Kontinuous Kontext: Continuous Strength Control for Instruction-based Image Editing

Supplementary Material

Contents

A. Appendix

A.1. Implementation Details.

We train a slider projector along with a rank 4 LoRA on all attention layers of the base diffusion transformer model. We train all our models at a resolution of 512×512. After filtering, our dataset consists of $66K$ edit trajectories, along with their edit instructions. We use a threshold value of 0.15 on the KL divergence to filter the non-smooth edit trajectories. We train all models on a 8 NVIDIA $A100$ $(80GB)$ GPU for $110,000$ iterations with an effective batch size of 8 and a constant learning rate of 2×10^{-5}. Training takes about 72 hours to complete. During training, we drop the slider conditioning with 10% of the time. For inference, we use the default Euler scheduler from Flux Kontext and use $T = 28$ inference steps for generation. The generation time is similar to the Flux Kontext model, as we only add a small MLP projector on the base model (details are in Sec.A.13).

A.2. Dataset Generation

In this section, we provide the details about our dataset generation process. Our dataset generation consists of three stages:

i) Generating Image Edit Pairs. We use the Subject200K [11] dataset to source our input images. This dataset consists of a variety of input objects and scenes captured in different environmental conditions. We extract $110K$ source images from this dataset. Next, we generate image-specific edit instructions for source images using Qwen-VLM [1]. For a good diversity of our dataset, we categorize our edit categories into global edits (stylization, scene reimagination, and environment change) and local edits (material and appearance editing, attribute modification, and shape morphing). For each image in the dataset, we randomly sample one of these editing categories and query the Qwen-VLM to generate an edit instruction from that category. We pass the input image along with the system prompt to the Qwen-VLM to generate instructions specific to the image. We use the system prompt for 'appearance change' edit (see **Q-1** on page 3) and query the VLM to generate the edit instruction in a *.json* format. We use similar system prompts for other editing categories. We sample a predefined set $50 - 100$ in-context examples per edit category and randomly sample 4 examples and combine them with the system prompt to generate rich prompts for generating diverse editing instructions. We present samples of in-context examples used in **Q-2** and **Q-3** on pages 3 and 4.

Generating image edits. We use the source images and obtain editing instructions to generate edited versions of the source image using Flux Kontext [2]. Flux-kontext, being a generalist editing model, can generate high-quality edits for the source images given text prompts. However, in some cases, it does not perform the edit and outputs the same input image. We filter such cases in our filtering stage, discussed next. We show some examples of the edits and instructions in Fig. 3.

ii) Generating intermediate edits with Image morphing. Given the source and edited image, we use Freemorph [3] - a training-free Diffusion-based image morphing approach to generate interpolations. These interpolations will be used as ground truths for the edits with intermediate strengths. Freemorph requires an input caption for the two images to be interpolated. To this end, we use LLaVA [9] to generate captions as suggested by their paper. Freemorph first inverts the two images and then interpolates the attention features during denoising. This requires a full denoising process to generate one morph image. In practice, we generate $N = 5$ intermediate morphs between the source and the edited im-

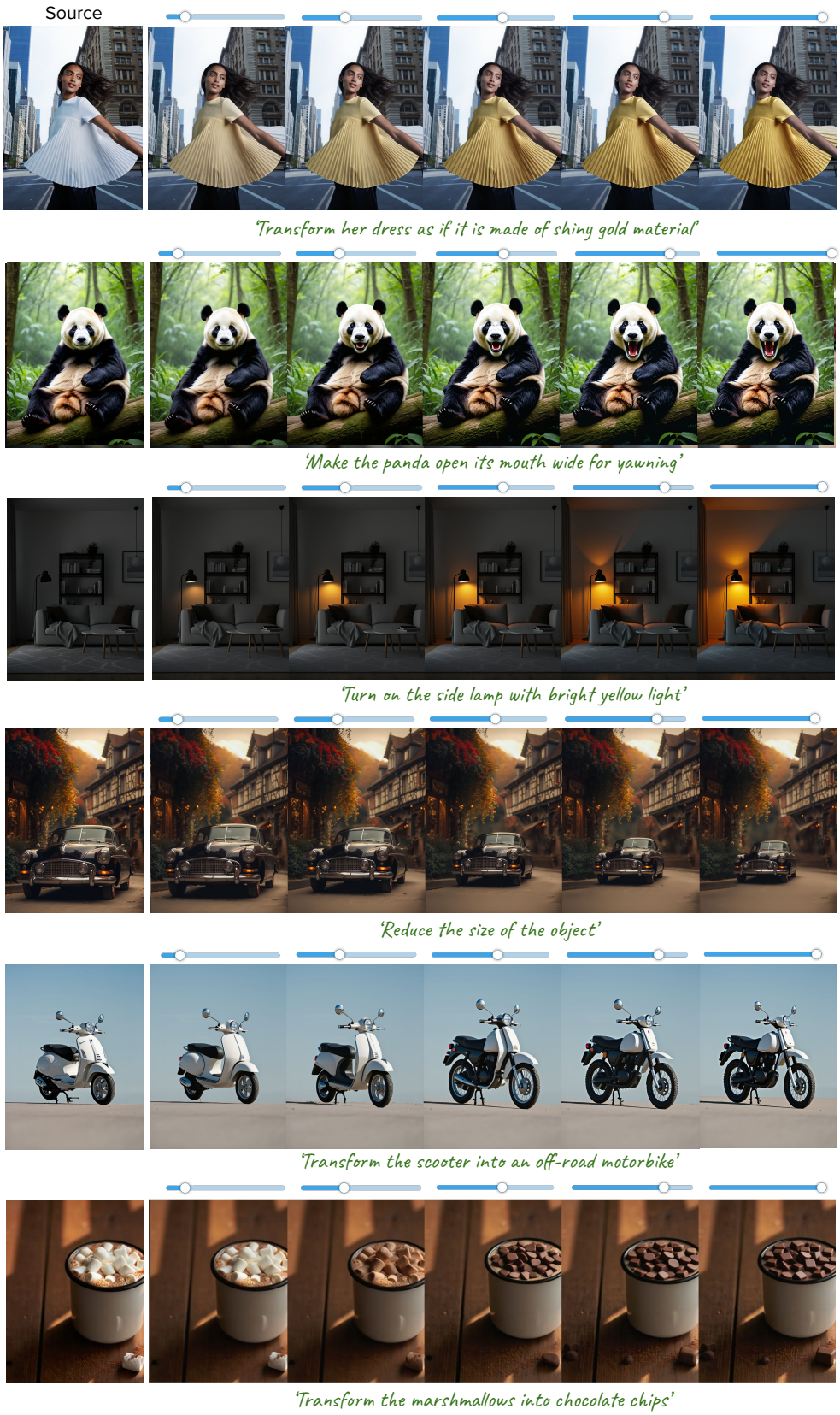

Figure 1. *Kontinuous Kontext* can enable fine-grained control over the edit strength for diverse instruction-driven image editing operations.

Figure 2. Samples from diverse image editing categories in our synthesized dataset spanning global edits (stylization, reimagination, and environment changes), as well as local edits (appearance changes, material changes, attribute editing, and object morphing)

(Q-1) System prompt for generating edit instructions

System Prompt: `You are a professional image editor. Generate an original, diverse, and detailed local appearance change instruction for the given object in the image. Create a unique instruction different in wording and content from the examples.`
`Examples:` {examples}
`Output ONLY a valid JSON object with EXACT keys "category" and "instruction". No additional text or explanation.`
`Example output: {"category": "Appearance_Change", "instruction": "Modify the fabric of the couch to a rich burgundy velvet with gentle sheen."} DO NOT include trailing commas or escape characters.`

age. We use the official code provided by the authors that is built on StableDiffusion-2.1 [10] and use the DDIM scheduler for generation with $T = 50$ steps. All the interpolations were generated at a native resolution 768×768 of SD-2.1 base model. Though Freemorph generates smooth interpolations in most of the cases, it may contain errors in some, that we filter out with our extensive filtering pipeline

iii) Data Filtering. As our data generation process involves the use of several generative models, it may lead to errors in some of the generated data samples. We filter such cases with an automated data filtering logic. Specifically, we remove samples that have - ***a)*** inversion errors, ***b)*** non-smooth trajectories, or ***c)*** weak edits. Some examples are shown in Fig. 4.

a) Poor inversion. We use Freemorph for generating image edit sequences that involve diffusion inversion of the input images. The inversion of the source and the edited image can be error-prone, resulting in an inaccurate training signal. To address this, we compute the LPIPS distance between the original image and its inversion and remove the sample if the LPIPS is above a threshold, indicating poor inversion. Some examples are shown in the first and last row in Fig. 4.

b) Non-smooth trajectories. We also filter out samples where the edit sequence has non-uniform transitions, i.e., change between the two adjacent images is not consistent across the sequence (see rows 3,4,5 in Fig. 4). For a good training sample (x, e, s, y_s), the extent of change between the source x and edit y_s should scale with the edit strength s. Equivalently, the distance between adjacent images in the sequence should remain consistent. We define the sequence of deltas as $D = \{d_{0,1}, d_{1,2}, \ldots, d_{N-1,N}\}$, where $d_{i,i+1}$ is the distance between image y_i and y_{i+1} and measures its uniformity via the KL-divergence from a discrete uniform distribution. In an ideal edit sequence, this sequence of image distances should follow a uniform distribution. We use a KL threshold value of 0.15 to filter such cases with non-smooth trajectories.

c) Weak edits. As we use Flux Kontext to generate the edited version of the source image, we filter out cases where Flux Kontext is not able to perform the edit (see row 2 in Fig. 4). In such cases, Kontext replicates the source image

(Q-2) In context example for local edits

Appearance_change

```
examples = [ "Transform the chair into plush candy-colored marshmallow material
with soft reflections",
"Make the bicycle frame appear as flowing liquid metal with dynamic
highlights",
"Turn the lampshade into glowing crystalline material with internal refracted
light"]
```

Material_change

```
examples = [ "Replace the chair's wooden legs with polished chrome metal,
emphasizing its reflective specularity",
"Make the tabletop appear carved from dark mahogany wood with visible grain and
a semi-matte roughness",
"Transform the bag's fabric into smooth black leather with glossy highlights
and subtle texture"]
```

Attribute_change

```
examples = ["Open the laptop lid halfway to reveal the keyboard",
"Rotate the handlebar of the bicycle by 45 degrees",
"Raise the adjustable lamp arm to maximum height"]
```

Intra_object_morph

```
examples = ["Morph a teapot into a lantern while keeping the spout as a
decorative handle",
"Transform a bicycle into a motorbike with parts composed naturally",
"Morph a chair into a bench while preserving the backrest shape"]
```

as output without any meaningful change. We filter such cases by computing the LPIPS between the source image and the output image and thresholding on LPIPS value.

Deciding filtering thresholds. We select these thresholds by manually inspecting a small set of edit sequences (≈ 100) per edit category. Additionally, one can use a VLM to automatically decide on these thresholds if data needs to be added from new other sources.

A.3. Model Architecture

Our projector is a 4 layer MLP with dimensions $1536 \rightarrow 6144 \rightarrow 6144 \rightarrow 6144 \rightarrow 6144$. The output dimension of $D = 6144$ is divided into two chunks, each of 3072, representing offsets for modulation parameters - Δy_{scale} and Δy_{shift}. The 1536 dimensional input to the model consists of an embedded scale value s of dimension 768 and a pooled CLIP text embedding of dimensions 768. We first apply sinusoidal positional encoding to s to bring it to 128 dimensions, followed by a linear layer to transform it to a similar dimension of 768. The CLIP embedding and the encoded scale embeddings are concatenated and passed as a single input to the projector network. As our slider projector is lightweight and requires negligible computation and inference overhead over the base Flux Kontext model.

A.4. Additional ablations and evaluations.

A.4.1. Inference-time control in modulation space

We perform a simple experiment to analyse the effect of modulation parameters on the edited images. We scale the modulation parameters with $v = (0.5, 1.3)$ for the text token and visualize the generated edit image in Fig. 5. Though the generated edits are diverse for different scale values, the scaling value v does not directly correlate with the strength of the edit. This raises a need of learning a calibrated mapper like our slider projector, which can expose accurate strength control by properly manipulating the modulation parameters.

(Q-3) In context example for global edits

Stylization

```
examples = [ "Render the scene in Studio Ghibli style with dreamy backgrounds
and soft pastel hues",
"Transform the image into Pixar-style 3D animation with vibrant colors and
cinematic lighting",
"Stylize the composition as a Van Gogh oil painting with thick impasto brush
strokes"]
```

Environment_change

```
examples = ["Blanket the entire landscape with fresh, thick snow, covering
trees and rooftops with crystalline frost",
"Transform the scene into a harsh winter blizzard with swirling snow and
reduced visibility",
"Age the entire scene to look like a weathered medieval village with cracked
stone walls"]
```

Scene_reimagination

```
examples = ["Place the entire village on a massive turtle's back slowly moving
through the ocean",
"Transform the bustling marketplace into a floating bazaar carried by hot air
balloons",
"Reimagine the city skyline as colossal crystal formations reflecting rainbow
light"]
```

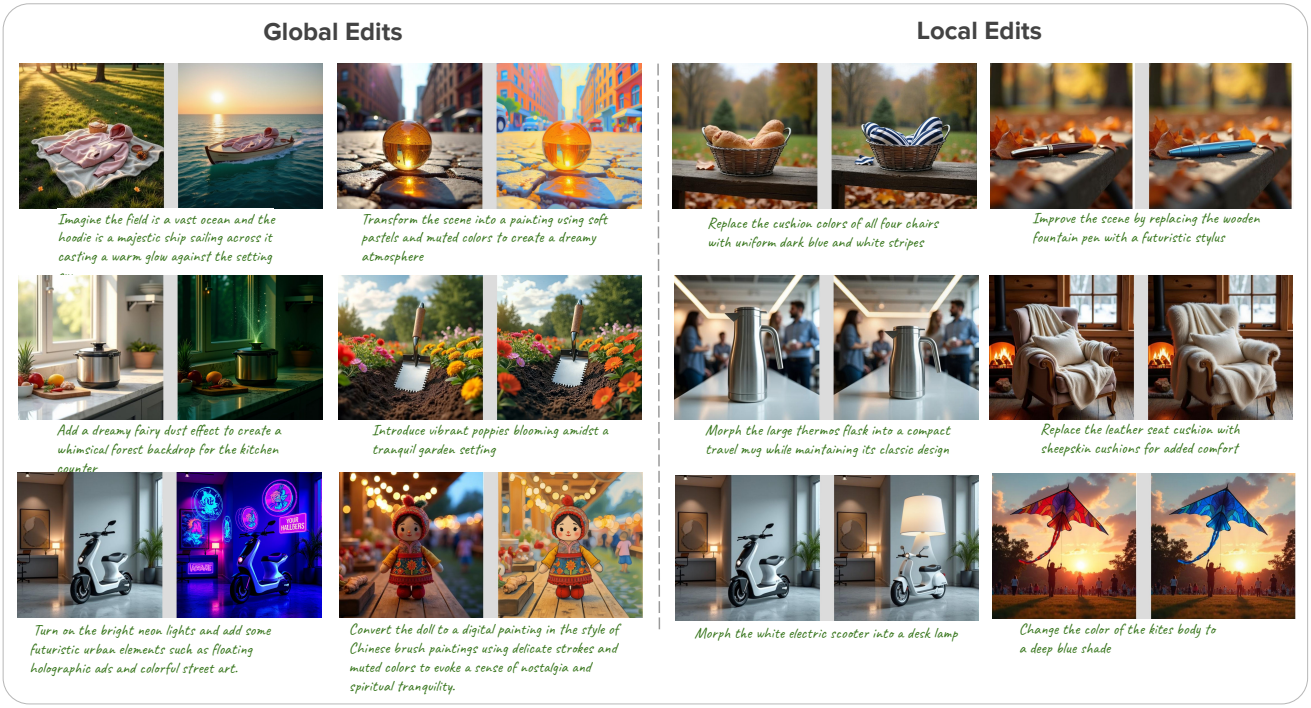

Figure 3. Samples for generated edit instructions and the generated edits from Flux Kontext

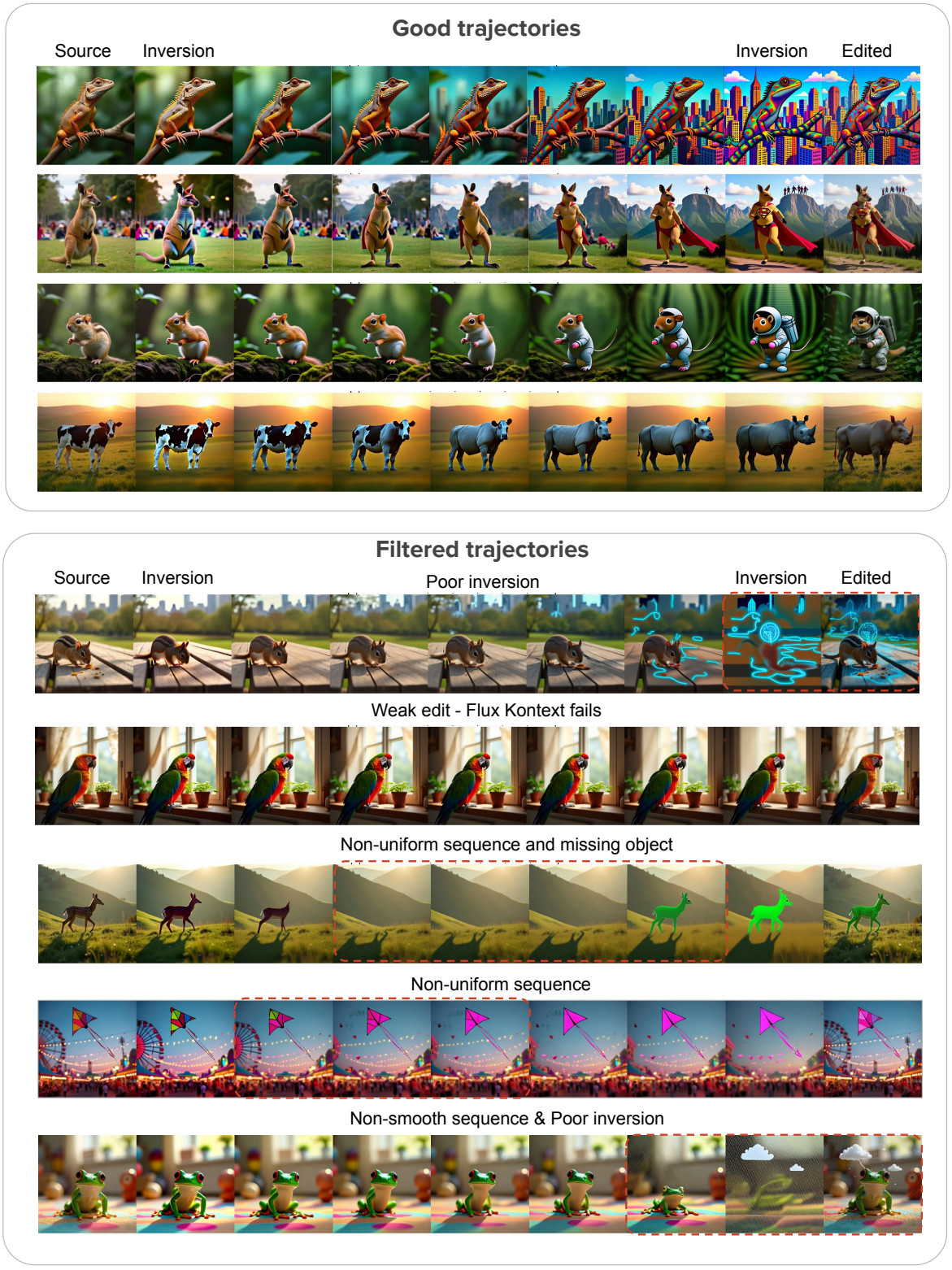

Figure 4. Samples trajectories from our synthesized dataset

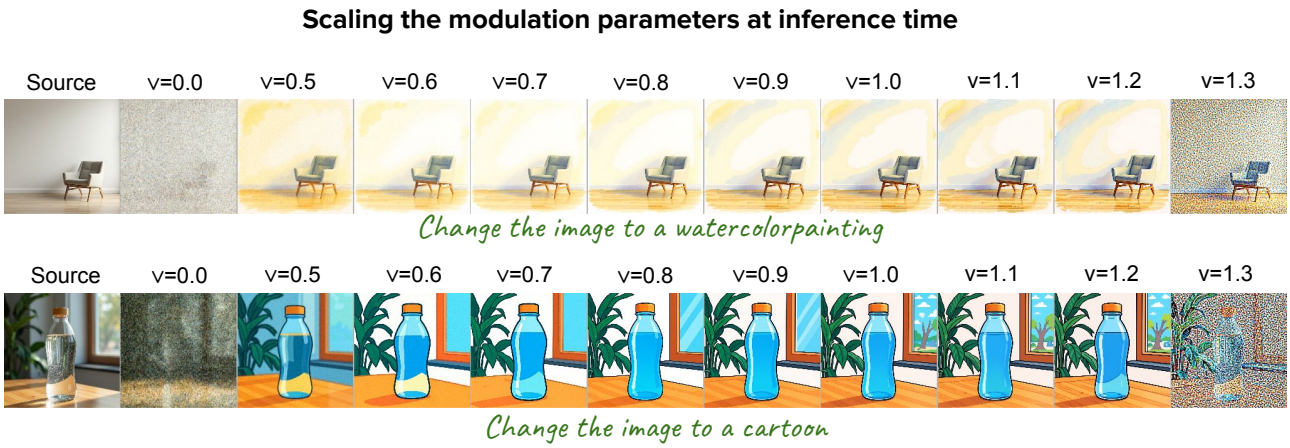

Figure 5. **Inference time control in modulation space.** We conducted a simple experiment by scaling the text modulation parameters with values of $v \in (0.5, 1.3)$ to generate multiple edits. While these edits varied across different scales, the variations did not consistently correlate with the intended edit strength. This highlights the need for a dedicated learning module that can translate such variations into user-interpretable strength control by accurately manipulating the modulation parameters.

A.4.2. Evaluation of identity preservation

We quantify the image identity preservation by computing the CLIP-Image similarity and DINO-Image similarity between the source image and the edited image across different edit strengths. We present a plot of the image similarity value across the edit strengths in Fig. 6. In an ideal case, the image similarity should decay linearly with the increase in edit strength s. Our method achieves the desired trend while still keeping the images with the strongest edits ($s = 1$) close to the source image. In comparison, the baselines either do not follow the linear trend or change the identity substantially when strong edits are performed (MARBLE, Concept Sliders).

A.4.3. Faithfulness and quality tradeoff.

We plot the Pareto front between faithfulness (CLIP-dir) and identity preservation (LPIPS(x_s,x_0)) across all edit strengths in Fig. 7. Our method exhibits monotonic and smooth transition for both metrics across the full range of s. In contrast, baselines quickly diverge (higher LPIPS) from the source even for small edit strengths ($0 < s < 0.50$) or are not monotonic. Notably, at intermediate strength (e.g., $s = 0.50$), we achieve significantly higher faithfulness with lesser identity change than baselines.

A.5. Ablation study

We present an ablation study in Fig. 8 for different architecture choices. Adding the output of the slider projector in the text embedding space leads to edit transitions with abrupt jumps. Similarly, without adding the pooled text embedding in the projector leads to non-smooth edit trajectories. Our design of injecting the slider control in the modulation space and making the projector adapt to the edit instruction embedding results in smooth trajectories, enabling fine-grained control to the user.

A.6. Evaluation Metrics

A.6.1. Smoothness of the edit sequence

We consider first and second-order smoothness of an edit trajectory for quantitative evaluation. For a given source image x and edit instruction, we generate a sequence of N edited images $\{y_{s_1}, y_{s_2}, \ldots, y_{s_N}\}$, and include the source image as the initial element $y_{s_0} = x$, yielding a sequence of $N{+}1$ images. We use an image distance metric $d(\cdot, \cdot)$ to compare the images and explore a semantic metric (Dreamsim [5]) and LPIPS.

First-order smoothness. We define adjacent distances between the images in the sequence as:

$$d_i = d(y_{s_i}, y_{s_{i+1}}), \quad i = 0, \ldots, N{-}1,$$

and compute the cumulative path length

$$L = \sum_{i=0}^{N-1} d_i.$$

The first-order smoothness is then computed as:

$$\delta^1 = \max_i \frac{d_i}{L},$$

which captures the largest normalized jump in the generated edit trajectory.

Second-order smoothness. For local consistency, we compute the triangle deficit given by

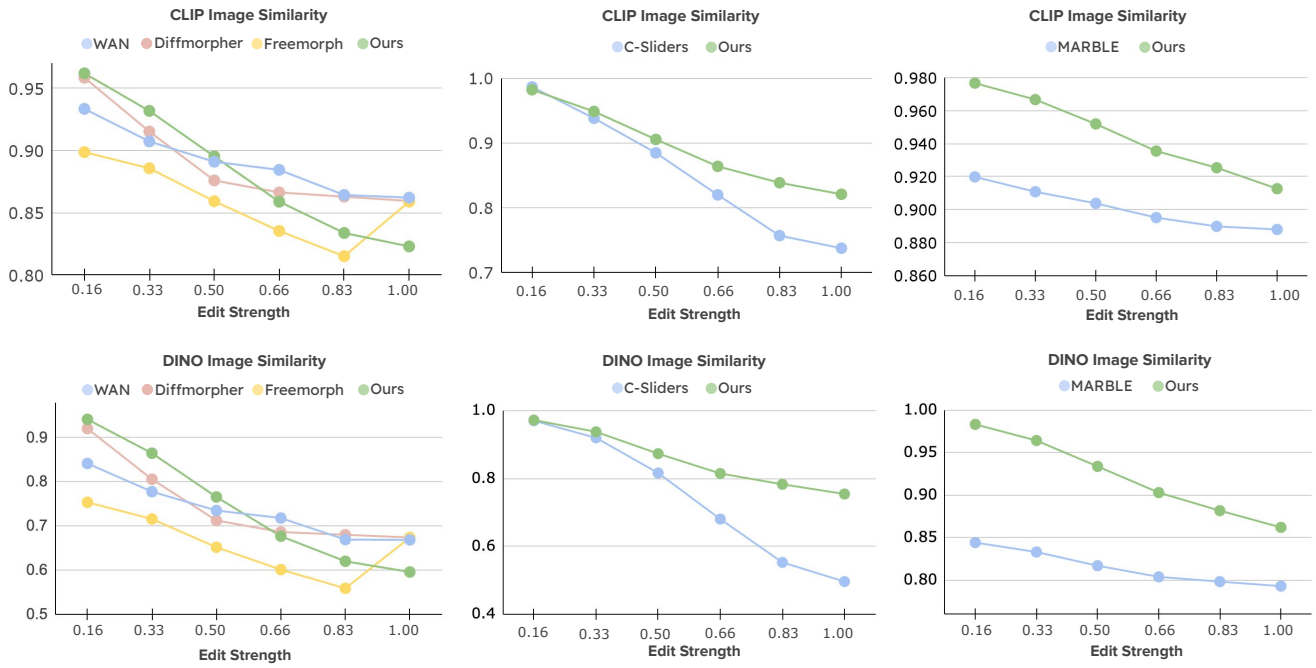

Figure 6. **Identity preservation comparison.** We evaluate the identity preservation of the source image during editing by computing image similarity with CLIP and Dino features. We plot the similarity scores across each edit strength to analyse the trend. Ideally, as we increase the edit strength, the image similarity should decay linearly. As compared to the baseline methods, our method achieves smooth decay trend for comparison against all the baselines. Notably, for MARBLE the image similarity is small even for the first edit, indicating its inferiority in identity preservation. For Concept-Sliders though the trend is good, it significantly changes the subject identity for higher edit strengths, resulting in poor image similarity. Our method achieves the desired trend while keeping the identity intact, even for the strongest edits, with a strength close to full edit.

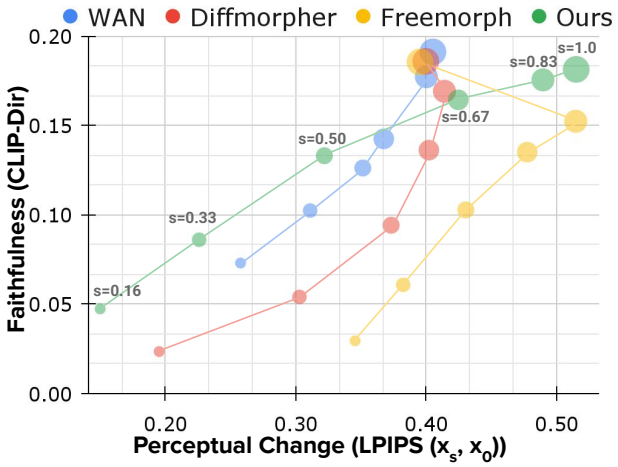

Figure 7. We plot the faithfulness of the edits along with the perceptual change in the source image. Our method achieves a monotonic and smooth transitions for both the metrics indicating superior tradeoff.

$$\Delta_i = d(y_{s_i}, y_{s_{i+1}}) + d(y_{s_{i+1}}, y_{s_{i+2}}) - d(y_{s_i}, y_{s_{i+2}})$$
$$i = 0, \ldots, N-2.$$

Each deficit is normalized by the direct distance between the endpoints:

$$\tilde{\Delta}_i = \frac{\Delta_i}{d(y_{s_i}, y_{s_{i+2}})}.$$

The second-order smoothness is then computed as:

$$\delta^2 = \max_i \tilde{\Delta}_i,$$

where smaller values indicate smoother local transitions.

Analysis. We conducted a user study to evaluate how well smoothness metrics align with human preferences. Participants were shown pairs of edit sequences and asked which of the two images have smoother transitions. The study included 20 volunteers and 40 sequence pairs. For each sequence, we computed first- and second-order smoothness using two distance functions: LPIPS [13] and DreamSim [5]. We then measured agreement between user choices and each of the four metric configurations (Fig. 10). Results show that δ^2 (DreamSim) aligns best with user preferences, as it captures fine-grained semantic changes reflected in slider adjustments. While first-order smoothness prevents abrupt jumps, second-order smoothness ensures con-

Figure 8. **Ablation over model architecture.** Injecting the slider projector in the text space results in an abrupt transition in the first example and is non-monotonic in the second example. Further, using a slider projector in the modulation space without the pooled text embedding input results in sudden transitions. Our projector takes the text embedding as input and makes appropriate adjustments to the modulation parameters for smooth transitions in the edit sequences.

sistency in the rate of change, producing natural and continuous transitions that match user expectations for image editing. Fig. 9 illustrates this: although Sequence 1 has better first-order smoothness (lower δ^1), Sequence 2 is semantically smoother, captured by a lower δ^2_{smooth}. From these findings, we define the smoothness metric as:

$$\delta_{smooth} = \delta^2(Dreamsim)$$

A.6.2. Instruction following with CLIP directional similarity

For a given input image x, and edit instruction e, we edit the image with uniformly sampled edit strengths $\{s_i = i/N | i = 1, ..., N\}$ to obtain the edited image sequence $\{y_i | i = 1, ..., N\}$. We compute the CLIP-direction similarity [6] for each of the edits at each strength as:

$$d_i = d_{clip-sim}(y_{s_i}, x, e), \quad i = 1, ..., N$$

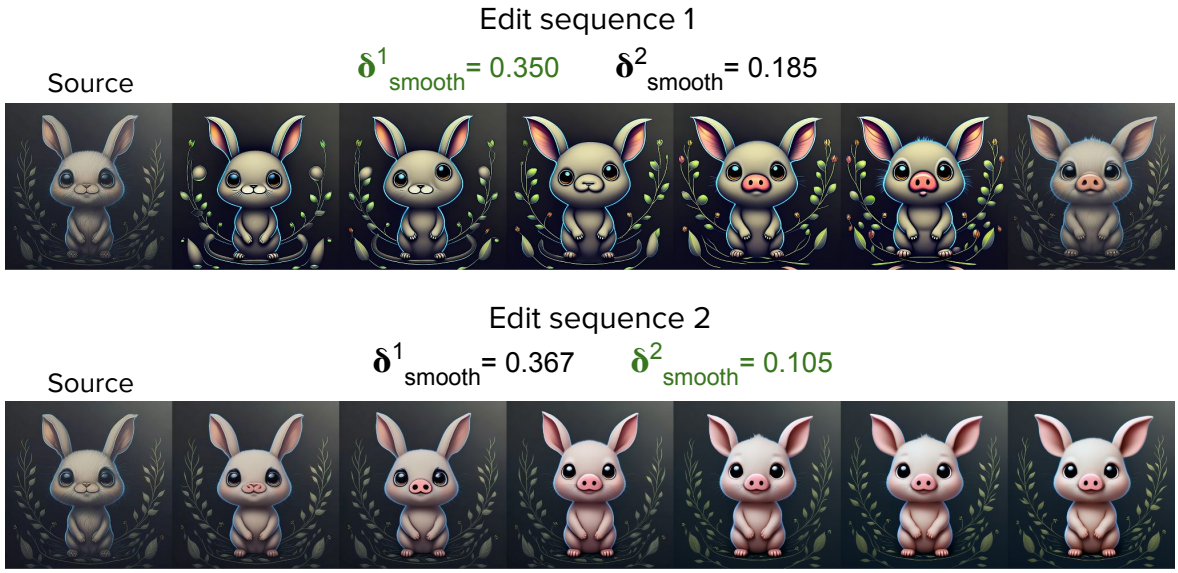

Figure 9. **Qualitative interpretation for first order and second order smoothness.** For slider-based image editing, second-order smoothness is more important than first-order smoothness, as it captures the local consistency needed for gradual, nuanced changes with slider.

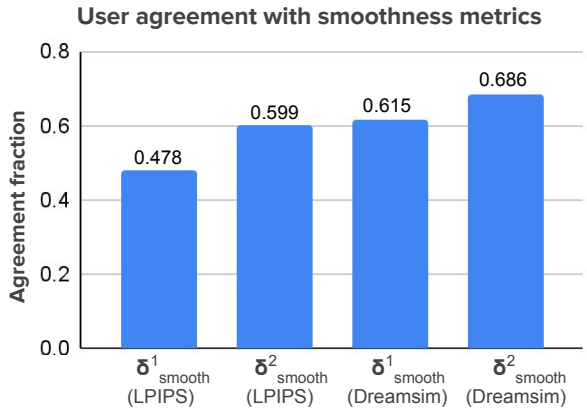

Figure 10. We perform a user study where we computed the alignment of the users' scores given for the smoothness of the sequence with the different variations of smoothness metrics. We found $\delta^{(2)}_{smooth}$ aligns well with the user preferences for smoothness, indicating that it is a good metric to measure the smoothness.

and report the aggregated normalized CLIP-sim as:

$$D_{clip-dir} = \frac{\sum_{i=0}^{N}(d_i/s_i)}{N}$$

adjusting the directional similarity based on the edit strength.

A.7. Qualitative Comparison

We present additional comparison with interpolation-based baselines in Fig. 11, 12 and with domain-specific methods ConceptSliders in Fig. 14, and MARBLE in Fig. 13.

A.8. Additional Baselines

We compared *Kontinuous Kontext* with two additional simple baselines: a) CFG-Scale - We change the classifier free guidance scale to control the extent of the edit, as we expect that with a higher CFG scale, the generated edit should follow the edit instruction more closely. b) Attention reweighting - We scale the cross-attention maps between the text tokens and the generated visual tokens, inspired by Prompt2Prompt [8]. The insight is that, if we increase the cross-attention weight between the target latents and the text instruction, the target image will pay more attention to the edit instruction, resulting in stronger edits. We present a comparison in Tab. 1 and Fig. 15. Both the inference time baselines fail the generate smooth edit transitions and distort the input image identity significantly. These abrupt transitions are also evident as very high scores in δ_{smooth} smoothness metric.

A.9. Failure case - Extrapolation beyond the training strength $s > 1$

One of the failure cases of our method is in extrapolating edits beyond the strength value $s = 1$. Our method either does not perform the edits for $s > 1$ or reduces the extent

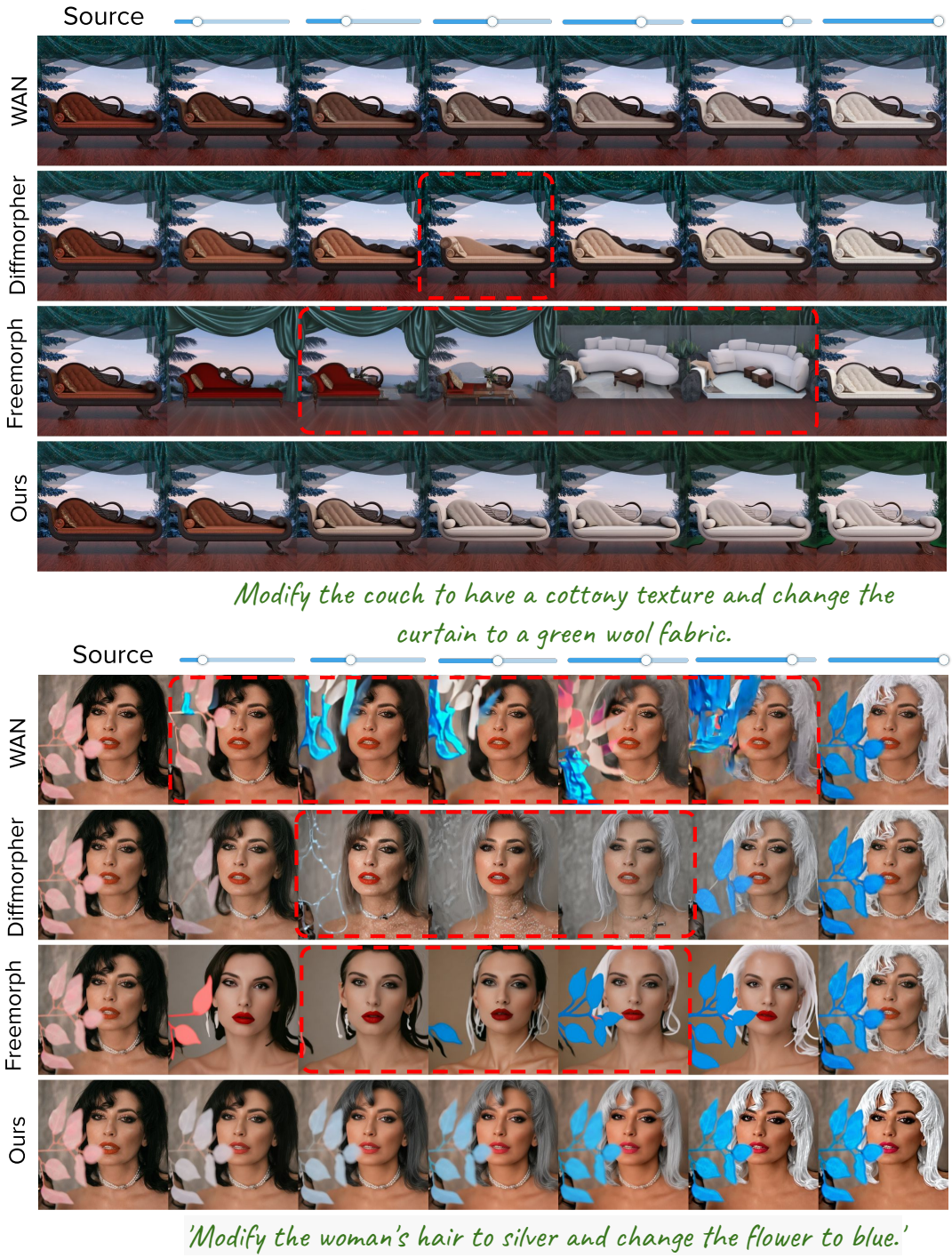

Figure 11. **Comparison with interpolation baselines.** Morphing-based methods generate smooth transitions; however, they often introduce artifacts in the intermediate images or omit details such as leaves. Similarly, the video inbetweening model WAN produces strong artifacts in intermediate frames, as these appearance transitions are out of the domain for an inbetweening model trained only on real videos.

of the edit as shown in Fig. 16.

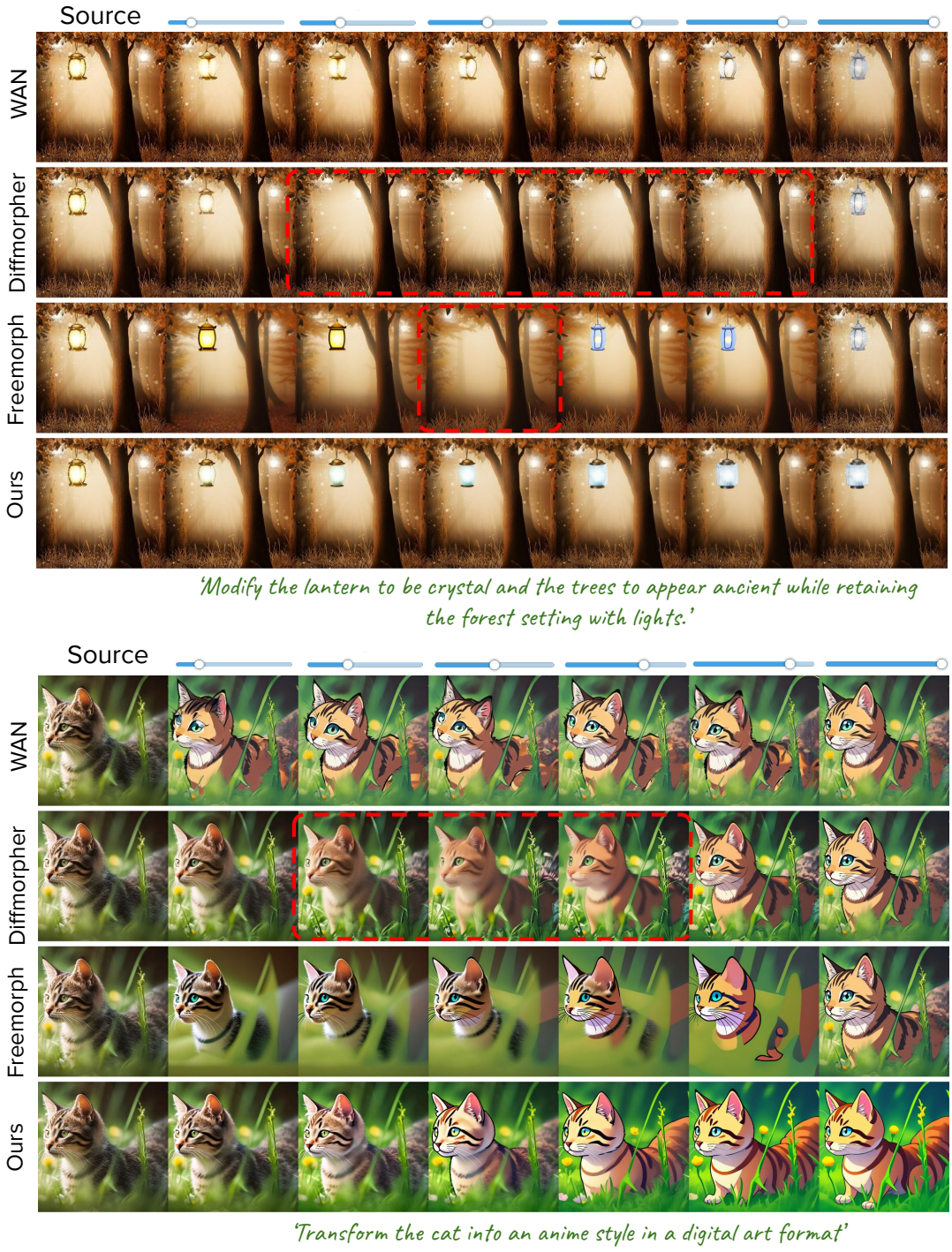

Figure 12. **Comparison with interpolation baselines.** DiffMorpher and FreeMorph remove objects in the intermediate edits of the first examples. DiffMorpher produces blurred outputs even for simple stylization transitions. The WAN inbetweening model generates transitions with abrupt jumps in both examples. In contrast, our method produces smooth transitions while preserving image identity.

A.10. Evaluation dataset

PIEBench: We evaluate our method on the widely used image editing benchmark PIEbench, which consists of diverse and challenging instruction-driven image editing test examples. The benchmark consists of edits from the following categories: change object, add/remove object, change pose,

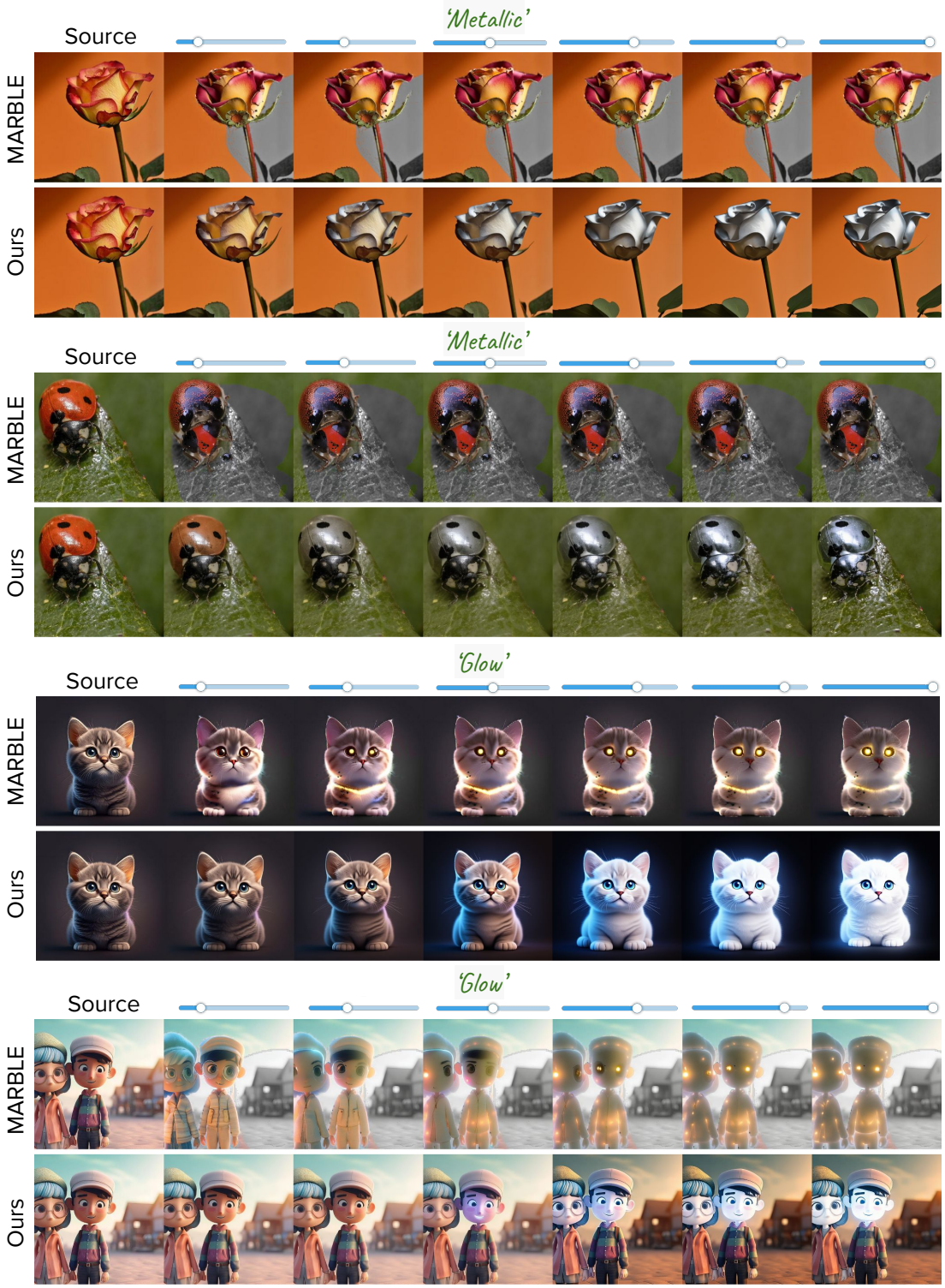

Figure 13. Comparison with MARBLE for material control

change color, change material, change background, and change style. We kept all edit categories except add/remove object, as it's a discrete edit, and our method focuses on continuous attribute editing.

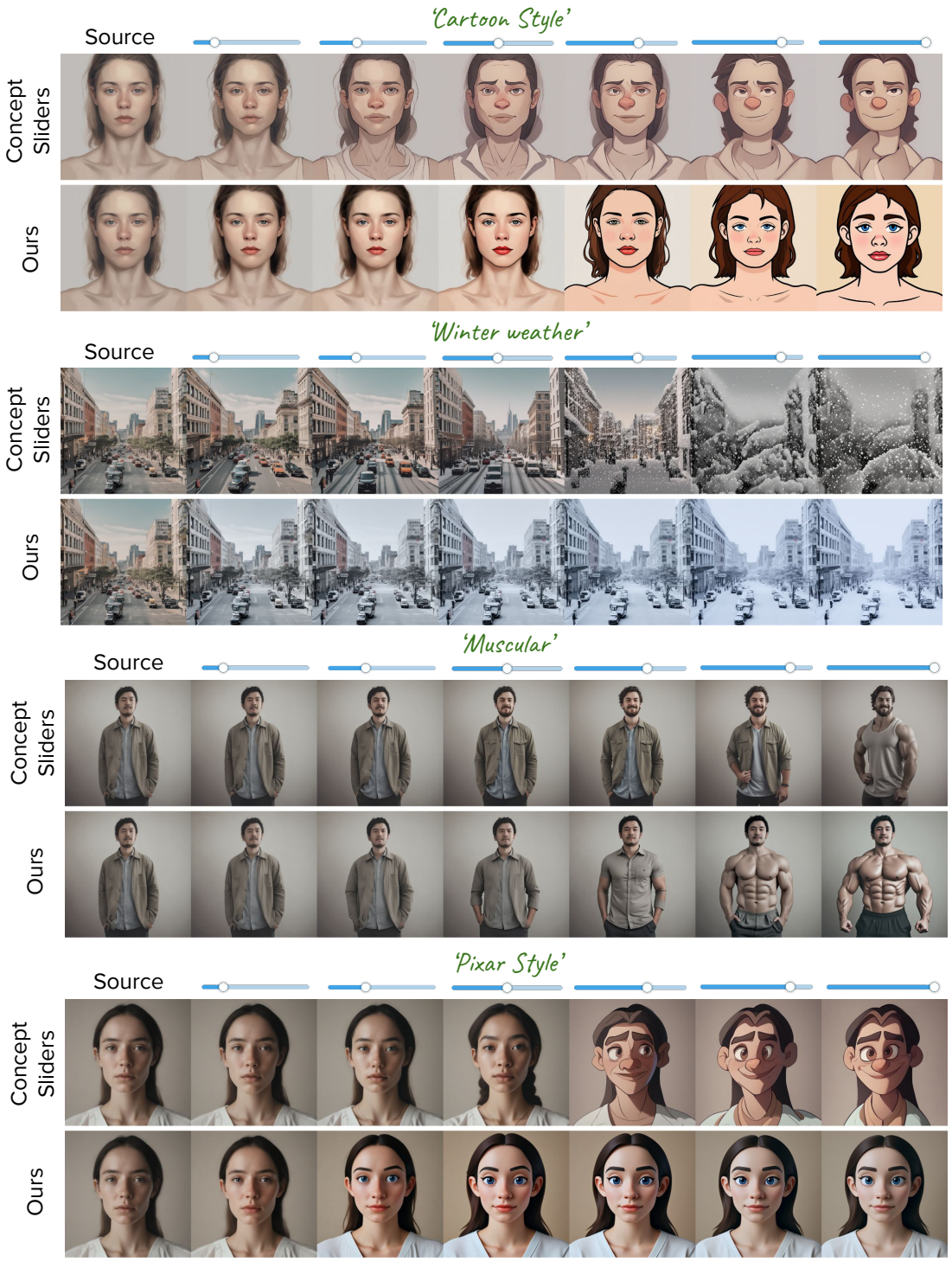

Figure 14. Comparison with Concept Sliders for diverse attribute editing.

Domain-specific comparison: As ConceptSliders optimize for specific attributes (e.g., smile, chubby), we accordingly select the dataset suitable for such edits (e.g., face images) for comparison. Further, as concept-sliders is designed to achieve continuous control during image generation, and is not directly suitable for image editing, we evaluate it on 44 generated images across 11 sliders covering facial attributes, stylization, and scene edits. For material

Figure 15. We compare with additional inference time baselines.

Methods	$\delta_{\text{smooth}} \downarrow$	CLIP-dir $\uparrow$
CFG-scale	152.205	**0.242**
Attention-weighing	120.760	0.237
Ours	**0.329**	0.241

Table 1. Comparison with additional inference time baselines.

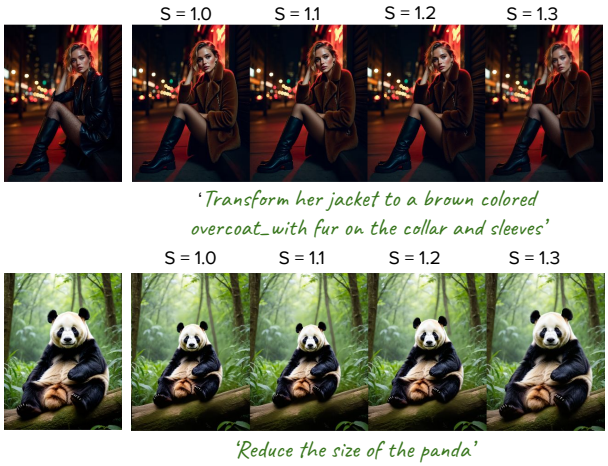

Figure 16. **Extrapolation of edit strengths**. One of the failure cases of our method is that it cannot generate edits with extrapolation well. In most cases, it either recreates the full edit image ($s = 1$), or reduces the extent of edit in the extrapolation region.

control, we evaluate MARBLE on 40 images from the material editing category from the PIEBench dataset.

A.11. Compositional Editing

Kontinuous Kontext enables multi-turn editing like the base Flux Kontext model. We first edit the input image with the first edit instruction and followed by editing with the second edit instruction. The results are shown in Fig. 17.

A.12. Generalization to other architectures

Our method builds on the findings [4, 7] that the modulation space of recent DiT based diffusion models enables fine-grained control over the scene contents. Leveraging this observation, we train a strength-conditioned slider projector to predict the offsets for the modulation parameters of DiT blocks. Though we have demonstrated this capability with Flux Kontext, most of the recent instruction-driven image editing models consist of the same DiT blocks. Hence, our approach is compatible with all the recent instruction-based editing (e.g., [12]) and can easily be integrated to achieve fine-grained strength control for image editing.

A.13. Latency and Computational Cost

Our method adds negligible inference overhead over the base Flux Kontext model in terms of latency ($60s$ vs $56s$) on a single NVIDIA $A6000$ GPU for 28 denoising steps. In comparison to the baselines, our method uses a comparable number of diffusion model inferences. As the baselines use different base diffusion models, we compare with them in terms of the number of diffusion evaluations in Table. 2. Our trainable parameters include a small 4 layer projector and LoRA parameters for the projection matrices of the base model, which accounts for $\approx 1\%$ of parameters of the base Flux Kontext model.

Method	# Diffusion Evaluations	Models used
Diffmorpher	N+1	Flux Kontext (1) + SD-2.1 (N)
Freemorph	N+1	Flux Kontext (1) + SD-2.1 (N)
Wan	2	Flux Kontext (1) + WAN-2.1 (1)
ConceptSliders	N	SD-XL (N)
MARBLE	N	SD-XL (N)
Ours	N	Flux Kontext (N)

Table 2. **Inference cost comparison.** We compare against baselines in terms of the number of diffusion model evaluations, as the baselines use different base diffusion models. Here, N represents the number of sequential edits generated for a given image and prompt. Our method uses the same or fewer inferences than all the baselines, except WAN, which requires a costly video model evaluation.

References

[1] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025. 1

[2] Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, Sumith Kulal, et al. Flux. 1 kontext: Flow matching for in-context image generation and editing in latent space. *arXiv e-prints*, pages arXiv–2506, 2025. 1

[3] Yukang Cao, Chenyang Si, Jinghao Wang, and Ziwei Liu. Freemorph: Tuning-free generalized image morphing with diffusion model. *arXiv preprint arXiv:2507.01953*, 2025. 1

[4] Yusuf Dalva, Kavana Venkatesh, and Pinar Yanardag. Fluxspace: Disentangled semantic editing in rectified flow transformers, 2024. 16

[5] Stephanie Fu, Netanel Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola. Dreamsim: Learning new dimensions of human visual similarity using synthetic data. *arXiv preprint arXiv:2306.09344*, 2023. 7, 8

[6] Rinon Gal, Or Patashnik, Haggai Maron, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. Stylegan-nada: Clip-guided domain adaptation of image generators. *ACM Transactions on Graphics (TOG)*, 41(4):1–13, 2022. 9

Figure 17. Compositional editing with Kontinuous Kontext. We perform composition of two edits with fine-grained strength control for individual edits. We first generate the edits for the first prompt and then apply the second edit on the edited images from the first pass. Our method can generate realistic compositional edits, providing finer control for editing multiple attributes together.

[7] Daniel Garibi, Shahar Yadin, Roni Paiss, Omer Tov, Shiran Zada, Ariel Ephrat, Tomer Michaeli, Inbar Mosseri, and Tali Dekel. Tokenverse: Versatile multi-concept personalization in token modulation space, 2025. 16

[8] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. 2022. 10

[9] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023. 1

[10] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 3

[11] Zhenxiong Tan, Songhua Liu, Xingyi Yang, Qiaochu Xue, and Xinchao Wang. Ominicontrol: Minimal and universal control for diffusion transformer. *arXiv preprint arXiv:2411.15098*, 2024. 1

[12] 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. 16

[13] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 8