# Long Video Instructional Editing in the Wild

**Yiqi Lin**[1,2] **Guoqiang Liang**[1,2] **Ziyun Zeng**[1] **Zechen Bai**[1] **Yanzhe Chen**[1] **Mike Zheng Shou**[1]✉
[1]Show Lab, National University of Singapore
[2]National University of Singapore Guangzhou Research Translation and Innovation Institute, Guangzhou, Guangdong 510700, China

## Abstract

Instructional video editing has advanced rapidly for short, single-shot clips, yet long videos, the dominant form of video in practice, remain largely unaddressed. Edit targets recur across distant shots with varying viewpoints, scenes, and scales, requiring consistent edits across shots. Non-target intervals must remain unchanged, which requires accurate temporal grounding of the instruction. Our hierarchical framework uses a *planning agent* to ground the instruction and partition the timeline into shot-aligned segments. A *long-term anchor bank* uses a strong image editing prior and critic-driven refinement to maintain sparse, high-quality cross-shot anchors. An *anchor-conditioned video editor* propagates these edits; it is adapted from a pretrained short-video editor through LoRA, without architectural changes. The editor is trained exclusively on short-video pairs and applied to long videos at inference through its conditioning interface, requiring no long-video supervision. We introduce **LVEdit-Bench**, the first benchmark designed for long-video instructional editing in the wild. Its three-tier evaluation protocol uses a structured VLM judge to isolate per-shot fidelity, cross-shot identity preservation, and robustness to target-absent content. Across all three settings, our method matches strong short-video editors on per-segment quality and substantially improves cross-shot consistency and grounding as temporal complexity grows. Our model and benchmark are available at https://github.com/showlab/LVEdit.

## 1 Introduction

Recent progress in instruction-based video editing [1, 2, 3, 4] has produced increasingly capable systems [5, 6, 7, 8, 9, 10] for short, single-shot clips. Real-world videos often span minutes, contain multiple shots, and follow recurring subjects across changes in viewpoint and scale. Editing requests range from recoloring a recurring car to restyling an entire episode. Applied to such inputs, even state-of-the-art short-video editors [9, 10] exhibit cross-shot identity drift, edit leakage into non-target segments, and globally incoherent concatenation. As Figure 1 illustrates, long videos differ from short clips along three dimensions that current systems are not designed to handle: *temporally sparse targets, multi-shot structure with cuts and viewpoint shifts, and long-range consistency*. Long-video editing requires more than longer inputs.

We argue that this problem decomposes along two distinct axes. *Long-range semantic recurrence* requires that edits on temporally distant segments depicting the same target agree in identity, attributes, and style. *Short-range temporal continuity* requires reliable and smooth edits within local frame windows. Most existing methods [5, 6, 7, 8, 9, 10] address the second axis but have no mechanism for the first, since cross-shot consistency cannot emerge from a model that processes each segment in isolation. Nor can it be obtained by scaling such methods to longer contexts: long-video editing supervision is essentially unavailable and long-context video diffusion is prohibitively expensive [11]. The two axes therefore call for different mechanisms operating on different temporal horizons.

✉Corresponding author.

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

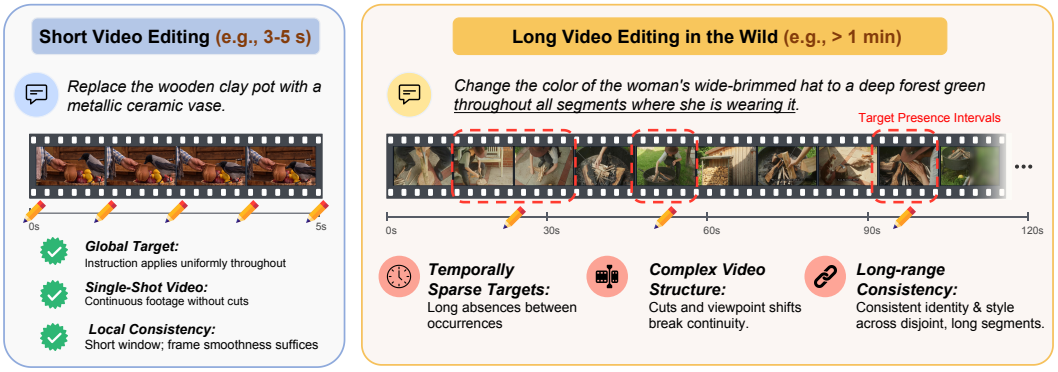



Figure 1: **Short-video editing vs. long-video editing in the wild.** Standard video editing methods target short, single-shot clips with explicit, globally applicable instructions, where local frame smoothness suffices for temporal consistency. Real-world long videos differ along three dimensions that short-video benchmarks do not test. Edit targets appear sparsely along the timeline (red dashed boxes mark target presence intervals). Shot cuts and viewpoint shifts break temporal continuity. Edits must also remain consistent across long, disjoint segments containing the same target.

We propose a framework with components tailored to these two axes. A vision-language *planning agent* [12] grounds the instruction, partitions the timeline into shot-aligned segments, and selects a sparse set of representative anchors. A *long-term anchor bank* edits these anchors using a strong image editing prior [13] under a critic-driven iterative procedure. An *anchor-conditioned video editor*, adapted from a state-of-the-art short-video editor [10] via LoRA fine-tuning [14], propagates these anchors to dense frames: the anchor supplies appearance; the source segment drives motion. Training and inference share the same conditioning interface. The editor learns appearance conditioning from a short context video during training. At inference, it uses the assigned edited anchor as context, while the planning agent routes the full timeline. This avoids the need for long-video supervision when training the editor.

A second obstacle is that existing benchmarks [15, 7] target short clips and do not capture the failure modes specific to long-video editing. We introduce **LVEdit-Bench**, the first benchmark designed for long-video instructional editing in the wild. It pairs manually authored prompts with multi-shot videos. Temporal grounding is initialized by a VLM [12] and refined by human annotators. Short-video baselines cannot ingest full-length inputs, so we compare methods under three settings of increasing temporal complexity: *single-shot, cross-shot with target persistence, and cross-shot with target-absent intervals*. These isolate per-shot fidelity, cross-shot identity preservation, and robustness to irrelevant content, respectively.

Our contributions are as follows. **(i)** We formulate long-video instructional editing along two temporal axes and build a framework that addresses both. A critic-driven anchor bank and an anchor-conditioned editor with a shared training-inference interface extend a model trained on short videos to long videos without long-video supervision. **(ii)** We introduce **LVEdit-Bench**, the first benchmark for long-video instructional editing in the wild, with a three-tier protocol isolating per-shot fidelity, cross-shot identity, and robustness to irrelevant content. **(iii)** We show that our method matches strong short-video baselines [6, 9, 10] on per-segment quality and substantially improves cross-shot consistency. The margin widens as temporal complexity grows.

## 2 Related Work

**Instructional Video Editing.** Text-guided video editing extends image diffusion to the temporal domain through latent sharing, attention manipulation, or motion priors [16, 17, 18, 19]. Inversion- and prompt-based variants enable localized attribute or object-level edits [20, 21]. Recent instruction-driven editors incorporate vision-language models for semantically grounded edits and unify attribute change, object insertion or removal, and global stylization in a single framework [22, 8, 9, 6, 23, 10, 5]. These methods operate within their design regime: short, single-shot clips of a few seconds, where instruction grounding is a single clip-wise decision and temporal consistency reduces to local frame

smoothness. They lack mechanisms for reasoning about *where* along a long timeline an edit applies, or for keeping the same target consistent across distant segments.

**Long-form Video Modeling and Editing.** Long-video understanding has explored hierarchical architectures, temporal abstraction, and memory or retrieval mechanisms for extended context [24, 25, 26, 27]. In editing, sliding-window or chunk-based strategies extend short-clip models to longer inputs, but cannot enforce consistency across segments separated by long gaps and exhibit identity drift after shot changes. Recent training-free pipelines target minute-level editing through cross-segment keyframe blending [28] or velocity-field and attention-sink anchoring [29]. These pipelines assume visually continuous footage. Our setting instead involves multiple shots and recurring targets, as found in long videos in the wild. Evaluation is similarly underdeveloped: existing benchmarks [15, 7] target either short clips or continuous long footage, and do not annotate cross-shot consistency groups, target sparsity, or multi-shot structure. We address both gaps with a hierarchical framework that solves cross-shot consistency at the anchor level and local stability at the dense-frame level, together with LVEdit-Bench for evaluating consistency, sparsity, and recurrence in video editing.

# 3 Method

## 3.1 Overview

We address long-video instructional editing by decomposing the task along its two principal axes: *long-range semantic recurrence* across temporally distant segments, and *short-range temporal continuity* within local windows. Given an input video $V = \{x_t\}_{t=1}^{T}$ and a natural-language instruction $p$, our framework produces the edited video $\hat{V} = \{\hat{x}_t\}_{t=1}^{T}$. The length $T$ may exceed standard editors' temporal context. As shown in Figure 2, a *planning agent* grounds the instruction and produces an editing plan. A *long-term anchor bank* maintains sparse, coherent anchor edits across shots, and *local propagation* transfers these edits to dense frames. The hierarchical design is based on two observations: temporal consistency within individual segments does not ensure consistent target identity across shots, and image editing priors offer substantially better instruction-following fidelity than current video editing models. We handle long-range semantics at the anchor level and short-range stability through dense-frame propagation.

## 3.2 Planning Agent

The planning agent $\Phi$ partitions a long video $V$ into editable segments and selects sparse anchors:

$$(\mathcal{S}, \Pi, A) = \Phi(V, p), \tag{1}$$

where $\mathcal{S} = \{S_n\}_{n=1}^{N}$ is the set of editable segments, $\Pi = \{\pi_n\}_{n=1}^{N}$ stores per-segment edit specifications (localized prompt, target visibility, applicability), and $A = \{a_n\}_{n=1}^{N}$ contains one representative anchor frame per segment. Shot detection provides candidate boundaries. A VLM classifies frames and segments to identify where the edit applies. We exclude non-applicable segments and split overlong editable segments to fit the propagation model's reliable temporal range. For each finalized segment, the anchor $a_n$ is selected as the frame in which the target is most visible and representative.

## 3.3 Long-Term Anchor Bank

**Bank Construction.** The anchor bank produces an edited set $Y = \{y_n\}_{n=1}^{N}$ that remains consistent across all shots, so that propagating from these anchors yields a coherent edit throughout the video. We build $Y$ with an image editing prior $G$ and judge it with a multimodal critic $J$. We seed the bank by editing the anchor with the strongest target visibility, the easiest case for $G$, without any reference. This first edit becomes the visual template: when editing the remaining anchors, we feed already-edited anchors into $G$ as references so the new edits inherit the same identity and style. After this seed pass, every anchor has an edited version, but quality is uneven across the bank.

**Iterative Refinement.** We refine the bank over multiple rounds, each driven by the critic $J$. For every edited anchor, $J$ returns an overall quality $q_n^{(t)} \in [0, 1]$ and a per-axis vector $\mathbf{s}_n^{(t)} = (s_{\text{pres}}, s_{\text{fid}}, s_{\text{bg}}, s_{\text{art}}, s_{\text{id}})$ covering target presence, reference fidelity, background preservation, artifact-free quality, and identity consistency. Each round freezes a bank snapshot $Y^{(t)}$ and draws references only from it. Updates within the round are therefore independent and parallelizable.

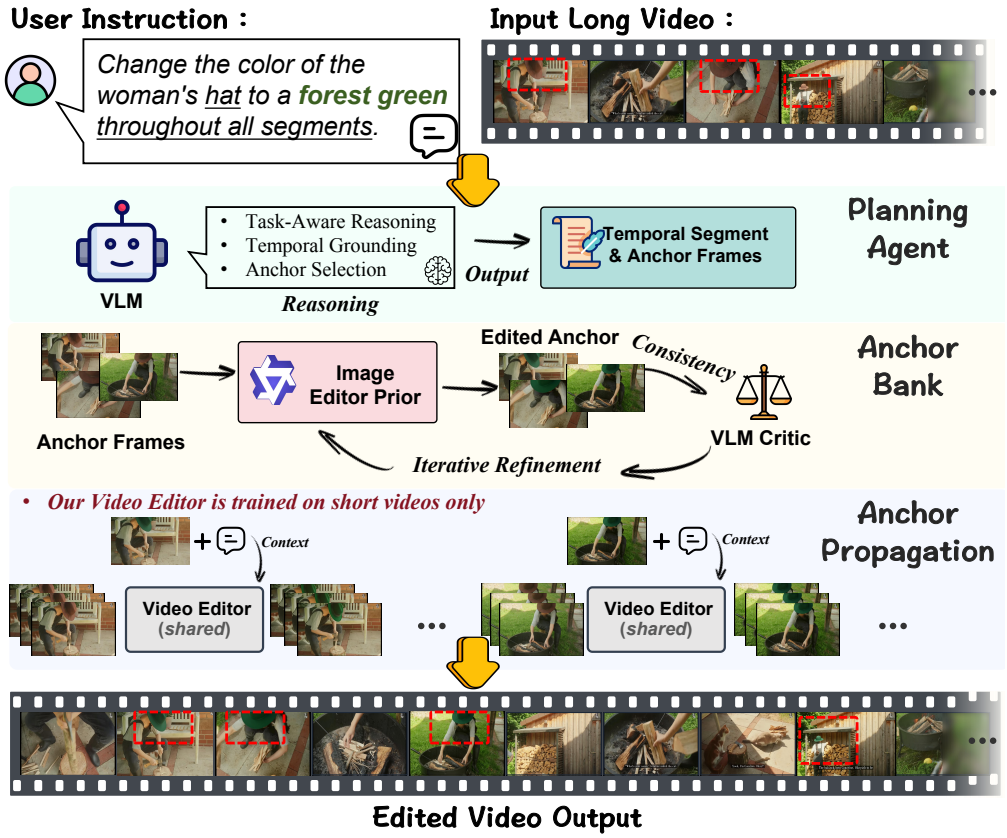



Figure 2: **Overall framework.** Given a long video and an instruction, the *Planning Agent* performs task-aware reasoning, temporal grounding, and anchor selection to produce shot-aligned segments and a sparse anchor frame set (red dashed boxes mark target occurrences). The *Anchor Bank* edits each anchor with a strong image editing prior under critic-driven iterative refinement, producing a refined set of high-quality edited anchors that capture cross-shot consistency. The *Anchor Propagation* module propagates anchor edits to dense frames. Its shared video editor is **trained exclusively on short videos and applied to long videos through the anchor-conditioned interface**. Each segment uses its assigned anchor as the appearance condition.

Every update re-edits the *original source frame* rather than the previous edit to avoid compounding errors. We select the reference set $\mathcal{R}_n^{(t)} \subset Y^{(t)}$ using target visibility and an affinity score over lighting, viewpoint, layout, pose, and scale. This lets anchors edited under similar conditions guide each other. Anchors are handled according to error severity: reliable anchors are retained; compromised anchors use a new reference set and seed; near-correct anchors retain the reference set and use a new seed. An update is accepted only if it does not decrease the worst-case critic score. Refinement stops when all anchors are reliable or after $T_{\max}$ rounds, yielding $Y^*$.

### 3.4 Anchor Propagation via Efficient Adaptation

Rather than train a propagation model from scratch, we build $G_{\text{prop}}$ on top of a pretrained instruction-driven video editor [10], which already provides strong editing priors for instruction following, target reconstruction, and temporal smoothness. The base editor passes information into the DiT through an MLLM-produced query embedding. We retain this query pathway unchanged but inject the context video frames as additional content alongside the source segment, allowing the query to encode appearance from $C_n$ in addition to the instruction. We add LoRA adapters to the DiT to incorporate this conditioning; the MLLM, query interface, and DiT backbone remain frozen.

**Training.** For each segment $S_n$, the model receives noise tokens $\epsilon_n$ together with an MLLM query embedding produced from the instruction $p$, the source $S_n$, and the context video $C_n$. The DiT denoises $\epsilon_n$ under this query. At its input, $C_n$ is also sequence-padded and concatenated with $\epsilon_n$ along the temporal axis. Training tuples $(S_n, \tilde{S}_n, C_n)$ are drawn from real source/target editing pairs, where $C_n$ contains the preceding target frames used as appearance context. Only the LoRA adapters are trained with a flow-matching objective. We aim to emphasize appearance conditioning over motion modeling.

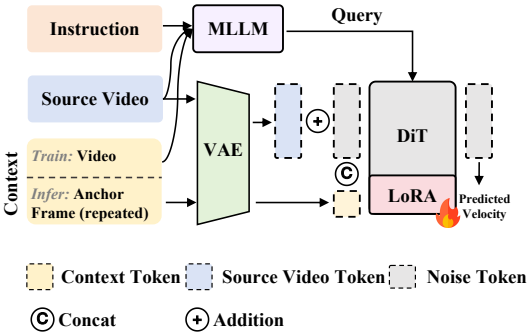


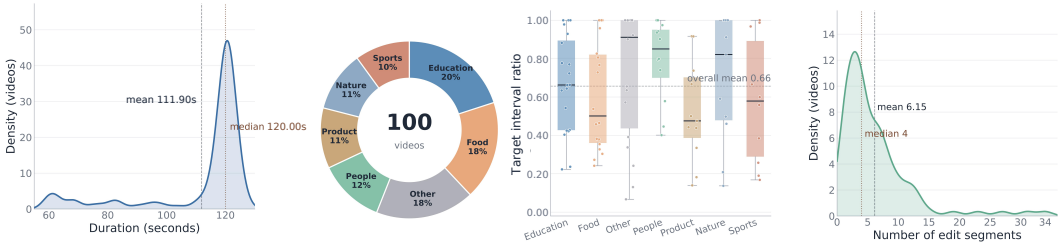

Figure 3: **Anchor-conditioned video editor.** Training and inference share the same conditioning interface: a context video supplies appearance during training, while a temporally repeated anchor occupies the same slot at inference.

**Inference.** At inference, no real context video is available; the editor is conditioned on the edited anchor bank $Y^*$. For segment $S_n$, the planning assignment $\pi_n$ selects the relevant anchor $y^*_n$. We repeat this anchor to match the training context length, $C_n = \mathrm{Repeat}(y^*_n)$, and use it as a pseudo context video. The repeated anchor supplies the appearance condition through the trained interface and requires no further adaptation. Longer segments are decoded autoregressively in windows, using adjacent generated frames as context.

## 4 LVEdit-Bench: A Benchmark for Long-Video Editing

Existing video editing benchmarks target short, single-shot clips where temporal consistency reduces to local frame smoothness. These conditions do not reflect realistic long-video usage, in which an edit target may recur across multiple shots or disappear and re-emerge after long gaps, and an instruction may apply only to a small subset of the timeline. We introduce **LVEdit-Bench**, designed to evaluate two requirements that short-video benchmarks do not stress-test: *global consistency* across temporally distant segments and shot transitions, and *local consistency* within dense temporal windows.

**Benchmark Curation.** Source videos are sampled from open-source long-video understanding benchmarks [30, 31] and span a range of durations, scene types, and shot counts. We manually author 100 editing instructions across four categories: *addition*, *removal*, *modification*, and *global style*. For each (video, prompt) pair, temporal grounding is annotated via a two-stage pipeline: a vision-language model produces an initial temporal grounding annotation, which human annotators then review and refine. The full benchmark curation schema is detailed in Appendix B.




(a). Video Duration (b). Video Category (c). Target Interval Ratio (d). Target Segments per Video

Figure 4: **LVEdit-Bench data statistics.**

**Benchmark Statistics.** Figure 4 summarizes LVEdit-Bench. Videos have a mean duration of 111.90 s (median 120 s), well beyond the few-second range of existing editing benchmarks (a). The 100 videos span seven domains (b), covering diverse real-world content. Per-video target interval ratios (c) average 0.66 with substantial within-category variance, mixing dense and sparse editing scenarios. The number of edit segments per video (d) has a mean of 6.15 and a long tail to 34, reflecting multi-shot videos in which the same target recurs across many disjoint segments.

### 4.1 Evaluation Settings

To make head-to-head comparison with short-video baselines meaningful, we organize evaluation into three settings of increasing temporal complexity. Each setting focuses on one of three evaluation requirements: per-shot fidelity, cross-shot identity preservation, and robustness to target-absent content. All inputs are cropped to fit within every baseline's native context window, so differences reflect algorithmic capability rather than failures caused by exceeding context limits. Full-length behavior is examined separately in Section 5.3.

**Setting 1: Single-shot.** We sample 61 continuous clips under 5 seconds from LVEdit-Bench. This setting matches the existing short-video setting and measures per-clip editing quality (instruction adherence, temporal consistency, preservation, visual quality) without long-video-specific factors.

**Setting 2: Cross-shot, target present.** We use short clips depicting the same target across multiple video transitions, drawn from the same consistency group. The target is present throughout, but the clip spans shot cuts with changes in viewpoint, scale, lighting, pose, and background. Across 54 such clips, this setting isolates whether a method can align the same target across transitions and maintain a coherent edited identity, evaluated by *global consistency*, which scores cross-shot coherence of the edited target across the full clip.

**Setting 3: Target-absent intervals.** We use 65 short clips, each containing at least one target-containing shot followed by intervening target-absent content, where the target disappears across shot transitions rather than within a single shot. Unlike Setting 2, the target is absent for portions of the clip and the clip may include distractors that must not be edited. This setting evaluates *robustness to irrelevant content*, capturing whether non-target intervals are correctly preserved — the failure mode most characteristic of long videos in the wild.

Beyond the three head-to-head settings above, we also use full-length LVEdit-Bench videos (no cropping) for an *end-to-end analysis* in Section 5.3. Because these inputs exceed the baselines' native context windows and cause performance degradation primarily through context limits, this regime is reserved for analyses of our own method, length scaling, qualitative comparisons, and failure-mode characterization, rather than head-to-head numerical comparison.

**Evaluation Metrics.** We use a structured VLM judge as the primary evaluator, since standard metrics miss cross-shot drift, capture only adjacent-frame consistency, or cannot distinguish correctly preserved content from merely similar content. The judge model (GEMINI-3.1-PRO [12]) is distinct from the VLMs used inside our pipeline (planning agent and anchor-bank critic); Appendix C gives metric definitions and additional OpenVE results under its original protocol. For each dimension, the judge receives a dimension-specific evaluation pack (e.g. consistency-group pairs, target-absent segments). Three dimensions are shared across all settings: *instruction adherence* (IA), *temporal consistency* (TC), and *visual quality* (VQ). Setting 1 adds *preservation* (PRS); Setting 2 adds *cross-shot semantic consistency* (CSC) and *transition robustness* (TR); Setting 3 adds *cross-shot identity after reappearance* (CSIR) and *robustness to irrelevant content* (RIC). The *Average* column is the unweighted mean of per-dimension scores; all scores are on a 0–100 scale (higher is better). Judge prompts are in Appendix D.

## 5 Experiments

### 5.1 Implementation Details

We initialize from a pretrained instructional video editor [10] that uses QWEN2.5-VL-3B [32] as the multimodal encoder. We adapt the DiT self-attention layers with LoRA (rank $r = 128$), keeping the base DiT weights and the MLLM frozen. We sample $\sim$60k examples from the training corpus of [10], use 16 context frames, and fine-tune for 1.50k steps. The learning rate is $5\times10^{-5}$, with a global batch size of 32 and 8 gradient accumulation steps. At inference, the assigned anchor is temporally repeated to form a pseudo context video, and the MLLM's input frame is randomly sampled from this context. The image editing prior $G$ is QWEN-IMAGE-EDIT-2511 [13], referred to as Qwen-Image below. The multimodal critic $J$ and the planning module $\Phi$ are both instantiated with GEMINI-3-FLASH [12] under task-specific prompts (Appendix D).

Table 1: **Setting 1: single shot.**

| Method | IA | TC | PRS | VQ | Avg. ↑ |
|---|---|---|---|---|---|
| Ditto | 33.39 | 41.48 | 19.34 | 36.23 | 32.61 |
| UniVideo | 71.64 | 59.02 | 70.25 | 59.92 | 65.20 |
| Lucy-Edit | 44.59 | 59.51 | 66.39 | 61.48 | 57.99 |
| Kiwi-Edit | 81.26 | 62.02 | 73.11 | 66.15 | 70.64 |
| **Ours** | **92.40** | **72.68** | **88.28** | **76.39** | **82.44** |

Table 2: **Results on OpenVE [7].**

| Method | LC | LR | LA | BG | GS | Overall |
|---|---|---|---|---|---|---|
| Ditto | 40.60 | 30.60 | 28.20 | 33.60 | 80.20 | 42.00 |
| Lucy-Edit | 64.00 | 35.00 | 46.00 | 31.40 | 45.40 | 44.40 |
| UniVideo | 77.20 | 49.60 | **57.40** | 49.40 | 69.40 | 61.00 |
| Kiwi-Edit | 76.60 | 52.60 | 47.20 | 52.80 | 72.80 | 60.40 |
| **Ours** | **93.60** | **75.20** | 55.20 | **57.80** | **83.00** | **73.00** |

*OpenVE scores:* 0–5 *rescaled to* 0–100. *LC/LR/LA: local change/removal/addition; BG: background; GS: global style.*

Table 3: **Setting 2: cross-shot, target present.**

| Method | IA | CSC | TR | TC | VQ | Avg. ↑ |
|---|---|---|---|---|---|---|
| Ditto | 16.30 | 22.91 | 26.54 | 36.54 | 27.04 | 25.87 |
| UniVideo | 41.11 | 44.76 | 46.17 | 57.53 | 50.74 | 48.06 |
| Lucy-Edit | 38.52 | 39.68 | 41.36 | 61.48 | 59.63 | 48.13 |
| Kiwi-Edit | 51.02 | 53.49 | 50.99 | 62.72 | 51.11 | 53.87 |
| **Ours** | **71.85** | **65.19** | **64.44** | **71.85** | **69.51** | **68.57** |

Table 4: **Setting 3: target-absent intervals.**

| Method | IA | CSIR | RIC | TC | VQ | Avg. ↑ |
|---|---|---|---|---|---|---|
| Ditto | 14.29 | 39.39 | 3.86 | 40.24 | 29.33 | 25.42 |
| UniVideo | 45.00 | 68.56 | 34.87 | 55.76 | 49.18 | 50.68 |
| Lucy-Edit | 49.82 | 72.71 | 62.73 | 70.98 | 63.06 | 63.86 |
| Kiwi-Edit | 56.82 | 86.31 | 25.41 | 61.18 | 44.63 | 54.87 |
| **Ours** | **86.29** | **92.94** | **83.62** | **89.65** | **84.00** | **87.30** |

### 5.2 Main Results

We compare against four representative open-source short-video editors that span the current design space: **Ditto** [6], **Kiwi-Edit** [10], **Lucy-Edit** [23], and **UniVideo** [9]. The results in this subsection use the structured VLM judge described in Section 4.1 and a **0–100 scale (higher is better)**.

**Setting 1 (single-shot).** Table 1 reports per-clip editing quality on single-shot inputs. Our method outperforms all four baselines on every dimension, with margins of $+11.14$ on instruction adherence and $+15.17$ on preservation over the next-best baseline, Kiwi-Edit. To verify that these gains generalize beyond LVEdit-Bench, Table 2 reports performance on the existing short-video benchmark OpenVE [7], where our method again leads on the overall score. The largest gains are in local change and removal, while UniVideo remains strongest on local addition. These evaluations show that the long-video components retain short-video fidelity: on inputs within the baselines' intended scope, anchor conditioning produces sharper and more faithful edits than direct video-to-video baselines.

> **Takeaway 1.** The anchor-conditioned propagation model transfers the image editing prior to video, matching dedicated video editors on short clips and remaining robust in the wild.

**Setting 2 (cross-shot, target present).** Table 3 reports results on composite clips in which the target remains present across shot transitions and viewpoint changes. Our method achieves the best score on every dimension. The most diagnostic metric here is *cross-shot semantic consistency*, which directly captures cross-shot identity drift: our method exceeds the next-best baseline (Kiwi-Edit) by $11.70$ points, showing that per-shot editing quality alone does not ensure a consistent appearance across views. *Transition robustness* exhibits the same pattern: baselines edit individual shots competently but fail at shot boundaries, a failure mode the anchor bank is designed to address.

> **Takeaway 2.** Current video editors are sensitive to view changes; cross-shot consistency requires anchoring all segments to a shared edited reference.

**Setting 3 (target-absent intervals).** Table 4 reports the realistic long-video regime, where the target is absent for portions of the input and the intervening intervals must be left unchanged. The two long-video-specific dimensions reveal a sharp separation. On *cross-shot identity after reappearance*, our method reaches $92.94$, exceeding the next-best baseline (Kiwi-Edit, $86.31$) by $6.63$ points. On *robustness to irrelevant content*, our method scores $83.62$, compared with $62.73$ for the strongest baseline, Lucy-Edit; UniVideo scores $34.87$, Kiwi-Edit $25.41$, and Ditto $3.86$. Baselines with reasonable per-segment quality still edit irrelevant intervals and distractors. The planning agent sends only segments marked for editing to the anchor-conditioned editor. All other segments remain

**Takeaway 3.** Baseline errors on irrelevant content highlight the importance of *frame-level temporal grounding*. Without it, editors over-edit target-absent intervals and distractors.

Table 5: **Component-level ablations on Setting 1. (a)** Anchor-selection strategies. **(b)** MLLM input with and without a context frame. **(c)** Critic refinement from $t$=0 to $t$=3.

**(a) Anchor selection**

| Anchor | Overall ↑ |
|---|---|
| First frame | 72.55 |
| Random frame | 70.19 |
| Planner-selected | **78.15** |

**(b) Context frame**

| MLLM input | Instr. Adh. | Preserv. | Vis. Qual. |
|---|---|---|---|
| w/o context | **93.11** | 73.67 | **77.33** |
| w/ context | 91.11 | **75.39** | 76.33 |

**(c) Refinement rounds**

| Round | Instr. Adh. | Temp. Cons. | Vis. Qual. |
|---|---|---|---|
| $t$=0 | 70.51 | 71.41 | 66.55 |
| $t$=1 | 76.12 | 70.80 | 67.82 |
| $t$=2 | 72.41 | 73.79 | 71.49 |
| $t$=3 | **77.19** | **76.02** | **72.28** |

Table 6: **System-level ablations with first-frame anchors. (a)** Vary the image editor with our propagator fixed. **(b)** Fix the Qwen-Image anchor and vary the video backbones. Both controls omit planning, cross-shot sharing, and refinement. Setting 1 uses Gemini-3.1-Pro (0–100); OpenVE uses Gemini-2.5-Pro (0–5) under its original protocol.

**(a) Different image editors**

| Image editor | Setting 1 ↑ | OpenVE ↑ |
|---|---|---|
| Kiwi-Edit (single frame) | 64.29 | 3.05 |
| Qwen-Image | 72.55 | **3.64** |
| FLUX Kontext | **74.61** | 3.46 |

**(b) Different video backbones**

| Video backbone | Setting 1 ↑ | Δ vs. Kiwi-Edit |
|---|---|---|
| Kiwi-Edit | 71.29 | 0.00 |
| UniVideo | 68.53 | −2.76 |
| Ours | **72.55** | **+1.26** |

unchanged. In this setting, our method achieves its largest gain in average score over the strongest baseline, even with distractors.

### 5.3 Ablation Studies and End-to-End Analysis

We first examine three design choices within the framework: anchor selection, context input, and critic refinement. We then separate the contributions of the image editor and video propagator using a shared first-frame configuration, including an evaluation on OpenVE. Finally, we assess the complete pipeline as video duration increases.

**Component-level Ablations.** *Anchor selection.* To isolate anchor selection, we fix the image editor and our propagator and use one image-editing pass without refinement. Planner-selected anchors improve the Setting 1 Overall score by 5.60 and 7.96 points over first-frame and random-frame anchors, respectively (Table 5(a)). All scores measure complete output videos. These gains show that choosing a suitable view of the target benefits downstream editing before critic refinement.

*Context input.* We next test whether a context frame helps the MLLM preserve the source content. This comparison and the refinement-round study use the subset with complete three-round trajectories. Their absolute scores are therefore not directly comparable to the full Setting 1 results in Table 1. Adding context improves preservation by 1.72 points, at the cost of 2.00 in instruction adherence and 1.00 in visual quality (Table 5(b)). We retain the context frame to preserve unrelated content.

*Refinement rounds.* On the same subset, Table 5(c) tracks the effect of successive critic rounds. Instruction adherence, temporal consistency, and visual quality improve by 6.68, 4.61, and 5.73 points from $t$=0 to $t$=3. Visual quality improves at every round, while instruction adherence and temporal consistency fluctuate. All three metrics peak at $t$=3 on this subset, supporting $T_{\max}$=3.

**System-level Ablations.** We next vary the off-the-self image and video editing models using first-frame anchors, without planning, cross-shot sharing, and refinement modules.

*Image-editor choice.* Table 6(a) varies the image editor while retaining our video propagator. Each variant uses one image-editing pass; Kiwi-Edit supplies only the edited anchor, and our propagator

Table 7: **End-to-end evaluation on full-length LVEdit-Bench.**

| Duration | Instr. Adh. ↑ | Long-Range Cons. ↑ | Temp. Loc. & Preserv. ↑ | Temp. Cons. ↑ | Vis. Qual. ↑ | Avg. ↑ |
|---|---|---|---|---|---|---|
| 30 s | 60.10 | **58.63** | **41.40** | **47.72** | **44.23** | **50.42** |
| 60 s | 61.12 | 57.12 | 40.24 | 45.61 | 41.71 | 49.16 |
| 90 s | **61.51** | 55.66 | 37.68 | 43.66 | 39.84 | 47.67 |
| full | 56.77 | 54.99 | 33.92 | 37.67 | 35.42 | 43.75 |

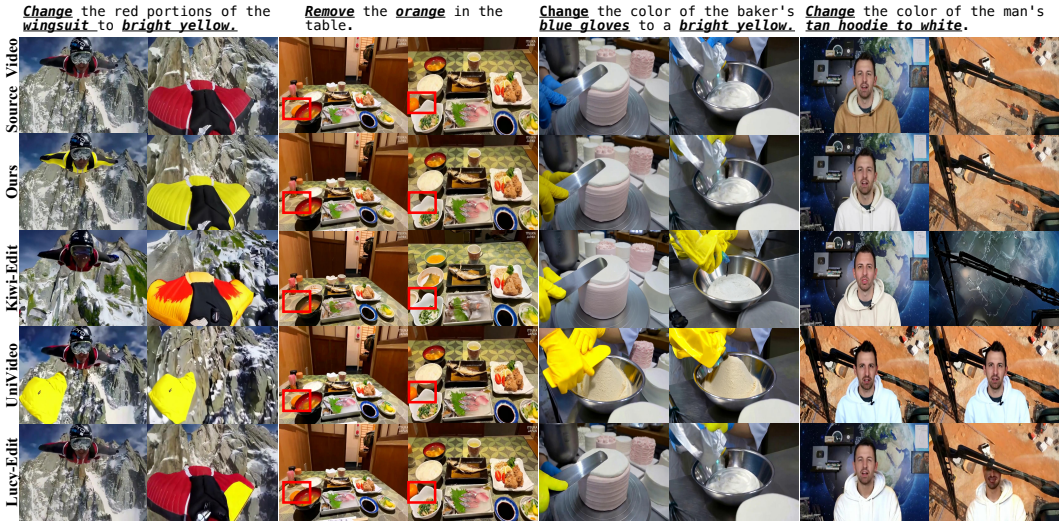



Figure 5: **Qualitative comparison.** Our method follows the instruction precisely, keeps the edited target consistent across shot cuts and viewpoint changes, and leaves non-target content unchanged.

generates the final video in every row. We evaluate complete outputs on Setting 1 and OpenVE under the standard protocols. Appendix C.2 gives the detail task breakdown on OpenVE benchmark.

Replacing Kiwi-Edit with Qwen-Image raises Overall by 8.26 points on Setting 1 and 0.59 on OpenVE. FLUX Kontext leads on Setting 1 (74.61), while Qwen-Image leads on OpenVE (3.64). Both alternatives improve video editing quality; their ranking depends on the benchmark.

*Video-backbone choice.* To separate propagation quality from anchor quality, Table 6(b) gives all three propagators the same first frame edited by Qwen-Image. The Qwen-Image + Ours configuration is shared by both panels (72.55). Our propagator exceeds Kiwi-Edit and UniVideo by 1.26 and 4.02 points, respectively. The higher full-system score of 82.44 (Table 1) also uses planner-selected, critic-refined anchors. Our propagator improves video quality with the edited anchor held fixed.

**End-to-End Analysis (Table 7).** The preceding controls use short clips. We now evaluate the complete pipeline on LVEdit-Bench at 30 s, 60 s, 90 s, and full video duration. The average score decreases progressively by 6.67 points from 30 s to full duration. Instruction adherence rises from 60.10 to 61.51 between 30 s and 90 s before dropping at full duration. Over the full duration range, temporal consistency shows the largest decrease (−10.05), followed by visual quality (−8.81) and target localization and preservation (−7.48). Long-range consistency decreases less (−3.64), consistent with the anchor bank helping maintain target appearance across distant segments. These results identify local continuity and visual quality as priorities for longer videos.

### 5.4 Qualitative Analysis

Figure 5 compares our method with the three strongest baselines on four representative cases. Baselines miss target regions (Kiwi-Edit on the wingsuit and Lucy-Edit on the hoodie) or extend edits to unrelated content (UniVideo recolors the bowl as well as the gloves). The glove and hoodie cases also show changes in color or saturation across cuts, while our method maintains a more

consistent appearance. In the removal case, baselines leave residual content when the target reappears, whereas our method removes it in both views. These observations agree with the quantitative results in Tables 1–4, particularly for cross-shot consistency and target-absent intervals.

**Open Challenges.** Two challenges in long-video editing remain, both related to modeling extended temporal context (Figure 6). *Reasoning over long context.* A long video may contain the target in only a few segments, while other segments contain visually similar content, such as other people in the same scene or objects viewed from different angles. The model must examine the full timeline and identify the segments to which the instruction applies. Our planner processes one shot at a time. Distinguishing the target from visually similar content requires joint reasoning across shots. Current VLMs struggle with this when irrelevant content dominates. Grounding errors can cause missed edits or changes to distractors before propagation begins. An edited anchor cannot recover a target that the planner has excluded. *Staying consistent across views.* The same target can appear at different angles, scales, and poses across shots (Figure 6b). One anchor frame specifies the target appearance but does not describe how it should change with the viewpoint. This problem is less pronounced in short clips with limited camera motion and becomes more frequent in long videos. Anchors with geometric or multiview information may help.




Figure 6: **Emergent challenges in long videos.**

## 6 Conclusion

We studied long-video instructional editing in the wild by decomposing it into long-range semantic recurrence and short-range temporal continuity. Our framework combines a planning agent for instruction grounding and shot-aligned segmentation, a critic-driven anchor bank for cross-shot consistency, and an anchor-conditioned editor. The editor uses a shared training-inference interface to extend a pretrained short-video model to long videos without long-video supervision. On LVEdit-Bench, the first benchmark for this regime, our method matches strong short-video editors on per-segment quality and substantially improves cross-shot consistency and grounding, with the margin widening as temporal complexity grows. The two-axis formulation, adaptation strategy, and LVEdit-Bench provide a basis for evaluating long-video instructional editing.

## Acknowledgements

This research is supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG3-RP-2022-030). Yiqi Lin and Guoqiang Liang were also supported by the NUS Guangzhou Research Translation and Innovation Institute (GRTII)-Affiliated PhD Scholarship Program.

## References


[1] 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*, pages 16692–16701, 2025.

[2] Bojia Zi, Penghui Ruan, Marco Chen, Xianbiao Qi, Shaozhe Hao, Shihao Zhao, Youze Huang, Bin Liang, Rong Xiao, and Kam-Fai Wong. Se\~ norita-2m: A high-quality instruction-based dataset for general video editing by video specialists. *arXiv preprint arXiv:2502.06734*, 2025.

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

[4] Shoubin Yu, Difan Liu, Ziqiao Ma, Yicong Hong, Yang Zhou, Hao Tan, Joyce Chai, and Mohit Bansal. Veggie: Instructional editing and reasoning video concepts with grounded generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15147–15158, 2025.

[5] Zeyinzi Jiang, Zhen Han, Chaojie Mao, Jingfeng Zhang, Yulin Pan, and Yu Liu. Vace: All-in-one video creation and editing. *arXiv preprint arXiv:2503.07598*, 2025.

[6] 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.

[7] 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.

[8] Chong Mou, Qichao Sun, Yanze Wu, Pengze Zhang, Xinghui Li, Fulong Ye, Songtao Zhao, and Qian He. Instructx: Towards unified visual editing with mllm guidance. *arXiv preprint arXiv:2510.08485*, 2025.

[9] 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.

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

[11] 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.

[12] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

[13] Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng ming Yin, Shuai Bai, Xiao Xu, et al. Qwen-image technical report, 2025.

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

[15] 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*, pages 16672–16681, 2025.

[16] Jay Zhangjie Wu, Yixiao Ge, Xintao Wang, Stan Weixian Lei, Yuchao Gu, Yufei Shi, Wynne Hsu, Ying Shan, Xiaohu Qie, and Mike Zheng Shou. Tune-a-video: One-shot tuning of image diffusion models for text-to-video generation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7623–7633, 2023.

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

[18] Duygu Ceylan, Chun-Hao P Huang, and Niloy J Mitra. Pix2video: Video editing using image diffusion. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 23206–23217, 2023.

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

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

[21] Ozgur Kara, Bariscan Kurtkaya, Hidir Yesiltepe, James M Rehg, and Pinar Yanardag. Rave: Randomized noise shuffling for fast and consistent video editing with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6507–6516, 2024.

[22] Xichen Pan, Satya Narayan Shukla, Aashu Singh, Zhuokai Zhao, Shlok Kumar Mishra, Jialiang Wang, Zhiyang Xu, Jiuhai Chen, Kunpeng Li, Felix Juefei-Xu, et al. Transfer between modalities with metaqueries. *arXiv preprint arXiv:2504.06256*, 2025.

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

[24] Enxin Song, Wenhao Chai, Guanhong Wang, Yucheng Zhang, Haoyang Zhou, Feiyang Wu, Haozhe Chi, Xun Guo, Tian Ye, Yanting Zhang, et al. Moviechat: From dense token to sparse memory for long video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18221–18232, 2024.

[25] Ziyang Wang, Shoubin Yu, Elias Stengel-Eskin, Jaehong Yoon, Feng Cheng, Gedas Bertasius, and Mohit Bansal. Videotree: Adaptive tree-based video representation for llm reasoning on long videos. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 3272–3283, 2025.

[26] Xiaoqian Shen, Yunyang Xiong, Changsheng Zhao, Lemeng Wu, Jun Chen, Chenchen Zhu, Zechun Liu, Fanyi Xiao, Balakrishnan Varadarajan, Florian Bordes, et al. Longvu: Spatiotemporal adaptive compression for long video-language understanding. *arXiv preprint arXiv:2410.17434*, 2024.

[27] Yongdong Luo, Xiawu Zheng, Guilin Li, Shukang Yin, Haojia Lin, Chaoyou Fu, Jinfa Huang, Jiayi Ji, Fei Chao, Jiebo Luo, et al. Video-rag: Visually-aligned retrieval-augmented long video comprehension. *arXiv preprint arXiv:2411.13093*, 2024.

[28] Shuheng Zhang, Yuqi Liu, Hongbo Zhou, Xiao Chen, Jun Peng, Xiaoshuai Sun, Yiyi Zhou, and Rongrong Ji. Adaflow: Efficient long video editing via adaptive attention slimming and keyframe selection. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 10187–10191. IEEE, 2026.

[29] Yangyi Cao, Yuanhang Li, Lan Chen, and Qi Mao. Mlv-edit: Towards consistent and highly efficient editing for minute-level videos. *arXiv preprint arXiv:2602.02123*, 2026.

[30] Haoning Wu, Dongxu Li, Bei Chen, and Junnan Li. Longvideobench: A benchmark for long-context interleaved video-language understanding. *Advances in Neural Information Processing Systems*, 37:28828–28857, 2024.

[31] Guo Chen, Yicheng Liu, Yifei Huang, Yuping He, Baoqi Pei, Jilan Xu, Yali Wang, Tong Lu, and Limin Wang. Cg-bench: Clue-grounded question answering benchmark for long video understanding. *arXiv preprint arXiv:2412.12075*, 2024.

[32] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

## Supplementary Material

This appendix provides additional details that complement the main paper. We organize it into five parts: (**A**) implementation details and editor training; (**B**) LVEdit-Bench curation, annotation procedures, and dataset statistics, expanding Section 4; (**C**) the structured VLM-judge evaluation protocol, including metric definitions; (**D**) full prompt listings for the planning agent, the anchor-bank critic, the image-editing prior, and the evaluator; (**E**) discussion of limitations, broader impact, and ethical considerations.

# A Implementation Details

### A.1 Planning Agent

Each input is first decoded into RGB frames at its native frame rate, with a maximum spatial budget of $921{,}600$ pixels per frame. We use PySceneDetect to obtain fixed scene/shot boundaries. If no boundary is detected, the entire requested frame range is treated as one shot. For each detected shot, we sample up to $4$ uniformly spaced frames, with a global cap of $96$ sampled frames per video. Each sampled frame is passed to the planner VLM together with its frame index, timestamp, shot ID, and local temporal window. The VLM makes the primary decision at the frame level and then aggregates frames into segment-level decisions. The returned plan contains a video summary, target and edit descriptions, localized prompts, frame-level visibility, reference-frame candidates, and a best global anchor view. Editable planner segments are converted from seconds to frame indices and clipped to the requested evaluation horizon. The video propagation chunk size is $81$ frames by default and is rounded down to the nearest $4n{+}1$ value. Requests shorter than $65$ frames are padded by repeating the last source frame, and the generated tail is trimmed after inference. Each editable execution segment receives exactly one anchor. Candidate anchors are selected from grounded planner frames inside the segment; if no grounded frame is available, uniformly spaced frames from the segment are used. The keyframe-selection VLM favors frames that show the complete target clearly, in sharp focus, and with minimal occlusion. It also considers whether each frame represents the shot well and is suitable for propagation.

### A.2 Anchor-Conditioned Editor

The propagation model starts from the Kiwi-Edit [10] video editor, using a Wan2.2-TI2V-5B [11] DiT backbone and Qwen2.5-VL-3B-Instruct [32] as the multimodal encoder. We inject LoRA adapters with rank $r{=}128$ and $\alpha{=}128$ into the DiT self-attention `q`, `k`, and `v` projections in all $30$ DiT blocks. The output projection, MLP layers, MLLM, query vectors, VAE, and all base DiT weights are frozen. During training, each paired clip is split into $P{=}16$ previous target frames and a current target clip. The previous target frames are encoded by the same VAE, passed through a lightweight conditional embedder, patchified, and prepended as prefix tokens before the current noisy target tokens. During training, we draw a random total length between $49$ and $81$ frames. After removing the $16$-frame prefix, the target-frame count must satisfy the $4n{+}1$ constraint. Frames are resized/cropped by the dataset operator to respect the $921{,}600$-pixel budget and dimensions divisible by $32$. We train for $1.50$k steps with mixed-precision `bf16`. The main script uses $8$ processes, gradient accumulation $4$, AdamW with learning rate $5 \times 10^{-5}$ and weight decay $0.01$, and a constant learning-rate scheduler. The default inference configuration uses $50$ sampling steps with `sigma_shift`$= 5.00$.

### A.3 Anchor Bank

All anchor-frame image edits are completed before the first video-propagation request. The VLM first selects the best visible overall seed anchor. This seed is edited without cross-shot references (or with a clean non-edit reference for removal tasks). Every non-seed anchor is then edited from its original source keyframe while conditioning on the edited seed. This produces an initial globally consistent anchor memory. We use Qwen-Image as specified in Section 5.1. Before each image-edit call, the planner VLM rewrites the user instruction into a short model-facing prompt. If references are present, the prompt must identify "Image 1" as the source and "Image 2" as the reference. It must also forbid copying reference background, camera, pose, layout, or unrelated objects into the source frame. The deterministic fallback prompt is used whenever rewriting fails.

After each keyframe edit, a VLM critic scores the candidate on target presence, reference fidelity, background preservation, artifact severity, and identity consistency. A candidate is accepted only if the critic marks it accepted, the aggregate score is at least $0.70$, and no axis is below $0.30$. Rejected edits are retried with a different seed and, when available, the critic's improved prompt. For refinement rounds, each anchor is re-edited from the original source keyframe. Candidate references are other accepted edited anchors and, for removal, the selected clean non-edit frame. The VLM assigns a Reference Affinity Score based on lighting, viewpoint, scene/background compatibility, pose/layout, and scale. The highest-scoring compatible references are passed to the image editor. In each refinement round, anchors judged as good are carried forward without another image-edit request. Anchors with strong failure or slight error are re-edited using affinity-ranked references from the previous memory snapshot.

# B LVEdit-Bench: Curation and Statistics

Source videos are sampled from LONGVIDEOBENCH [30] and CG-BENCH [31], retaining videos that satisfy the following criteria: (i) duration between $30$ s and $300$ s; (ii) at least three distinct shots; (iii) presence of one or more recurring objects, persons, or settings; (iv) sufficient visual quality (no severe compression artifacts or extreme low light). After filtering, we sample to balance domain coverage across seven categories: *daily life, sports, travel, animation, documentary, vlog, and tutorial* (Figure 4b). The 100 instructions are manually authored by three annotators familiar with the video-editing literature. The annotators are instructed to write prompts that span the four edit categories (addition, removal, modification, global style). We pair each instruction with a suitable source video and exclude instructions trivially satisfied by a single-shot edit.

**Duration.** Mean $111.90$ s, median $120$ s, range $[60.00, 126.90]$ s for the released 100-video split. **Shot count.** Mean $9.40$, median $7$, max $34$. **Edit-segment count.** Mean $6.15$, median $4$, max $34$. **Positive-interval ratio.** Mean $0.66$, std $0.21$. The ratio is bimodally distributed: dense edits ($> 0.80$) account for $\sim 45\%$ of examples, sparse edits ($< 0.40$) for $\sim 22\%$. **Temporal-regime distribution.** Each example's temporal regime is stored as `temporal regime`; the released statistics script reports the exact counts for continuous, sparse, disjoint, and multi-shot recurring cases from the final annotations.

For each example, we construct head-to-head clips for Settings 1–3 by cropping the source according to the annotation: **Setting 1** clips are continuous single shots of $\leq 5$ s containing the target. **Setting 2** clips concatenate 2–4 shots from the same consistency group, all containing the target, with shot transitions preserved. **Setting 3** clips concatenate target-containing and target-absent shots with at least one transition between them. All clips are capped at the shortest baseline's native context window so that no method is penalized by truncation.

# C Evaluation Protocol

The judge is implemented using GEMINI-3.1-PRO [12]. For each evaluation dimension, we construct a dimension-specific *evaluation pack*: a structured bundle containing the source and edited videos, the instruction, and dimension-specific auxiliary information (e.g., consistency-group pairs, target-absent intervals, preservation regions). The judge returns a score in $[0, 100]$ and a brief justification.

## C.1 Per-Dimension Definitions

**Instruction Adherence (IA).** Whether the edit faithfully realizes the instruction in semantic content, attributes, and scope. The judge is shown the instruction, the source, and the edit, and asked to rate the edit's match to the requested change. **Temporal Consistency (TC).** Whether the edited video is locally smooth and free of flicker, jitter, and within-shot identity drift, evaluated over dense temporal windows. **Visual Quality (VQ).** Sharpness, absence of artifacts, and plausibility of textures, lighting, and composition. **Preservation (PRS, Setting 1).** Whether non-target regions, background, and unrelated content are unchanged. **Cross-Shot Semantic Consistency (CSC, Setting 2).** For each consistency group, the judge is shown the edited renderings of all segments in the group and rates whether the same target is depicted with consistent identity, attributes, and style. Pairwise scores are averaged within each group, then across groups. **Transition Robustness (TR, Setting 2).** Whether the edit remains coherent across shot cuts and viewpoint changes — diagnostic of the failure mode

shared by all per-segment baselines. **Cross-Shot Identity After Reappearance (CSIR, Setting 3).** Whether the target retains the edited identity after disappearing and reappearing across one or more target-absent shots. **Robustness to Irrelevant Content (RIC, Setting 3).** Whether non-target shots are left visually unchanged. We compare the source and edited content in segments annotated as target-absent. Distractors are explicitly probed: the judge is asked whether visually similar non-target objects are spuriously edited.

### C.2 OpenVE under Its Original Evaluation Protocol

Table 8 reports per-task scores for the image-editor variants in Table 6(a) and native Kiwi-Edit (Overall 3.13). Our variants construct first-frame anchors as described in Section 5.3. Gemini-2.5-Pro scores complete output videos on OpenVE's original 0–5 scale. These scores cannot isolate the effect of judge choice because the system configuration also differs from Table 2.

Qwen-Image leads on global style and local removal, while FLUX Kontext leads on background change and local change. Native Kiwi-Edit achieves the highest score on local addition. The best configuration therefore depends on the edit type.

Table 8: **OpenVE task-level breakdown with Gemini-2.5-Pro.** Scores use the original 0–5 scale (↑). Overall scores for our three image-editor variants appear in Table 6(a).

| Method / first-frame prior | Bg. | Style | Add | Change | Remove |
|---|---|---|---|---|---|
| Kiwi-Edit baseline | 2.68 | 3.50 | 2.83 | 3.85 | 2.78 |
| Ours + Kiwi | 2.67 | 3.45 | 2.41 | 3.82 | 2.92 |
| Ours + FLUX Kontext | **2.91** | 3.63 | 2.61 | **4.71** | 3.47 |
| Ours + Qwen-Image | 2.89 | **4.15** | 2.76 | 4.68 | **3.76** |

## D Prompts

We list the prompt templates used by the planner, global-consistency module, image-editing prior, and evaluator. Runtime variables such as the instruction, frame manifest, candidate list, and metadata are inserted into the bracketed slots before the API request. All VLM calls request JSON-only outputs, and the implementation validates and normalizes the returned keys before use.

### D.1 Planning Agent

Prompt. 1: Segment classification and temporal grounding

**Role.** You are a planner for long-video instructional editing.
**Task.** The provided temporal segments are fixed shot/scene boundaries. Make the primary decision at the frame level: for every provided frame sample, predict whether the edit should be applied on that frame. Then classify each temporal segment as editable only if at least one grounded frame in that segment should be edited. Use `frame_index` as the canonical reference key. Do not invent new temporal boundaries outside the provided segments. Prefer best visible reference frames for modified-content consistency.
**Inputs.** Editing instruction; fixed temporal segments with `segment_id`, `scene_id`, `shot_id`, frame range, and time range; sampled-frame manifest with `sample_index`, `frame_index`, timestamp, and local window.
**Return JSON only** with fields `video_summary`, `target_description`, `edit_description`, `global_prompt`, `consistency_notes`, `memory_strategy`, `segment_classification`, `sampled_frame_grounding`, `best_view_frames`, and `best_anchor_view`. Each `segment_classification` item must contain `should_edit`, localized `prompt`, `target_visible`, `reference_sample_indices`, and `reference_frame_indices`. Each frame-grounding item must contain `edit_this_frame`, `best_view_score`, `should_use_as_reference`, `local_prompt`, and `reason`.

**Calibration example.** If the instruction is "change the baker's blue gloves to yellow," a frame where the baker's hands are visible should set `edit_this_frame=true`; an audience-only cutaway should set `edit_this_frame=false`; a sharp, frontal hand view should receive a high `best_view_score` and can be listed as a reference frame.

Prompt. 2: Task classification for execution policy

Classify the long-video edit into exactly one of `local_change`, `remove`, `style_transfer`, or `background_replacement`. Only `style_transfer` and `background_replacement` may set `whole_video_propagation=true`. Do not classify editing a small object in the background as background replacement; for example, "change the blue refrigerator in the background to pink" is `local_change`. If the instruction removes or replaces the whole background, use `background_replacement`. The prompt includes the user instruction, planner instruction, target description, edit description, global prompt, and up to twelve planner segments with prompt/reason previews. Return JSON only with `category`, `confidence`, `whole_video_propagation`, and `reason`.

Prompt. 3: Editable-segment keyframe anchor selection

Select the best keyframe anchor for one editable video segment. Do not blindly choose the first frame. Prefer a frame where the target is most visible and complete, sharp, minimally occluded, representative of the shot, stable in pose/layout, and suitable as the anchor for consistent editing across the whole segment. The prompt includes the instruction, target description, edit description, segment execution index, plan segment id, frame range, and the candidate frame list with timestamps. Return JSON only with `candidate_index`, `frame_index`, `visibility_score`, and `reason`.

Prompt. 4: Global seed-anchor selection

Select the best visible overall seed anchor for the global consistency module. The seed will be edited first and then used to initialize the edited reference memory for all other anchors. Prefer anchors where the target is complete, clear, minimally occluded, representative of the video, visually central, and useful for other views. The prompt includes the instruction, target description, edit description, and all candidate anchors with frame indices and segment ids. Return JSON only with `anchor_index` and `reason`.

Prompt. 5: Removal clean-reference selection

The user requested a removal/erasure edit. Choose the best non-edit reference frame to guide clean background or object removal. It should show useful background/scene context and ideally should not contain the target to remove. The prompt includes the instruction, target description, edit description, and a candidate list of target-absent or non-edit frames. Return JSON only with `candidate_index`, `frame_index`, and `reason`.

### D.2 Anchor-Bank Critic

Prompt. 6: Keyframe image-edit critic

You are judging a single edited keyframe before it is used to propagate a long-video edit. You receive two images: (1) *source frame*, the original unedited keyframe; and (2) *edited frame*, the candidate keyframe. Decide whether the edited frame follows the instruction while preserving unrelated content, identity, camera view, lighting, and geometry. Penalize missing edits, over-edits, severe artifacts, text corruption, identity drift, background copy, object copy, reference copy, and changes outside the requested target.

Return JSON only with `accepted`, aggregate `score` in $[0, 1]$, `axis_scores` for `target_presence`, `reference_fidelity`, `background_preservation`, `artifact_severity`, and `identity_consistency`, plus `error_severity`, `strong_failure`, `slight_error`, `failure_type`, `reason`, and `improved_prompt`. Higher is always better; for `artifact_severity`, 1.0 means no visible artifact.
**Axis calibration.** Missing target or missing requested attribute lowers `target_presence`; wrong transferred color/shape/style lowers `reference_fidelity`; copied reference background or damage to unrelated regions lowers `background_preservation`; warped hands, broken edges, or text corruption lowers `artifact_severity`; changed face, hairstyle, pose, or object identity lowers `identity_consistency`.

Prompt. 7: Reference-affinity scoring

Compute Reference Affinity Score for each reference candidate relative to the current anchor. This is not judging whether the edit is correct; it measures whether the reference is visually compatible with the current view. Consider lighting, viewpoint, scene/background, pose/layout, and scale/shot size. The prompt contains the current source anchor and each candidate edited reference image. Return JSON only with a `scores` list; each item contains `candidate_index`, aggregate `affinity`, component scores `light`, `view`, `scene`, `pose`, and `scale`, plus `reason`.

### D.3 Image Editing Prior

Prompt. 8: Deterministic image-edit prompt template

If no reference image is supplied: "Edit this single video frame according to the instruction with exact, minimal changes. Preserve facial features, hairstyle, expression, identity, pose, camera view, composition, lighting, background, and all unrelated content unless explicitly requested. Create a clean keyframe that can be used as temporal context for video propagation. Instruction: [instruction]. Segment detail: [segment detail]. Return only the edited image."
If reference images are supplied, prepend: "Image 1 is the source. Image 2 is the reference." For every reference, state that it should be used only for the requested subject, appearance, style, or clean-background role. Add: "Edit only Image 1; do not copy any reference background, camera view, pose, layout, or unrelated object into Image 1." For more than one reference, additionally state that Images 3 and above are additional references. Replacement/swap edits are limited to one reference image.

Prompt. 9: Qwen-Image prompt rewrite

Generate a minimal updated prompt for the Qwen image-edit model. If references exist, the prompt must include exactly: "Image 1 is the source. Image 2 is the reference." For more than one reference, still include that exact sentence and then mention that Images 3 and above are additional references. If there are no references, do not mention Image 2. Say exactly what to change; preserve character consistency, facial features, hairstyle, expression, identity, pose, source background, camera, lighting, layout, and unrelated content. If a reference is provided, use it only for the specified role and do not copy unwanted reference background/camera/pose/layout. Keep the prompt short and directly usable by the image model. Return JSON only with `prompt`.

### D.4 Evaluation Judge

Prompt. 10: Setting 1: single-shot judge

You are an expert video quality assessor and professional video editor. Evaluate an AI-edited single continuous shot against the original clip and the editing instruction. The uploaded videos are ordered as original then edited. Use the supplied metadata, target description, and dense validation summary. Score four dimensions without a total score: Instruction Adherence (0–30), Temporal Consistency and Local Stability (0–30), Preservation (0–20), and Visual Quality and Artifact Absence (0–20). Return JSON only with `Reasoning` and `Scores`. Calibration: a precise edit with stable boundaries, unchanged background, and no artifacts should score near the maximum; wrong-target edits, flicker, mask leakage, or warped anatomy should reduce the corresponding dimensions.

Prompt. 11: Setting 2: cross-shot judge

You are an expert video quality assessor, professional editor, and cross-shot consistency evaluator. The target is present throughout the evaluated clip, but the clip contains shot transitions and camera-view changes. Judge whether the same target remains semantically aligned and visually coherent across all views. Score Instruction Adherence (0–20), Cross-Shot Semantic Consistency (0–35), Transition Robustness (0–15), Temporal Consistency (0–15), and Visual Quality (0–15). Return JSON only with `Reasoning` and `Scores`. Penalize identity resets, abrupt attribute changes at cuts, and view-dependent drift even if individual frames look plausible.

Prompt. 12: Setting 3: target-absent and reappearance judge

You are an expert video quality assessor and long-video alignment robustness evaluator. The clip contains target-containing segments and irrelevant or target-absent segments. Evaluate whether the model maintains the correct edited identity after temporal gaps while avoiding edits to irrelevant content and distractors. Score Instruction Adherence (0–20), Cross-Shot Identity After Reappearance (0–25), Robustness to Irrelevant Content (0–25), Temporal Consistency (0–15), and Visual Quality (0–15). Return JSON only with `Reasoning` and `Scores`. Penalize leakage into negative segments, editing of distractors, missed target reappearances, and inconsistent attributes after a gap.

Prompt. 13: Full-length long-video judge

You are an expert long-video editing benchmark judge. Evaluate the edited long-video excerpt or full video against the original. The uploaded videos may be uniformly sampled to fit the judge API; do not penalize low playback frame rate or absent audio. Score Instruction Adherence (0–25), Long-Range Edit Consistency (0–25), Target Localization and Preservation (0–20), Temporal Consistency (0–15), and Visual Quality (0–15). Return JSON only with `Reasoning` and `Scores`. For target-conditional instructions, verify both positive target appearances and negative/irrelevant intervals; for global style edits, verify that the style is coherent across the whole horizon.

## E Limitations, Broader Impact, and Ethics

### E.1 Limitations

The main text discusses irrelevant-context confusion and view-induced drift (Section 5.4, Figure 6). We detail the remaining failure modes and dependencies below. The anchor bank's critic-driven refinement adds wall-clock cost beyond a single editor pass. Refinement within a round is parallelizable across anchors. For interactive applications, the bank can be warm-started across edit requests on the same video to amortize this overhead.

**Failure modes by pipeline stage.** Grounding errors, imprecise temporal boundaries, or incorrect target-absence decisions can route an edit to the wrong interval. Anchor generation and critique may miss fine-scale physical implausibilities, such as an inconsistent reflection in sunglasses. Large viewpoint and illumination changes can leave residual cross-shot appearance drift, as in the varying shade of an edited green hat. A still anchor constrains appearance more directly than newly requested motion, so a request for a dynamic lounge background can yield a static scene. Propagation can introduce local deformation. On continuous footage without clear cuts, boundaries are determined by the model's temporal range. Adjacent generated frames provide context, but the non-overlapping output chunks are concatenated directly, so visible seams can remain.

**External models and duration.** Planning, anchor construction, and evaluation inherit the limitations of their underlying foundation models. The planner and critic share GEMINI-3-FLASH, so their errors may be correlated. The larger GEMINI-3.1-PRO evaluator does not guarantee independent error detection. Our empirical claims remain within the benchmark's validated duration range (Appendix B), and do not establish performance on substantially longer videos.

### E.2 Broader Impact

This work advances instruction-driven video editing, with the potential to support creative content production and reduce post-processing effort. As with any generative media research, we acknowledge the risk of misuse for misleading or non-consensual content, and the long-video setting in particular makes manipulated content harder to detect with frame-level forensic tools.

Our experiments use publicly available datasets; the annotation safeguards are described below. The model is intended solely for responsible research and creative applications. We support transparent labeling of AI-generated content, together with forensic detection, watermarking, and cryptographic provenance. These safeguards are particularly relevant for long-form content, where temporal-consistency cues used to detect manipulation in short clips may be less reliable.

### E.3 Ethical Considerations in Benchmark Construction

LVEdit-Bench is built on top of publicly licensed long-video datasets [30, 31]; we redistribute only annotations and prompts, not video content. Instructions were authored to avoid sexualization, violence, harm-facilitation, and edits targeting identifiable real persons in non-public-figure contexts. Annotators were compensated according to the institution-approved hourly rate for research annotation and given the option to skip examples they found uncomfortable.

## NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The claims are reflected in the contributions stated in Section 1.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss technical limitations, failure modes, and the validated duration range in Appendix E.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [N/A]

Justification: The paper does not include theoretical results.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: Details needed for reproduction are included in the experiments section and the appendix.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: We will release the data and model in the https://github.com/showlab/LVEdit.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: The details are reported in the experiments section.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [No]

   Justification: We did not include error bars as the experiments are computationally expensive, and error bars are not conventionally reported in the existing literature.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [No]

   Justification: Training settings are described in Section 5.1 and Appendix A.2, but a complete hardware, memory, and runtime breakdown is not provided.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We have reviewed and followed the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We discuss the broader impact in the appendix.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [Yes]

    Justification: We expect our model to have risks similar to those of Kiwi-Edit [10] or other models trained on the same dataset. The appendix discusses mitigation strategies.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We properly cited all code, data, and models used in this paper.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: We provide samples of our benchmark in the supplementary material.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subject.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subject.

    Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [N/A]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

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