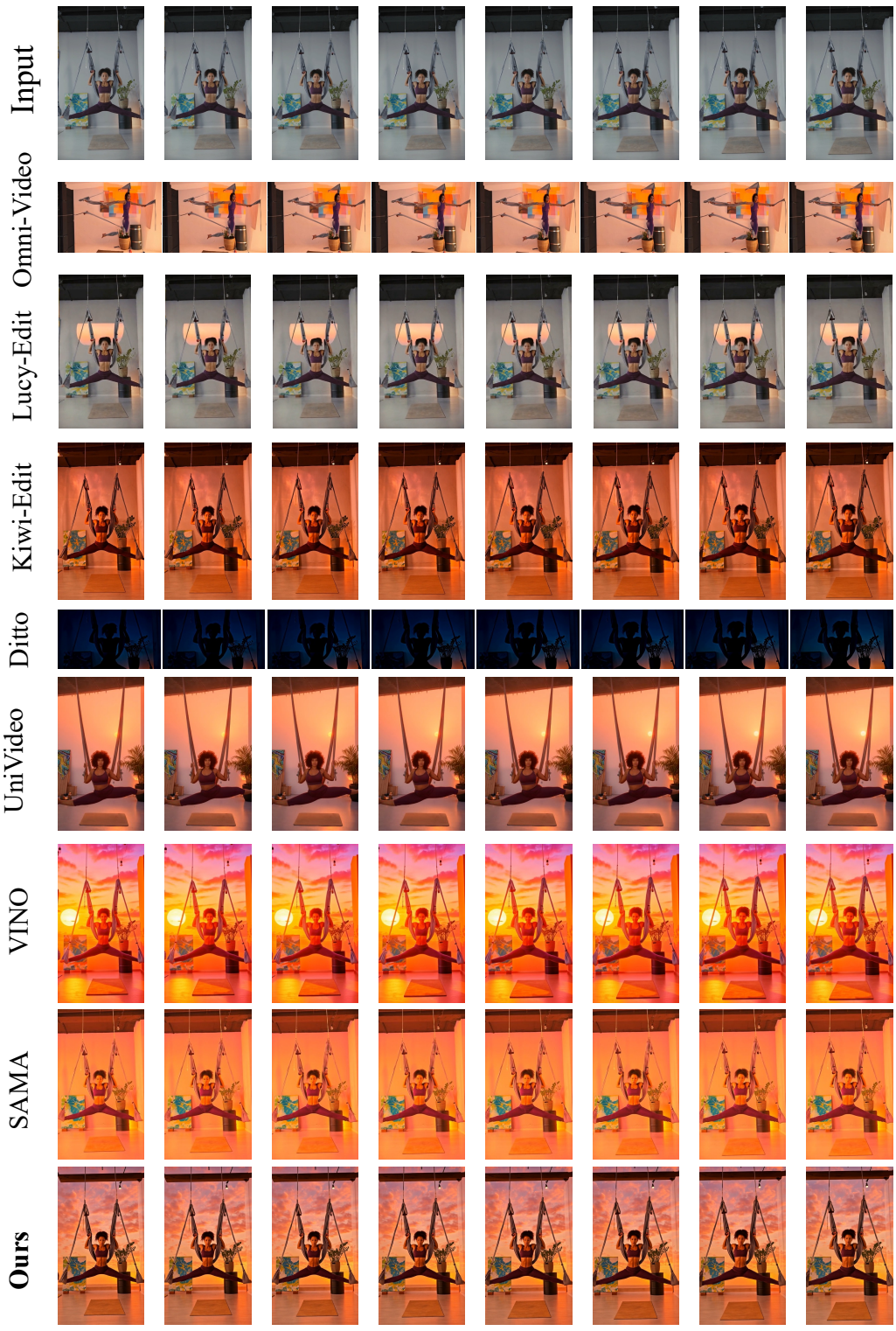


Figure 19: **Additional qualitative comparison (III).** Each example shows the source video and editing instruction together with outputs from the compared methods. VISTA performs the requested edit more faithfully while better preserving content and motion outside the target region.