SLIDEREDIT++: IMPROVED SLIDER TRAJECTORIES IN INSTRUCTION-BASED IMAGE EDITING

Anonymous authors
Paper under double-blind review

ABSTRACT

Slider-based control aims to give users continuous, fine-grained control over the intensity of individual edits in instruction-based image editing. However, existing approaches fail to deliver reliable continuous controllability: traversing a slider can produce abrupt visual jumps and non-monotonic changes, while jointly controlled sliders can interfere with one another. To systematically study these limitations, we introduce *SlideBench*, a comprehensive benchmark for continuous image editing spanning diverse editing categories and equipped with a unified suite of metrics quantifying monotonic progression, perceptual continuity, and inter-slider disentanglement. Building on these observations, we propose SliderEdit++, which explicitly regularizes the intermediate editing trajectory rather than supervising only its endpoints. Our method combines velocity- and semantic-space trajectory regularization with spectral control to encourage smooth, predictable slider traversal, together with a disentanglement objective that reduces interference between independently controlled edits. On SlideBench, SliderEdit++ achieves state-of-the-art performance, improving the overall score by 18.5% relative to the strongest baseline. At the same time, using a few-step editing backbone makes our method approximately $3\times$ to $20\times$ faster than existing slider-based approaches, enabling substantially more responsive interactive editing.

1 INTRODUCTION

Instruction-based image editing has advanced rapidly, with recent models such as FLUX-Kontext (Labs et al., 2025), Qwen-Image-Edit (Wu et al., 2025), FLUX.2 (Labs, 2025), Nano Banana, and GPT-Image enable users to modify images through flexible natural-language instructions. While these models can strongly execute complex language instructions, they inherently produce a fixed, one-shot edit that often fails to precisely capture the user's nuanced visual intent. Recent works therefore introduce *continuous slider control* into instruction-based editing, allowing users to interactively vary the strength of an entire edit instruction or independently control individual components of a compositional instruction (Zarei et al., 2026; Parihar et al., 2026; Li et al., 2026; Ekin & Gandelsman, 2026). Instead of committing to a fixed one-shot edit, users can traverse a continuum of possible outcomes and select the result that best matches their intent. Such continuous control is particularly appealing for interactive editing, where the desired appearance is often discovered through gradual adjustment rather than specified exactly beforehand.

Despite the growing interest in slider-based image editing, their evaluation remains fragmented and inconsistent. Existing works typically rely on method-specific datasets, isolated subsets, and ad-hoc evaluation protocols, limiting meaningful comparison across approaches. Moreover, commonly used metrics often fail to capture the fine-grained perceptual behavior that defines effective continuous control. To address these limitations, we introduce *SlideBench*, a comprehensive benchmark for continuous image editing spanning a diverse range of editing categories (Fig. 2) and fine-grained control axes that users may wish to adjust continuously in practice. Alongside the benchmark, we develop a unified and robust evaluation framework specifically designed to characterize slider behavior, with principled metrics for *monotonic progression*, *perceptual continuity*, and *disentanglement*.

Evaluating existing methods on SlideBench reveals a consistent limitation: while performance varies across editing categories, slider trajectories are often far from truly continuous. Equal changes in slider value can induce highly uneven perceptual changes, including abrupt visual jumps (Fig. 3),

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

while in the multi-slider setting, adjusting one control can unintentionally affect others. Motivated by these findings, we introduce SliderEdit++, a training framework that explicitly regularizes slider trajectories while leveraging step-distilled editing backbones for efficient inference. Building on SliderEdit (Zarei et al., 2026), we introduce complementary trajectory regularizers that encourage smooth, uniform semantic progression across intermediate slider values, together with a disentanglement objective that encourages different sliders to act along orthogonal edit directions, preserving independence between controls. The resulting sliders are smooth, responsive, and independently controllable. On SlideBench, SliderEdit++ achieves state-of-the-art performance, improving the overall score by 18.5% over the strongest baseline while being approximately $3\times$ to $20\times$ faster.

In summary, our main contributions are: **(i)** we introduce *SlideBench*, a comprehensive benchmark for continuous instruction-based image editing spanning diverse editing categories, together with a unified evaluation suite for monotonic progression, perceptual continuity, and multi-slider disentanglement; **(ii)** we propose *SliderEdit++*, which explicitly regularizes slider trajectories during training to encourage smooth and uniformly progressing edits, while improving independence between multiple controls; and **(iii)** SliderEdit++ achieves *SOTA performance on SlideBench*, improving the overall score by 18.5% over the strongest baseline while operating approximately 3–$20\times$ faster.

2 Related Works

Continuous control over semantic attributes has a long history in generative modeling. Early work on VAEs and GANs studied disentangled representations and semantically meaningful latent directions that enable continuous manipulation of visual attributes (Higgins et al., 2017; Kim & Mnih, 2018; Shen et al., 2020; Härkönen et al., 2020; Shen & Zhou, 2021; Abdal et al., 2021). With text-to-image diffusion and rectified-flow models, this idea has evolved into explicit continuous controls over semantic attributes. Concept Sliders (Gandikota et al., 2024) learn attribute-specific LoRA directions, while Prompt Sliders (Sridhar & Vasconcelos, 2024) and (Baumann et al., 2025) manipulate directions in the text-embedding space. FluxSpace (Dalva et al., 2024) instead discovers semantically interpretable representations within rectified-flow transformers for fine-grained and disentangled semantic editing, while more recent works further automate the discovery of controllable directions or realize them through lightweight adapters (Gandikota et al., 2025; Chiu et al., 2026).

More recently, continuous control has been extended to instruction-based image editing, where the goal is to modulate the strength of an arbitrary natural-language edit. SliderEdit (Zarei et al., 2026) learns globally shared LoRA adapters by training them to suppress individual edit instructions, enabling both global and per-instruction sliders. Kontinuous Kontext (Parihar et al., 2026) instead explicitly conditions the editor on a scalar edit strength through a learned projector trained on synthesized strength-labeled data. Training-free approaches such as VeloEdit (Li et al., 2026) and FlowSlider (Endo et al., 2026) obtain continuous control by intervening directly in the model's velocity field, while (Ekin & Gandelsman, 2026) achieve steering through interpolation in the text-embedding space. More generic inference-time interventions such as SDEdit (Meng et al., 2021) can also trade off source preservation and edit strength without dedicated slider training. Building on these advances, our method targets the limitations of existing sliders, improving their smoothness, consistency, and independence across controls.

3 SlideBench Evaluation Benchmark

In this section, we introduce *SlideBench*, a comprehensive evaluation benchmark for continuous control in image editing. SlideBench is designed to assess a broad range of properties that characterize effective continuous editing, together with a robust suite of evaluation metrics that capture the fine-grained nuances of this setting.

3.1 Dataset Description

To systematically evaluate continuous image editing, we construct a benchmark spanning a broad range of attributes that users may wish to adjust continuously in practice. The dataset is organized into seven high-level categories—*Lighting*, *Composition*, *Camera*, *Environment*, *Character*, *Style*, and *Object*—covering 57 fine-grained subcategories, with five samples per subcategory. Each sample pairs a source image with an instruction that defines a continuous transformation between two semantic endpoints, with the same editing axis instantiated across diverse visual contexts to test generalization. The benchmark covers naturally gradual attributes, such as lighting and camera changes;

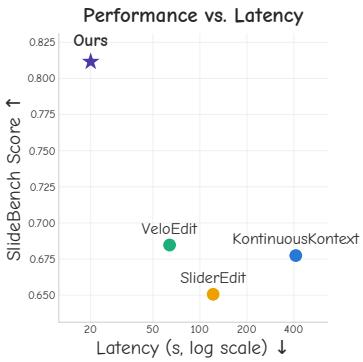

Figure 1: Our method achieves SOTA on SlideBench while being significantly faster.

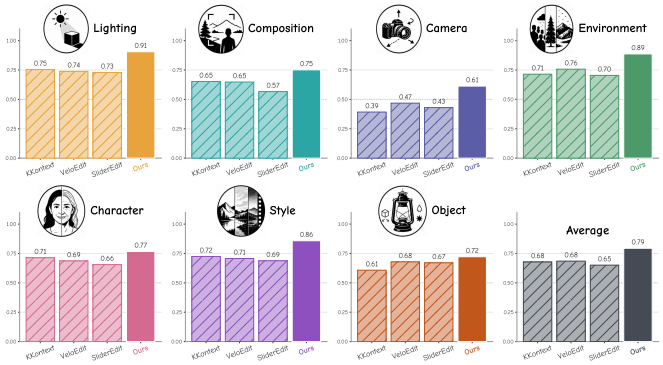

Figure 2: Comparison across SlideBench categories, spanning diverse continuous image editing scenarios, enabling a comprehensive evaluation of continuous controllability.

edits that progressively introduce or strengthen visual content, such as crowd density or aging; and fine-grained character edits over facial attributes, where precise control over small visual changes is particularly important. Together, these categories provide broad coverage of continuous editing behavior and enable systematic comparison across methods. Figure 2 showcases the benchmark categories and compares the performance of baselines. For Additional details, refer to Appendix C.1.

3.2 Evaluation Metrics

We evaluate each slider along two complementary dimensions: *monotonic progression*, which measures whether the strength of the edit increases consistently as the slider advances, and *continuity*, which measures whether these changes occur smoothly without abrupt transitions. For multi-slider models, we additionally evaluate *disentanglement*, measuring whether varying one slider preserves the remaining attributes. We describe all the metrics below:

Monotonic Progression. Let $I_1, \ldots, I_T$ denote the frames of a slider ordered by increasing edit strength. For each consecutive pair (I_t, I_{t+1}), we ask a set of powerful VLM judges which image applies the target edit more strongly. The two images are presented in randomized order to eliminate positional bias, and each comparison is scored as 1 if the later frame is judged stronger and 0 otherwise (including ties). The monotonic progression score is then the average accuracy over all consecutive pairs and all judges,

$$\mathrm{MP} = \frac{1}{|\mathcal{J}|(T-1)} \sum J \in \mathcal{J} \sum_{t=1}^{T-1} m_t^J \in [0,1], \qquad m_t^J = \begin{cases} 1, & \text{if } \hat{y}_t^J = t+1, \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

where higher values indicate more consistently increasing edit strength along the slider. We adopt pairwise VLM judgments because they capture fine-grained perceptual differences in edit strength that image–text similarity scores often fail to reflect in the general case.

Continuity. Let d_t^φ denote the distance between consecutive frames I_t and I_{t+1} under the feature representation induced by a feature extractor φ (e.g., DINOv2). We define the per-slider continuity score as: $U^\varphi = \exp\big(-\mathrm{std}\big(\{d_t^\varphi\}_{t=1}^{T-1}\big)\big) \in (0,1]$, where $U^\varphi = 1$ indicates perfectly uniform transitions. Since continuity alone cannot distinguish a smooth edit from a no-op slider (i.e., one that produces little or no visible change), we weight each sample by its overall edit strength $e \in [0,1]$, which measures how strongly the final frame exhibits the intended edit relative to the source image, as judged by a powerful VLM: $\bar{U}^\varphi = \frac{1}{M} \sum_{i=1}^{M} U_i^\varphi \, e_i$. The final continuity score $\bar{U}$ averages over all feature representations.

Multi-slider Disentanglement. We evaluate whether changing one slider preserves the remaining attributes. For each slider k, we fix all other sliders to obtain a 1D slice, and measure the alignment of every image in the slice with the captions corresponding to each non-varying attribute $j \neq k$. Disentanglement is defined as the mean absolute deviation of these alignment scores along the slice: $D_{k \to j} = \frac{1}{A} \sum_{a=1}^{A} \left| s_j(I_a^{(k)}) - \bar{s}_j^{(k)} \right|$, where $s_j(\cdot)$ is the alignment score for attribute j and $\bar{s}_j^{(k)}$ is its mean over the slice. We average $D_{k \to j}$ over all slices and slider pairs (k, j), where lower values indicate better disentanglement.

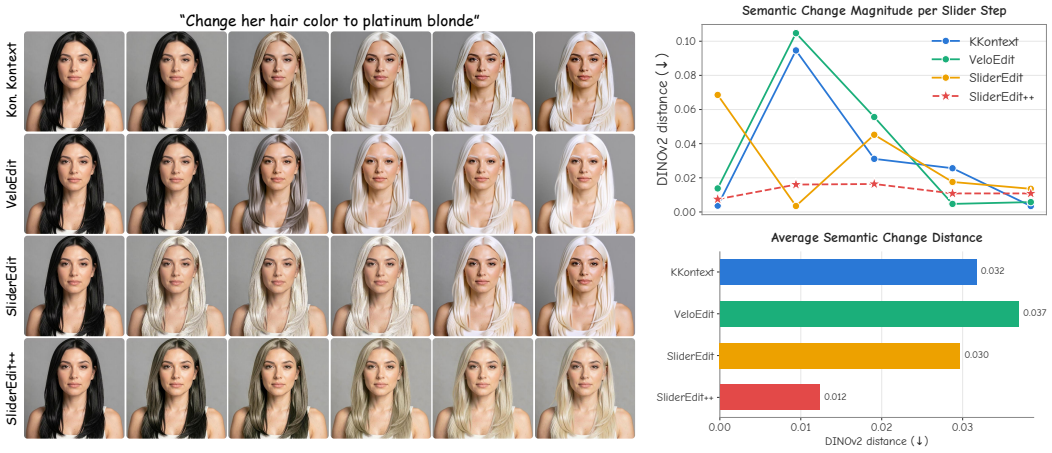

Figure 3: **Slider Discontinuity Analysis**. Baseline image-editing sliders often exhibit abrupt transitions during interaction. The semantic change distance on the right quantifies this instability. In contrast, SliderEdit++ produces smoother sliders by addressing the issue during training.

4 Method: Improving Continuity and Disentanglement in Sliders

Although recent image editing sliders provide intuitive and seemingly continuous control over semantic attributes, we observe that their editing trajectories are often far from smooth. As illustrated in Figure 3, traversing a slider frequently produces abrupt semantic transitions, where a small change in the slider value unexpectedly results in a disproportionately large change in the generated image (e.g., the sudden hair color transition in each method). Such discontinuities break the notion of continuous control, making fine-grained editing unstable and unpredictable. This phenomenon is not limited to qualitative observations. We quantify the semantic change between consecutive slider positions using the DINOv2 feature distance. Ideally, a well-behaved editing slider should induce *uniform semantic progress*, where consecutive slider steps produce similar semantic changes throughout the entire trajectory. Instead, existing methods exhibit large fluctuations in semantic distance, revealing irregular editing dynamics (Figure 3, top right). These results suggest that current slider-based editors learn an inconsistent control space, where equal slider increments do not correspond to equal semantic changes.

Motivated by these observations, we propose SliderEdit++, which explicitly regularizes the editing trajectory during training to encourage smooth and consistent semantic progression. As a result, our method produces stable, predictable slider behavior, yielding visually smooth transitions and substantially more uniform semantic changes across the entire editing trajectory. In the following sections, we describe the key components of our approach and show how they collectively enable the learning of a smooth and consistent editing space.

4.1 Preliminaries: Continuous Edit Slider using SliderEdit

We build on SliderEdit (Zarei et al., 2026), which learns continuous editing controls by training adapters to progressively suppress the effect of an editing instruction. Specifically, let $\epsilon(Z_t, X, P)$ denote the frozen base editing model's velocity prediction for noisy latent Z_t at timestep t, given the source image latent X and a text instruction P. SliderEdit injects a low-rank adapter at every layer ℓ, $\Delta W^\ell = B^\ell A^\ell$, and defines a *scaled* adapted model M_θ^α that applies $\alpha\,\Delta W^\ell$ at each layer, with $\alpha \in [0, 1]$ acting as the slider value: $\alpha = 0$ leaves the base model's instruction-following edit untouched, while $\alpha = 1$ is trained to fully *suppress* it. Concretely, the adapter is trained with a Partial Prompt Suppression loss,

$$\mathcal{L}_{\text{PPS}} = \mathbb{E}_{Z_t,t}\left\|\epsilon_{M_\theta^1}(Z_t, X, P) \;-\; \epsilon(Z_t, X, P_\emptyset)\right\|^2, \tag{2}$$

where $P_\emptyset$ is a neutral, edit-free prompt (e.g. "keep the image the same"), and the right-hand side is a stop-gradient target computed with the base model alone. In words: at full strength, the adapter is pulled toward reproducing what the *unedited* model would have predicted, so that sweeping α from $0 \to 1$ at inference sweeps the output continuously from "fully edited" to "neutral". In the multi-instruction setting, the same idea is applied *per instruction* rather than to the whole prompt.

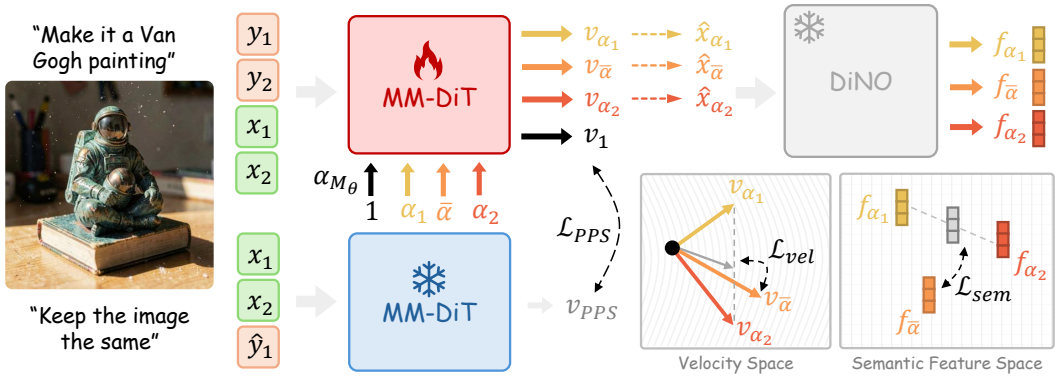

Figure 4: Overall pipeline of SliderEdit++ for improving the continuity in image editing sliders.

This objective supervises only the endpoints $\alpha = 0$ and $\alpha = 1$. Intermediate controls arise simply by continuously scaling the learned LoRA weights, without any direct supervision. Consequently, as shown in Figure 3, the resulting trajectories can exhibit discontinuities, which we address with the regularizers introduced below.

4.2 Towards Solving Continuity Issues in Sliders

Because $\mathcal{L}_{\text{PPS}}$ only anchors the discrete endpoints $\alpha \in \{0, 1\}$, the intermediate trajectory $\alpha \mapsto \epsilon_{M_\theta^\alpha}$ remains unconstrained. Consequently, two adapters with identical endpoint behavior can deviate arbitrarily at intermediate values, producing abrupt perceptual jumps rather than a smooth progression. To enforce uniform, monotonic behavior across $\alpha \in [0, 1]$, we introduce three complementary regularizers, each targeting a different reason the interior of the slider can misbehave.

Velocity-space midpoint regularizer. SliderEdit observes that *explicit* classifier-free guidance (Ho & Salimans, 2022) – linearly blending two model outputs, $\epsilon = (1-\alpha)\,\epsilon_{\text{edit}} + \alpha\,\epsilon_{\text{neutral}}$ – gives highly smooth transitions almost by construction, since it is a literal convex combination of two vectors. Scaling LoRA *weights* by α is not the same thing: $\epsilon_{M_\theta^\alpha}$ is a highly nonlinear function of α once it passes through many transformer layers, so linear interpolation of the *weights* gives no guarantee of linear interpolation of the *output*. We inject that missing guarantee directly as a training signal. For two random scales $\alpha_1, \alpha_2 \sim \mathcal{U}[0, 1]$ and their midpoint $\alpha_m = \frac{1}{2}(\alpha_1 + \alpha_2)$, we penalize

$$\mathcal{L}_{\text{vel}} = \mathbb{E}\left\| \epsilon_{M_\theta^{\alpha_m}}(Z_t, X, P) - \tfrac{1}{2}\big(\text{sg}[\epsilon_{M_\theta^{\alpha_1}}(Z_t, X, P)] + \text{sg}[\epsilon_{M_\theta^{\alpha_2}}(Z_t, X, P)]\big) \right\|^2, \tag{3}$$

where $\text{sg}[\cdot]$ denotes the stop-gradient operator. So the loss only shapes the *interior* of the curve and leaves the endpoints – already anchored by $\mathcal{L}_{\text{PPS}}$ – untouched. Enforcing the midpoint's prediction to equal the average of its neighbors' predictions, for randomly sampled neighbors everywhere in $[0, 1]$, is the discrete analogue of asking $\alpha \mapsto \epsilon_{M_\theta^\alpha}$ to be affine: exactly the property a trustworthy slider needs, equal steps in α should produce equal, predictable steps in the output.

Semantic feature-space regularizer. Smoothness in raw velocity space does not imply smoothness in what the user actually perceives. The map from latent velocity, through decoding, to a visible attribute (e.g. "how much is the person smiling") is highly nonlinear, and a small residual in ϵ can be perceptually invisible in one region of latent space and produce a visible jump in another. Since the slider is judged by what it looks like, not by its latent representation, we apply the same midpoint-consistency idea directly in a semantic feature space. Let $\hat{X}_0^\alpha = \mathcal{D}\big(Z_t - \sigma_t\,\epsilon_{M_\theta^\alpha}(Z_t, X, P)\big)$ be the model's one-step clean-image estimate at scale α, decoded to pixels, and let $D(\cdot)$ denote the embedding of a frozen DINOv2 (Caron et al., 2021) feature encoder. We penalize

$$\mathcal{L}_{\text{sem}} = \mathbb{E}\left\| D\big(\hat{X}_0^{\alpha_m}\big) - \tfrac{1}{2}\big(\text{sg}[D(\hat{X}_0^{\alpha_1})] + \text{sg}[D(\hat{X}_0^{\alpha_2})]\big) \right\|^2. \tag{4}$$

This directly regularizes the quantity a user cares about – the perceived attribute – rather than a latent proxy for it, and catches discontinuities that are invisible in velocity space but visible on screen.

Spectral-norm regularizer. SliderEdit reports needing early stopping to keep sliders smooth: continuing to train lets the adapter become extremely good, in raw loss terms, at suppressing the edit, but it does so by making the effective update $\Delta W^\ell = \alpha B^\ell A^\ell$ increasingly "sharp". Concretely,

its spectral norm $\sigma_{\max}(\Delta W^\ell)$ grows, and since the adapter's contribution composes through every subsequent nonlinear layer, a large per-layer operator norm is precisely what turns a small change in α into a disproportionately large, discontinuous change in the output – the network starts behaving like a hard switch rather than a gentle dial. Early stopping controls this only indirectly, by refusing to train long enough for it to happen, which caps edit strength along with it. We instead constrain it explicitly, penalizing the squared spectral norm of every layer's effective update:

$$\mathcal{L}_{\text{spec}} = \frac{1}{L}\sum_{\ell=1}^{L} \sigma_{\max}\big(\alpha\, B^\ell A^\ell\big)^2. \tag{5}$$

This allows training to full convergence while directly bounding adapter sensitivity to α, replacing an ad-hoc heuristic with a principled constraint.

Overall, we train SliderEdit++ by jointly optimizing the base SliderEdit loss and our proposed regularizers: $\mathcal{L} = \mathcal{L}_{\text{PPS}} + \lambda_{\text{vel}}\mathcal{L}_{\text{vel}} + \lambda_{\text{sem}}\mathcal{L}_{\text{sem}} + \lambda_{\text{spec}}\mathcal{L}_{\text{spec}}$. The overall pipeline of our method is illustrated in Figure 4 and Algorithm 1.

4.3 Towards Solving Disentanglement Issues in Multi-Sliders

In multi-attribute editing, a composite prompt contains K sub-instructions $P = \{P_1, \ldots, P_K\}$ modulated by independent sliders $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_K) \in [0,1]^K$. SliderEdit supports this via Selective-Token LoRA (STLoRA), which scales the adapter update at token positions corresponding to P_i by α_i while optimizing a subset-suppression objective $\mathcal{L}_{\text{PPS}}^{\text{multi}}$ (see Appendix B.1 for full details). However, while token-level gating restricts *where* the adapter is active, all instructions share the same LoRA weights and cross-attention mixing. Consequently, suppressing one instruction can inadvertently induce correlated output shifts along another, causing noticeable attribute leakage.

Explicit disentanglement regularizer. To prevent cross-attribute interference, we enforce that the velocity shifts induced by different sliders remain mutually orthogonal. For distinct instructions $a \neq b$, we define the individual effect direction of slider $i \in \{a, b\}$ relative to the unsuppressed baseline ($\boldsymbol{\alpha} = \mathbf{0}$) as:

$$\delta_i = \epsilon_{M_\theta(\mathbf{e}_i)}(Z_t, X, P) - \epsilon_{M_\theta(\mathbf{0})}(Z_t, X, P), \tag{6}$$

where $\mathbf{e}_i$ denotes the one-hot scale vector isolating instruction i at full suppression ($\alpha_i = 1$). Geometrically, true disentanglement requires that isolated changes along attribute a share no directional overlap with attribute b ($\delta_a \perp \delta_b$). We enforce this condition directly by penalizing the absolute cosine similarity between effect directions:

$$\mathcal{L}_{\text{dis}} = \mathbb{E}_{a\neq b}\left[\frac{|\langle \delta_a, \delta_b\rangle|}{\|\delta_a\|\,\|\delta_b\|}\right]. \tag{7}$$

Minimizing $\mathcal{L}_{\text{dis}}$ alongside $\mathcal{L}_{\text{PPS}}^{\text{multi}}$ pushes distinct editing trajectories toward orthogonality in velocity space, ensuring that modulating one slider alters the target attribute without leaking into others.

5 Experiments

We comprehensively evaluate SliderEdit++ on SlideBench through quantitative comparisons, qualitative results, and a human study. We compare against existing slider-based baselines, analyze the trade-off between controllability and inference speed, and ablate our proposed regularizers to isolate their contribution to smoothness and disentanglement.

5.1 Setting

We build SliderEdit++ on the FLUX-Klein 4B backbone, injecting low-rank adapters of rank 32 at every linear layer. All models are trained with the base $\mathcal{L}_{\text{PPS}}$ objective augmented with our regularizers, using loss weights $\lambda_{\text{vel}} = 10.0$, $\lambda_{\text{sem}} = 1.0$, and $\lambda_{\text{spec}} = 0.1$; the semantic regularizer uses a frozen DINOv2-L encoder. We train for 1000 steps with Adam at learning rate 4×10^{-5} and a global batch size of 32 on 8×H100. Unlike the baseline, we train to convergence rather than relying on early stopping, since the spectral-norm regularizer controls slider sharpness explicitly. At inference we use first-order Euler ODE solver with 8 denoising steps. For evaluation, each slider is rendered at $A = 8$ uniformly spaced positions $\alpha_i \in [0, 1]$, and all metrics are averaged over the 285 samples of the benchmark across its 7 categories. We compare against KontinuousKontext (Parihar et al., 2026), SDE-Edit (Meng et al., 2021), VeloEdit (Li et al., 2026), the original SliderEdit (Zarei et al., 2026), and our re-trained SliderEdit (Klein-4B) baseline, which applies the SliderEdit objective to the same backbone as ours and isolates the effect of the regularizers alone.

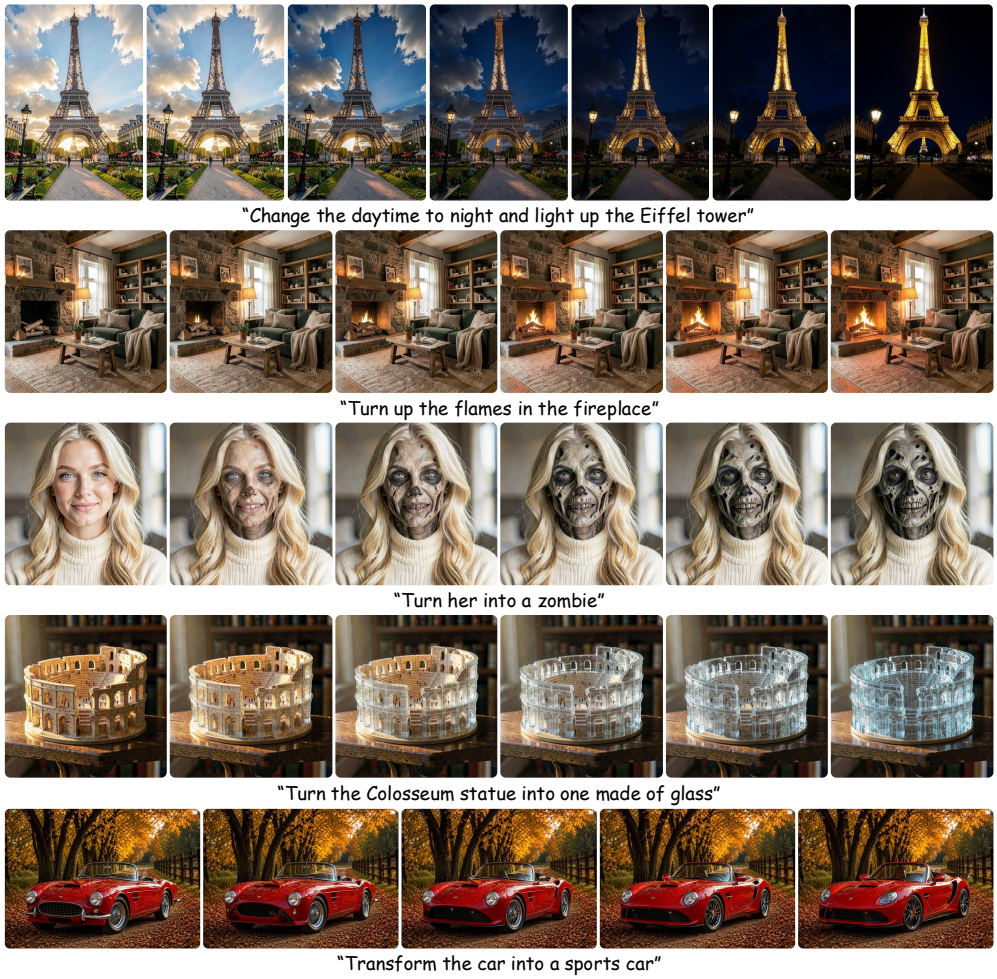

Figure 5: **SliderEdit++** displays strong continuity across different transition levels of the slider.

5.2 Single Slider

We first evaluate SliderEdit++ on SlideBench using the dataset and metrics from Section 3. Table 1 compares against existing slider-based methods. It consistently outperforms all baselines in both monotonic progression and frame uniformity, yielding an overall 18.5% relative improvement over the strongest prior method. This advantage is also consistent across editing categories: Figure 2 shows that SliderEdit++ achieves the best performance in every category of SlideBench. Importantly, this improved controllability does not come at the expense of efficiency; Figure 1 shows that SliderEdit++ achieves the best performance–latency trade-off, operating approximately 3× to 20× faster than competing slider methods. Qualitatively, Figure 6(a) shows that competing methods often exhibit abrupt visual changes along the traversal, whereas our method produces smoother and more gradual edit progression. Additional examples in Figure 5 demonstrate this behavior across diverse global and localized edits, including lighting, material, object, and character transformations. To validate that our automatic metrics reflect human perception, we conduct a human study where we ask annotators to select the smoothest and most edit-faithful slider sweep among randomized methods. Figure 8 (a) shows that SliderEdit++ outperforms all baselines by a substantial margin.

Furthermore, to isolate the effect of our proposed regularizers, we ablate them on the same backbone and training setup by enabling each term individually and in combination. As shown in Table 2, every regularizer improves slider quality over the unregularized baseline, while combining all three yields the strongest performance, indicating that their effects are complementary. The qualitative comparisons in Figures 6(b) and 12 further illustrate this effect: without regularization, edits often emerge abruptly at isolated slider positions, whereas our full model introduces them progressively and smoothly across the trajectory. Finally, the human study in Figure 8 (b) shows a significant preference for our method over the baseline.

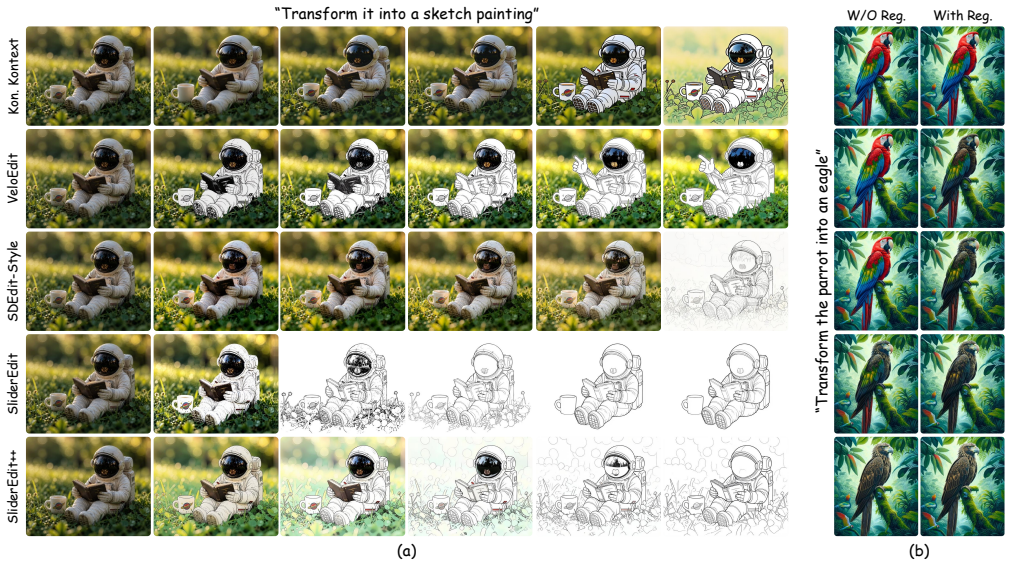

Figure 6: **Qualitative comparison with baselines and the effect of our proposed regularizers.** (a) demonstrates that SliderEdit++ produces smoother trajectories than competing baselines, while (b) illustrates the impact of our regularizers encouraging significantly smoother semantic transitions.

	Monotonic Progression			Frame Uniformity × VLM Edit Strength					Overall
	GPT-5	Opus-4.8	Avg	LPIPS	CLIP	SigLIP	DINOv2	Avg	Score
KontinuousKontext	0.7073	0.6992	0.7033	0.6223	0.6626	0.6630	0.6585	0.6516	0.6774
TextSteer	0.4980	0.5120	0.5050	0.6020	0.6320	0.6510	0.6530	0.6345	0.5690
SDE-Edit	0.5363	0.5820	0.5591	0.6528	0.6630	0.6634	0.6590	0.6596	0.6094
VeloEdit	0.6837	0.7098	0.6967	0.6381	0.6845	0.6868	0.6816	0.6727	0.6847
Explicit CFG	0.6381	0.6406	0.6393	0.6400	0.6795	0.6810	0.6746	0.6688	0.6541
SliderEdit	0.6135	0.6311	0.6223	0.6394	0.6925	0.6949	0.6891	0.6790	0.6506
Ours	**0.7801**	**0.7890**	**0.7845**	**0.8272**	**0.8431**	**0.8443**	**0.8399**	**0.8386**	**0.8116**
% gain	+10.3%	+11.2%	+11.5%	+26.7%	+21.7%	+21.5%	+21.9%	+23.5%	+18.5%

Table 1: Comparison on monotonic progression and frame uniformity. Best and second-best results are shown in **bold** and underline; the final row reports our relative gain over the strongest baseline.

Method	$\mathcal{L}_{\text{sem}}$	$\mathcal{L}_{\text{vel}}$	$\mathcal{L}_{\text{spec}}$	MP_{avg}	$\bar{U}_{avg}$	Overall
Baseline	✗	✗	✗	0.7161	0.7258	0.7209
+ $\mathcal{L}_{\text{sem}}$	✓	✗	✗	0.7643	0.8060	0.7851
+ $\mathcal{L}_{\text{vel}}$	✗	✓	✗	0.7552	0.7901	0.7726
+ $\mathcal{L}_{\text{spec}}$	✗	✗	✓	0.7421	0.7964	0.7692
+ All Regularizers	✓	✓	✓	**0.7845**	**0.8386**	**0.8116**

Table 2: Effect of each regularizer on SliderEdit. $\mathcal{L}_{\text{sem}}$: DINO-based semantic midpoint regularizer, $\mathcal{L}_{\text{vel}}$: latent velocity midpoint regularizer, $\mathcal{L}_{\text{spec}}$: spectral norm regularizer on the LoRA matrices.

5.3 Multi Slider

We next evaluate the multi-slider setting, training SliderEdit++ with the multi-instruction $\mathcal{L}_{\text{PPS}}$ objective together with our smoothness and disentanglement regularizers. Table 3 compares against the base SliderEdit model and shows consistent improvements across the evaluated metrics, increasing the overall score from 0.7097 to 0.8811. Figure 9 further illustrates the difference: SliderEdit exhibits cross-slider interference, where adjusting one instruction can unintentionally affect another, whereas SliderEdit++ produces smoother and more disentangled controls, with changes to one slider having minimal effect on the others.

5.4 Extending to Mask-Conditioned Sliders

We extend SliderEdit++ to *spatially localized* editing, enabling a continuum of edits restricted to a user-specified region while the complement is preserved. Given a source image $\mathbf{x}$, a binary mask

	Monotonic Progression			Frame Uniformity × VLM Edit Strength					Disentanglement				Overall
	GPT-5	Opus-4.8	Avg	LPIPS	CLIP	SigLIP	DINOv2	Avg	CLIP	SigLIP	BLIP	Avg	Score
SliderEdit	0.6511	0.6687	0.6599	0.7294	0.7638	0.7708	0.7740	0.7595	0.9930	0.9939	0.9694	0.9854	0.7097
SliderEdit++	**0.7810**	**0.7918**	**0.7864**	**0.9683**	**0.9747**	**0.9782**	**0.9818**	**0.9757**	**0.9932**	**0.9957**	**0.9719**	**0.9869**	**0.8811**

Table 3: Comparison of multi-slider baselines on monotonic progression, VLM-weighted frame uniformity, and disentanglement.



Figure 7: Extending SliderEdit++ to spatially localized editing, enabling continuous control within user-specified regions while preserving surrounding content.

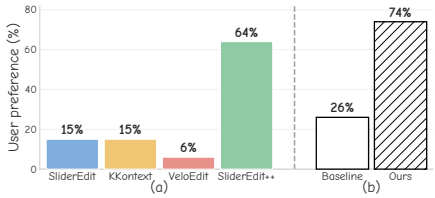

Figure 8: Human preference comparing our method with baselines (a) and our training strategies with the baseline (b).

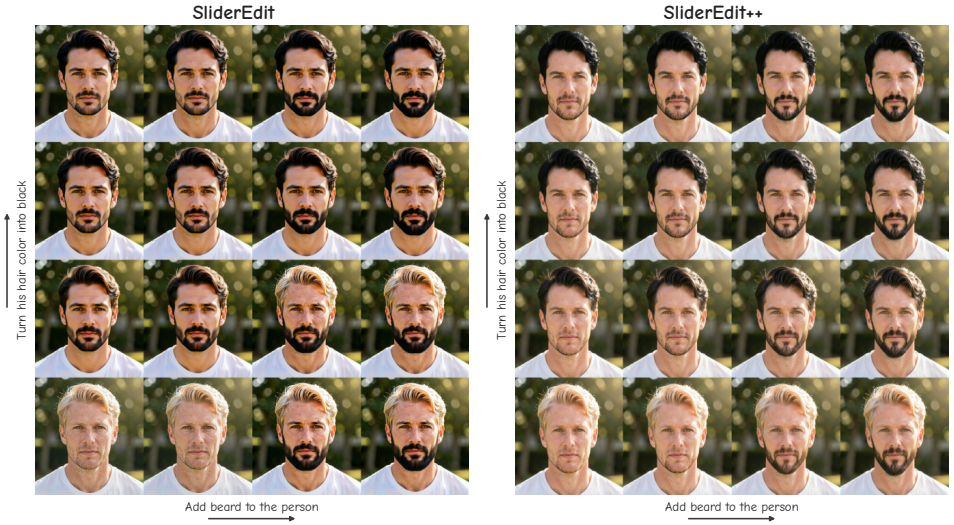

Figure 9: Qualitative results of SliderEdit++ showing strong disentanglement in multi-slider setting.

$\mathbf{m} \in \{0,1\}^{H\times W}$ (with 1 denoting edit and 0 preserve), and an instruction, the edit magnitude is governed by the LoRA adapter scale α. To confine the edit, we adopt a masked latent-blending scheme: the model performs a full-image edit, but during the final k denoising steps we replace the out-of-mask latents with a re-noised copy of the source,

$$z_t \leftarrow \mathbf{m} \odot z_t + (1-\mathbf{m}) \odot \big[(1-\sigma_t)\, z_0 + \sigma_t\, \boldsymbol{\epsilon}\big], \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{8}$$

where z_0 is the (VAE-)encoded source and σ_t the noise level at timestep t, and $\odot$ denotes element-wise (spatially broadcast) multiplication. Blending only the last k steps (we use $k = 2$) preserves a strong in-region edit while yielding a smooth latent boundary. Figure 7 shows a qualitative example in which the user specifies an edit region and instruction, and SliderEdit++ provides continuous control over the localized edit. We provide extensive qualitative results in Figure 17.

6 Conclusion

In this paper, we first introduced SlideBench, a comprehensive benchmark for evaluating continuous image editing, and showed that existing slider-based methods often suffer from discontinuous trajectories, abrupt visual jumps, and cross-slider interference. We then proposed SliderEdit++, which introduces complementary regularizers to explicitly improve slider smoothness, uniform progression, and disentanglement during training. On SlideBench, SliderEdit++ achieves state-of-the-art performance, improving the overall score by 18.5% over the strongest baseline while operating approximately 3× to 20× faster, enabling more reliable and responsive continuous image editing.

AI USE STATEMENT

Generative AI tools were used to assist with polishing the writing and improving the clarity of the manuscript. They were also used to generate portions of the dataset by expanding author-defined specifications and examples. All generated data were manually reviewed and verified by the authors before use. Generative AI was not used to formulate the research hypotheses or interpret the experimental results. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Rameen Abdal, Peihao Zhu, Niloy J Mitra, and Peter Wonka. Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. *ACM Transactions on Graphics (ToG)*, 40(3):1–21, 2021.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Stefan Andreas Baumann, Felix Krause, Michael Neumayr, Nick Stracke, Melvin Sevi, Vincent Tao Hu, and Björn Ommer. Continuous, subject-specific attribute control in t2i models by identifying semantic directions. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13231–13241. IEEE, 2025.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *2021 IEEE/CVF international conference on computer vision (ICCV)*, pp. 9630–9640. IEEE, 2021.

Pin-Yen Chiu, I Fang, Jun-Cheng Chen, et al. Text slider: Efficient and plug-and-play continuous concept control for image/video synthesis via lora adapters. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 613–622, 2026.

Yusuf Dalva, Kavana Venkatesh, and Pinar Yanardag. Fluxspace: Disentangled semantic editing in rectified flow transformers. *arXiv preprint arXiv:2412.09611*, 2024.

Yigit Ekin and Yossi Gandelsman. The unreasonable effectiveness of text embedding interpolation for continuous image steering. *arXiv preprint arXiv:2603.17998*, 2026.

Taichi Endo, Guoqing Hao, and Kazuhiko Sumi. Flowslider: Training-free continuous image editing via fidelity-steering decomposition. *arXiv preprint arXiv:2604.02088*, 2026.

Rohit Gandikota, Joanna Materzyńska, Tingrui Zhou, Antonio Torralba, and David Bau. Concept sliders: Lora adaptors for precise control in diffusion models. In *European Conference on Computer Vision*, pp. 172–188. Springer, 2024.

Rohit Gandikota, Zongze Wu, Richard Zhang, David Bau, Eli Shechtman, and Nick Kolkin. Sliderspace: Decomposing the visual capabilities of diffusion models. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15994–16003. IEEE, 2025.

Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls. *Advances in neural information processing systems*, 33:9841–9850, 2020.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Sy2fzU9gl.

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

Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In *International conference on machine learning*, pp. 2649–2658. PMLR, 2018.

Black Forest Labs. FLUX.2: Frontier Visual Intelligence. https://bfl.ai/blog/flux-2, 2025.

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

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International conference on machine learning*, pp. 12888–12900. PMLR, 2022.

Zongqing Li, Zhihui Liu, Yujie Xie, Shansiyuan Wu, Hongshen Lv, and Songzhi Su. Veloedit: Training-free consistent and continuous instruction-based image editing via velocity field decomposition. 2026.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021.

Rishubh Parihar, Or Patashnik, Daniil Ostashev, Venkatesh Babu Radhakrishnan, Daniel Cohen-Or, and Kuan-Chieh Jackson Wang. Kontinuous kontext: Continuous strength control for instruction-based image editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 37929–37939, 2026.

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

Yujun Shen and Bolei Zhou. Closed-form factorization of latent semantics in gans. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1532–1540. IEEE, 2021.

Yujun Shen, Ceyuan Yang, Xiaoou Tang, and Bolei Zhou. Interfacegan: Interpreting the disentangled face representation learned by gans. *IEEE transactions on pattern analysis and machine intelligence*, 44(4):2004–2018, 2020.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

Deepak Sridhar and Nuno Vasconcelos. Prompt sliders for fine-grained control, editing and erasing of concepts in diffusion models. In *European Conference on Computer Vision*, pp. 17–31. Springer, 2024.

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

Arman Zarei, Samyadeep Basu, Mobina Pournemat, Sayan Nag, Ryan A Rossi, and Soheil Feizi. Slideredit: Continuous image editing with fine-grained instruction control. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14430–14439, 2026.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11941–11952. IEEE, 2023.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE/CVF conference on computer vision and pattern recognition*, pp. 586–595. IEEE, 2018.

Algorithm 1 Training SliderEdit++ (SliderEdit in gray, our additions in black)

Require: $\epsilon(Z, X, P)$: Image Editing Model, M_θ: Trainable Adapter, $\{(X_{\text{orig}}^{(i)}, \mathcal{P}^{(i)})\}$: Dataset, ϕ: DINOv2 encoder, $\lambda_{\text{vel}}, \lambda_{\text{sem}}, \lambda_{\text{spec}}, \lambda_{\text{dis}}$

for each training step **do**
  $X_{\text{orig}}, \mathcal{P} = \{\mathcal{P}_1, \ldots, \mathcal{P}_K\} \leftarrow$ Sample a Data
  $\varepsilon \sim \mathcal{N}(0, I), t \sim \mathcal{U}[0, 1]$
  if use $\mathcal{L}_{\text{SPPS}}$ **then**
    $\mathcal{P} = \{\mathcal{P}_1'\}$
  end if
  $Z \leftarrow (1-t)\varepsilon + tX_{\text{orig}}$
  $\mathcal{P}_i \leftarrow$ Random target instruction from $\mathcal{P}$ to suppress
  $v^\star \leftarrow \epsilon(Z, X_{\text{orig}}, \mathcal{P} \setminus \{\mathcal{P}_i\})$
  $\hat{v} \leftarrow \epsilon_{M_\theta^1(\mathcal{P}_i)}(Z, X_{\text{orig}}, \mathcal{P})$
  $\mathcal{L}_{\text{PPS}} = \|\hat{v} - v^\star\|^2$
  $\alpha_1, \alpha_2 \sim \mathcal{U}[0, 1], \quad \alpha_m \leftarrow \frac{1}{2}(\alpha_1 + \alpha_2)$
  $v^\alpha \leftarrow \epsilon_{M_\theta^\alpha(\mathcal{P}_i)}(Z, X_{\text{orig}}, \mathcal{P})$ for $\alpha \in \{\alpha_1, \alpha_2, \alpha_m\}$
  $f^\alpha \leftarrow \phi\big(\hat{X}_0(Z, v^\alpha)\big)$
  $\mathcal{L}_{\text{vel}} = \left\| v^{\alpha_m} - \frac{1}{2}(\text{sg}[v^{\alpha_1}] + \text{sg}[v^{\alpha_2}]) \right\|^2$
  $\mathcal{L}_{\text{sem}} = \left\| f^{\alpha_m} - \frac{1}{2}(\text{sg}[f^{\alpha_1}] + \text{sg}[f^{\alpha_2}]) \right\|^2$
  $\mathcal{L}_{\text{spec}} = \frac{1}{L}\sum_\ell \sigma_{\max}(B^\ell A^\ell)^2$
  $\mathcal{L} = \mathcal{L}_{\text{PPS}} + \lambda_{\text{vel}}\mathcal{L}_{\text{vel}} + \lambda_{\text{sem}}\mathcal{L}_{\text{sem}} + \lambda_{\text{spec}}\mathcal{L}_{\text{spec}}$
  if $K \geq 2$ **then**
    $\mathcal{P}_j \leftarrow$ Random instruction from $\mathcal{P} \setminus \{\mathcal{P}_i\}$
    $\delta_k \leftarrow \epsilon_{M_\theta^1(\mathcal{P}_k)}(Z, X_{\text{orig}}, \mathcal{P}) - \epsilon(Z, X_{\text{orig}}, \mathcal{P})$ for $k \in \{i, j\}$
    $\mathcal{L}_{\text{dis}} = |\langle \delta_i, \delta_j \rangle| \,/\, (\|\delta_i\| \|\delta_j\|)$
    $\mathcal{L} \mathrel{+}= \lambda_{\text{dis}}\mathcal{L}_{\text{dis}}$
  end if
  Update θ via gradient descent on $\mathcal{L}$
end for

A Appendix

B Preliminaries

B.1 Continuous Multi-Slider Edit using SliderEdit

In the multi-attribute setting, a single edit instruction decomposes into K independent sub-instructions, $P = \{P_1, \ldots, P_K\}$, each with its own slider value $\alpha_i \in [0, 1]$ (e.g. "add a smile" and "make the hair curlier" issued together). The natural extension of Section 4.2's single global α is to let the adapter's contribution at each layer depend on *token position*: at a token belonging to instruction P_i's span, the effective update is $\alpha_i \, \Delta W^\ell$; elsewhere it is zero. This is the Selective-Token LoRA (STLoRA) mechanism: one shared set of weights realizes K independently addressable sliders in a single forward pass, and the training loss is the same suppression objective as Eq. equation 2, now applied per (randomly chosen) subset of suppressed instructions $\mathcal{S} \subseteq \{1, \ldots, K\}$, with the target computed from the base model on the prompt with exactly those instructions actually removed, $P \setminus \mathcal{S}$:

$$\mathcal{L}_{\text{PPS}}^{\text{multi}} = \mathbb{E}\left\| \epsilon_{M_\theta(\mathcal{S})}(Z_t, X, P) - \epsilon(Z_t, X, P \setminus \mathcal{S}) \right\|^2. \tag{9}$$

Confining each instruction's LoRA activation to its own tokens is already a strong structural bias toward disentanglement – attribute i is only ever edited through its own tokens – and inherits the same continuity treatment as the single-slider case at the per-instruction level.

C SLIDEBENCH

C.1 DATASET

To systematically evaluate continuous image editing, we construct a benchmark spanning a broad range of attributes that users may wish to adjust continuously in practice. The dataset is organized into seven high-level categories—*Lighting*, *Composition*, *Camera*, *Environment*, *Character*, *Style*, and *Object*—covering 57 fine-grained subcategories, with five samples per subcategory. Each subcategory defines a distinct continuous editing axis, such as *time of day*, *crowd density*, *smile intensity*, *color saturation*, or *object aging*. Formally, each sample is represented as a triplet (c, I, p), where c is a source caption, I is the corresponding source image, and p is the editing instruction. The caption specifies one endpoint of the editing axis and is used to generate the source image, while the instruction defines the desired transformation toward the opposite endpoint. For example, a source caption may describe "a village street photographed at golden hour with long warm shadows", while the corresponding instruction requests "changing the scene to midnight with a dark sky and dim street lamps". Within each subcategory, we instantiate the same editing axis across diverse visual contexts to evaluate whether continuous control generalizes beyond a particular scene or image.

The benchmark is designed to capture qualitatively different forms of continuous control. Attributes in *Lighting*, *Style*, and *Camera* naturally admit gradual variation and therefore directly test smooth and monotonic editing behavior. In contrast, *Environment* and *Object* edits, such as crowd density, clutter, aging, and corrosion, require models to progressively introduce or strengthen visual content rather than transition through abrupt discrete changes. *Character*, our largest category with 21 subcategories, further evaluates fine-grained control over facial and bodily attributes, where even small variations can be perceptually significant. Together, these categories provide broad coverage of continuous editing behavior and expose where existing methods succeed or fail across different types of control.

C.2 METRICS

For evaluating *monotonic progression*, we initially explored text-image similarity models such as CLIP Radford et al. (2021), BLIP Li et al. (2022), and SigLIP Zhai et al. (2023). However, we found these models insufficiently sensitive to the fine-grained differences that arise along an editing trajectory. In particular, small changes in attribute strength often produce noisy or inconsistent similarity scores, making these models unreliable for assessing whether an edit progresses smoothly and monotonically. We therefore rely on strong vision-language models (VLMs), following the evaluation protocol described in Section 3.2. We further observed that highly capable VLMs, such as GPT-5 Singh et al. (2025) and Claude Opus 4.8, provide substantially more consistent assessments than smaller open-source alternatives such as LLaVA Liu et al. (2023) and Qwen Bai et al. (2023). We consequently use these stronger models for our main evaluation.

For *frame uniformity*, we measure perceptual and semantic changes between consecutive frames using LPIPS Zhang et al. (2018), CLIP Radford et al. (2021), SigLIP Zhai et al. (2023), and DINOv2 Caron et al. (2021). For *edit strength*, we again use the strong VLM-based evaluator described above, and report the aggregated metric to obtain a more robust assessment.

Finally, *disentanglement* is measured in the feature spaces of CLIP, SigLIP, and BLIP. While these metrics provide a useful proxy for unintended semantic changes, they remain limited by the same lack of sensitivity to fine-grained edits discussed above and should therefore not be viewed as a fully reliable measure of disentanglement. We complement them with human evaluation, while developing more robust feature spaces for capturing such granular editing effects remains an important direction for future work.

D EXPERIMENTS

D.1 RESULTS

Figure 11 presents additional qualitative examples of SliderEdit++, demonstrating smooth and continuous single-slider control over both global edits and fine-grained local changes. Figures 12 and 13 illustrate the effect of our proposed regularizers compared with the base training objective without regularization. While the base objective often exhibits abrupt transitions between neighboring slider values, our regularizers yield substantially more semantically continuous editing trajectories. Fig-

	Monotonic Progression			Frame Uniformity × VLM Edit Strength					Overall
	GPT-5	Opus-4.8	Avg	LPIPS	CLIP	SigLIP	DINOv2	Avg	Score
$T = 8$	0.7589	**0.8000**	**0.7794**	0.7977	0.8130	0.8146	0.8104	0.8089	0.7942
$T = 4$	**0.7629**	0.7950	0.7789	0.8141	0.8275	0.8287	0.8244	0.8237	**0.8013**
$T = 2$	0.7398	0.7845	0.7622	0.8291	**0.8396**	**0.8411**	**0.8378**	**0.8369**	0.7995
$T = 1$	0.6937	0.7639	0.7288	**0.8304**	0.8369	0.8387	0.8361	0.8355	0.7822

Table 4: **Ablating the number of generation steps for slider quality.** Results show that using fewer generation steps has minimal impact on slider performance while significantly improving generation speed.

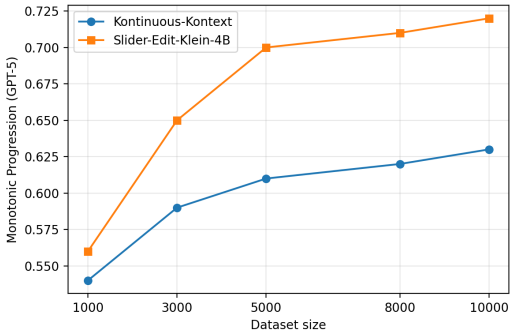

Figure 10: At similar data-budgets, SliderEdit on a stronger base model such as Klein-4B outperforms Kontinuous-Kontext on the same Klein-4B model.

ure 14 provides additional comparisons with existing baselines, where SliderEdit++ consistently produces smoother and more continuous control. Figure 15 presents multi-slider examples, showing that the learned controls remain largely disentangled while maintaining smooth transitions along each editing dimension. Figure 16 demonstrates the applicability of SliderEdit++ to text-rich design posters, where sliders provide continuous control over visual themes while preserving the underlying content and layout. Finally, Figure 17 shows additional examples of our mask-conditioned extension, enabling spatially localized continuous editing within a user-selected region.

D.2 Ablating on few-step generation

We ablate the number of denoising steps used by our few-step model. While our default setting uses $N = 8$ steps, we further reduce it to $N = 4$, 2, and 1. As shown in Table 4, SliderEdit++ maintains strong performance even with substantially fewer steps. This suggests that the inference latency can be further reduced by up to $8\times$, enabling an even more responsive interactive editing experience.

D.3 Ablating the Underlying Slider Training Objective

Our regularizers can be applied on top of different slider training objectives. We compare the two primary candidates, SliderEdit and Kontinuous-Kontext, which differ substantially in their data requirements: Kontinuous-Kontext uses approximately 110k paired samples, while SliderEdit requires $\leq$10k unpaired samples. As shown in Fig. 10, on the few-step Klein-4B backbone, SliderEdit achieves stronger performance than Kontinuous-Kontext at comparable data budgets. We therefore adopt SliderEdit as our base objective, as it provides stronger performance while avoiding the need for paired training data.

E Limitations

While SliderEdit++ consistently improves continuous editing across categories, its performance remains constrained by the capabilities of the underlying image-editing model. In particular, challenging categories such as *Camera* achieve lower absolute performance across all methods, suggesting that some limitations stem from the base model's ability to perform the corresponding edits rather than from slider control itself. Improving the underlying editing model could therefore further enhance SliderEdit++ in these challenging categories.

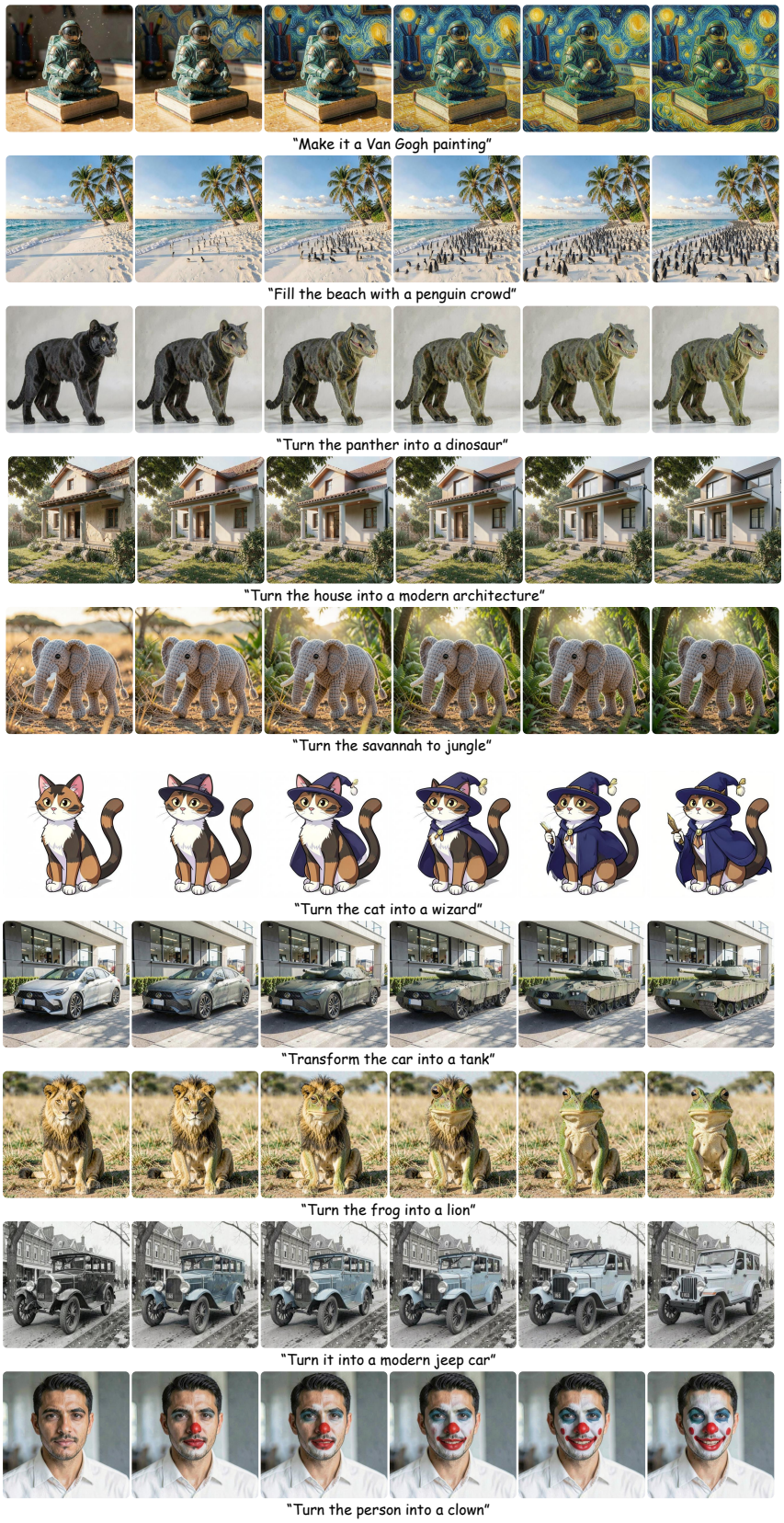

Figure 11: SliderEdit++ displays strong continuity across different transition levels of the slider.

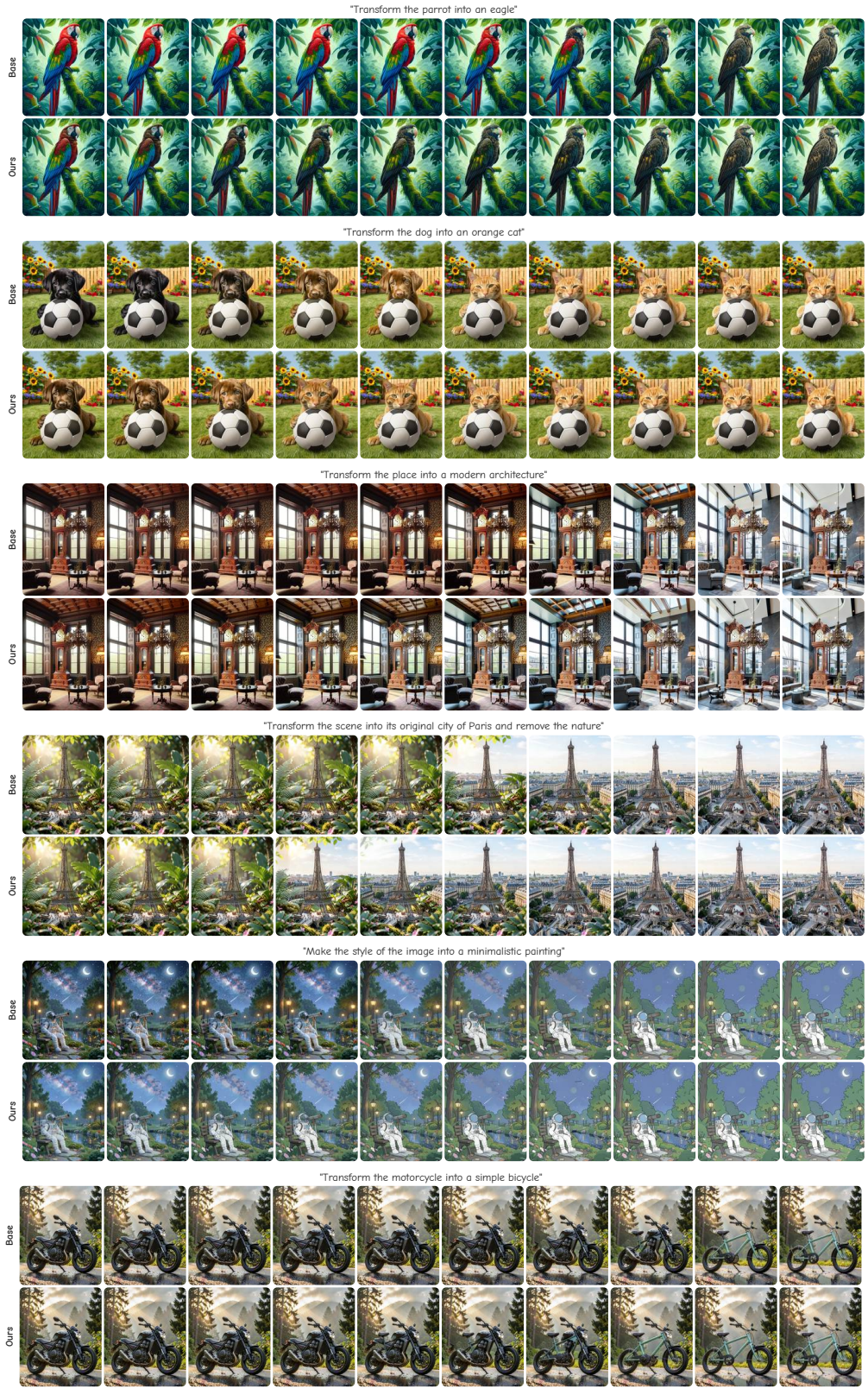

Figure 12: Qualitative comparison demonstrating the effect of our regularizer relative to the baseline SliderEdit method under the same backbone.



Figure 13: Qualitative comparison demonstrating the effect of our regularizer relative to the baseline SliderEdit method under the same backbone.



Figure 14: Qualitative comparison of continuous edit-control methods. SliderEdit++ demonstrates superior performance.

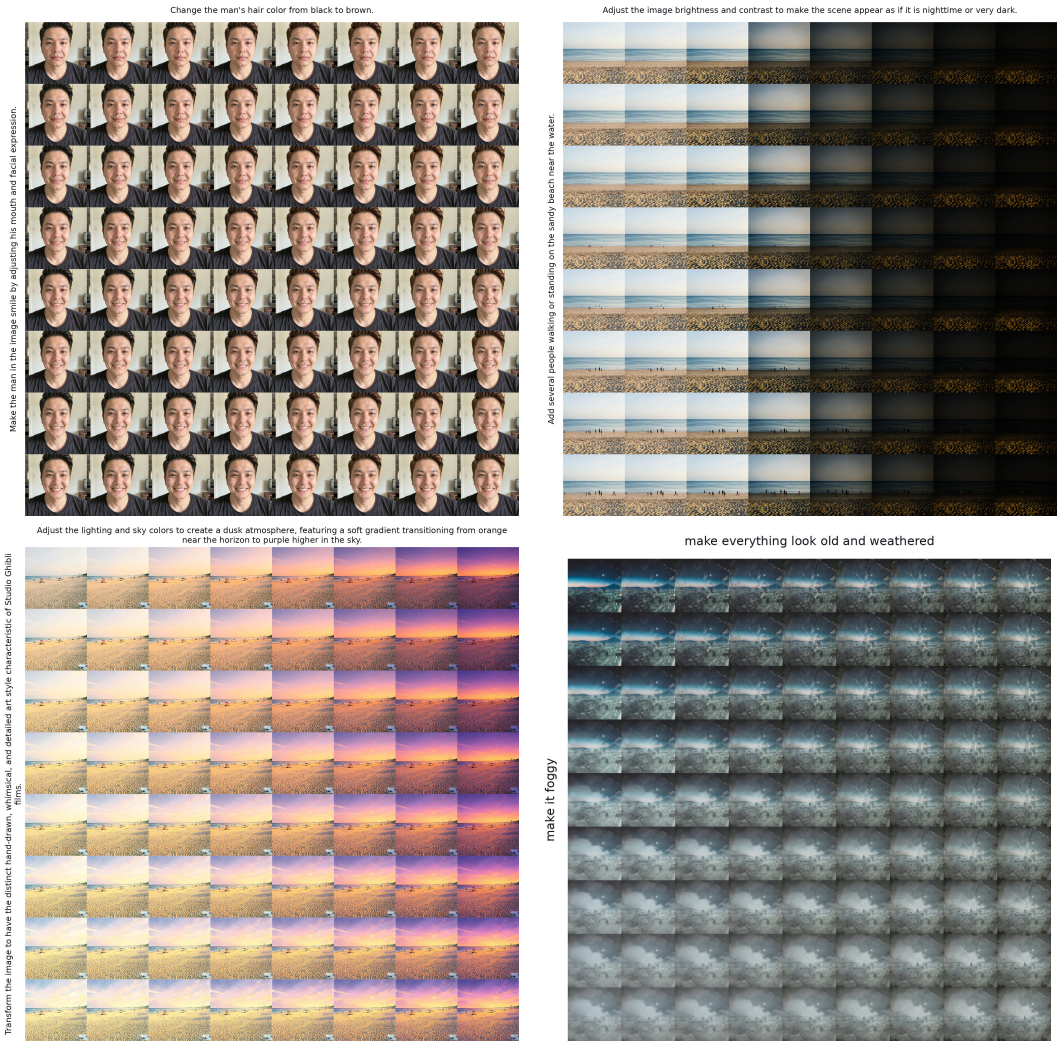

Figure 15: Qualitative Results on Multi-Slider Outputs. Best viewed zoomed in.

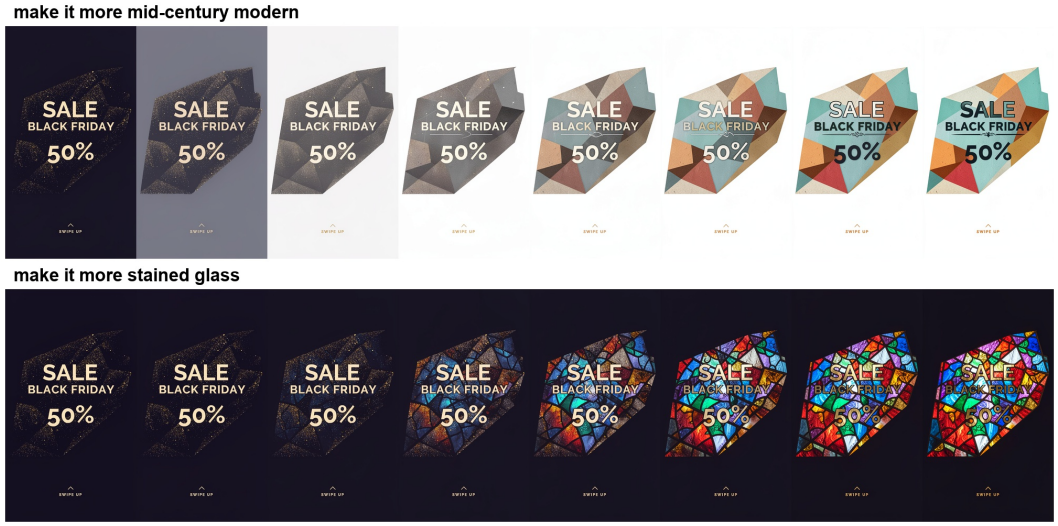

Figure 16: Qualitative Results on Design Images.

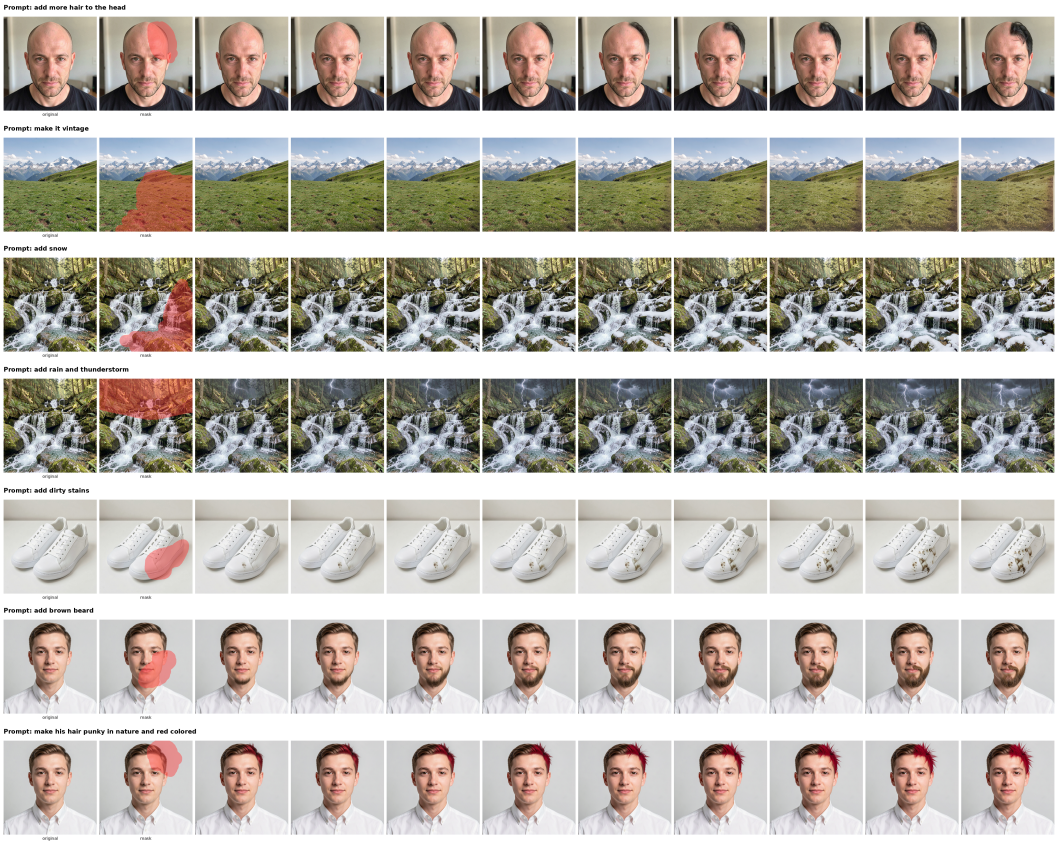

Figure 17: Qualitative Results on Mask-Conditioned SliderEdit++.