Paint-Anything: Unified Any-Color Control for Image Generation and Editing

Ji Xie[1,2] **Dewei Zhou**[1,2] **Xinyu Huang**[1] **Xun Wang**[1]
[1]ByteDance Seed [2]Zhejiang University



Figure 1: **One hex-prompt interface for generation and editing.** A single finetuned model supports object recoloring and hex-conditioned generation through explicit color values in the prompt.

Abstract

Any-color control, specifying an object's color by any 24-bit hex value instead of a coarse color word, is a practical requirement for image generation and editing in design work. Prior work often relies on dedicated color representations or specialized inference procedures, which can be hard to scale across colors or transfer to new architectures. In this paper, we present *Paint-Anything*, a single model that reads hex values directly from the prompt for both generation and editing. Our key insight is that the text encoders of recent image generators are language models that already associate hex values with colors, so a generator can learn hex control by grounding these values in object pixels. To this end, we construct **Paint-500K**, which labels object colors in real images and adds editing pairs, and combine it with **pure-color anchors**, solid-color images applied only at high-noise timesteps, for lightweight RGB alignment. We further introduce **Any Color Benchmark (ACBench)**, with **ACBench-T2I** and **ACBench-Edit**, to measure object-level hex color fidelity. Experiments and ablations show that *Paint-Anything* substantially improves any-color generation and editing over the base model, suggesting that arbitrary hex-color control can be learned as a native prompt-following behavior of modern image models.

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

1 Introduction

How far is AI from a designer? Today's image models can imagine rich scenes and render them in remarkable detail (Podell et al., 2024; Esser et al., 2024; Wu et al., 2025; Deng et al., 2025; Zhou et al., 2026a, 2024a; Xie et al., 2026; Zhang et al., 2026; Fang et al., 2026; Fan et al., 2026; Chen et al., 2026; Li et al., 2026b; Zhou et al., 2026c, 2024b; Jin et al., 2025; Diao et al., 2026; Liao et al., 2025). Yet professional design (Ye et al., 2026b,a; Zhou et al., 2025; Du et al., 2025; Xu et al., 2026a; Li et al., 2026a; Liu et al., 2026) requires more than a convincing image: it requires following a color specification. A brand designer needs a logo in the brand's signature blue, and an online retailer needs product images in the specified red of a new collection. Neither requirement is captured by asking for "blue" or "red": both call for **explicit hex values**. Users should be able to choose an object's color as directly as a painter chooses colors for a canvas. We call this capability *any-color control*: users specify target colors through 24-bit hex values.

Prior work has explored various approaches to color control. Some methods introduce task-specific modules or learned color tokens, while others use training-free inference-time techniques such as sampling guidance or attention/value manipulation (Butt et al., 2024; Laria et al., 2026; Shukla et al., 2026; Lobashev et al., 2025; Shum et al., 2025; Qiu et al., 2025; Aharoni et al., 2026; Yin et al., 2026). These designs can be difficult to scale across colors, hard to transfer to new architectures, or costly at inference. As a result, any-color generation, colorization, and editing are often treated as separate problems rather than as one native prompt-following capability.

Fortunately, advances in large language models make hexadecimal color prompting a promising interface for this goal. A 24-bit RGB value can be written directly in text and bound to an object phrase, making it possible for a single model to support different any-color generation tasks with the same prompt language. As a simple motivating probe, Figure 2 shows that even a small LLM, Qwen3-4B (Yang et al., 2025a), can parse raw hex strings such as `#F0FFF1` and `#E8D00A` into plausible color descriptions. This suggests that hex strings can be used as a prompt-native numeric color interface.

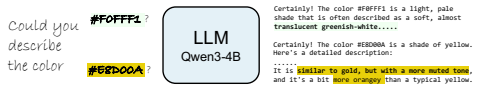


Figure 2: **Hex color strings are legible to modern language models.** Even a compact 4B language model such as Qwen3-4B can associate raw hex codes such as `#F0FFF1` and `#E8D00A` with plausible color semantics, suggesting that hex strings can serve as a unified text interface for any-color generation and editing.

In this work, we present *Paint-Anything*, a unified model for any-color controllable generation and editing. *Paint-Anything* keeps color specification entirely inside the text prompt using explicit spans such as `<color>#AABBCC</color>`, so that one interface covers both generation and editing. Instead of adding a separate inference-time color controller, we construct **Paint-500K** through a filtered data pipeline that turns real-world images into object-level hex-labeled supervision. VLM grounding identifies caption-relevant objects, segmentation masks isolate object pixels, and MeanShift clustering in CIELAB space (Commission Internationale de l'Eclairage, 2018) estimates dominant colors under natural illumination. Shadows make these real-image labels *approximate colors*. To supply a clean low-level color reference, we additionally train on pure-color images as a **"pure-color anchor"**: each sample pairs a hex code with a solid-color target, and these samples are activated only at high-noise timesteps. This anchor gives the model a clean mapping from numeric hex strings to RGB statistics while low-noise training uses natural images. Figure 3 summarizes the training pipeline, and Figure 4 shows how Paint-500K is built.

To better evaluate any-color generation and editing, we introduce **Any Color Benchmark (ACBench)**[1], an object-level color-fidelity benchmark with two components: **ACBench-T2I** and **ACBench-Edit**. It measures the color fidelity of generated or edited object regions to the requested hex values. On `FLUX.2-4B`, *Paint-Anything* improves ACBench color-fidelity scores by **85.3%** for generation and **28.3%** for editing relative to the base model, and ablations show that pure-color anchors are an important ingredient in learning reliable pixel-space color control. Beyond ACBench, the same finetuned model achieves the highest average CompColor score (Shomer-Chai et al., 2026) among the compared methods under both named-color and hex prompts. The gains are further supported by GenColorBench NCU evaluation and a human preference study (Appendices A and K).

[1]ACBench was called PCBench in the reviewed version of this paper.

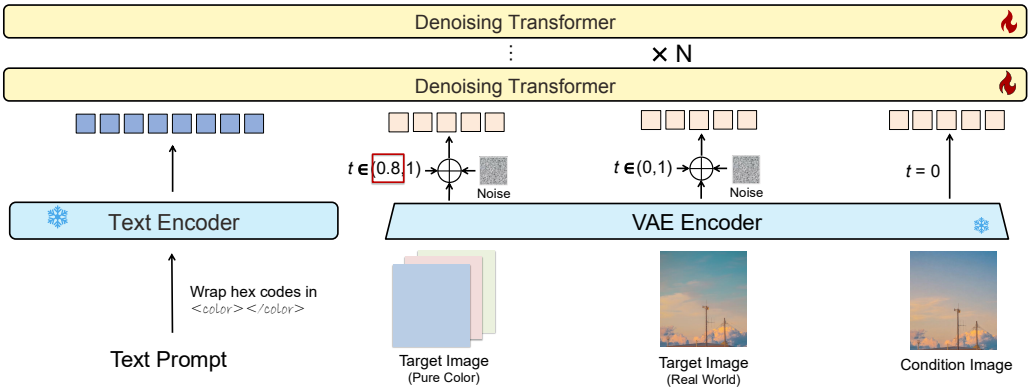


Figure 3: **Training architecture.** Hex colors are represented directly in the text prompt and encoded by the text encoder as text tokens. The VAE encodes target images, and the denoising transformer is trained with the flow-matching objective. Pure-color targets are used only at high-noise timesteps $t \in [t_{\text{gate}}, 1]$, real-world targets are sampled across $t \in (0, 1)$, and optional condition images for editing are encoded as clean image latents at $t = 0$.

Our contributions are:

- We discover that object-level hex supervision enables unified, prompt-native color control across generation and editing, and strengthen this capability with timestep-gated pure-color grounding.
- We develop a data pipeline that converts real images into object-level hex supervision for generation and editing. We use this pipeline to build Paint-500K.
- We introduce ACBench to evaluate object-level hex color fidelity in generation and editing, and demonstrate consistent gains across backbones and independent benchmarks.

2 Related Work

2.1 Color control in text-to-image generation

Recent work has begun to make color a first-class control signal in text-to-image generation. Some methods learn or modify color representations, such as learnable color prompts (Butt et al., 2024), CIELAB-aligned text embeddings (Tsai et al., 2025), or numeric-color encoders for RGB/hex strings (Butt et al., 2026b). Others use training-free inference-time techniques, such as sampling guidance, attention/value manipulation, or color-alignment objectives (Shukla et al., 2026; Shum et al., 2025; Lobashev et al., 2025; Agarwal et al., 2025; Laria et al., 2026; Qiu et al., 2025; Aharoni et al., 2026). These approaches improve color controllability, but often rely on additional control pathways, expensive inference-time optimization, or task-specific assumptions, which can limit unified task coverage.

NumColor (Butt et al., 2026b) learns numerical color embeddings, while BBQ-to-Image (Kachlon et al., 2026) uses structured prompts containing bounding boxes and RGB triplets.[2] Our model learns hex-to-color grounding through object-level supervision and uses hex codes directly in object descriptions and editing instructions. It unifies generation and editing without a dedicated color encoder or an intermediate layout representation containing bounding-box coordinates.

2.2 Color editing and image colorization

General image-editing systems based on prompt editing, instruction tuning, inversion, in-context editing, flow-based editing, or conditional control (Hertz et al., 2023; Brooks et al., 2023; Kawar et al., 2023; Zhang et al., 2025; Kulikov et al., 2025; Jiao et al., 2026; Zhang et al., 2023; Xu et al., 2023a,b; Zhou et al., 2026b; Xu et al., 2026b) can change visual attributes. Here, we focus on edits specified

[2]Public inference code and checkpoints needed to run these methods on ACBench were unavailable at evaluation time. We nevertheless compare with NumColor using its reported GenColorBench NCU results in Appendix A.

by numerical color values. More specialized color-editing methods improve language-guided, region-aware, or continuous object color control (Wang et al., 2023; Dong et al., 2024; Yin et al., 2024, 2026; Yang et al., 2025b, 2026). Palette- and histogram-based recoloring methods (Chang et al., 2015; Chao et al., 2023; Afifi et al., 2021) provide explicit controls over image colors, rather than binding hex specifications to objects through text prompts. Image-reference conditioning methods such as IP-Adapter (Ye et al., 2023) provide continuous visual cues, but global image conditioning can entangle target color with style or texture. In parallel, image colorization methods (Bahng et al., 2018; Chang et al., 2023b,a; Li et al., 2024; Liang et al., 2024; An et al., 2025; Cong et al., 2024) use masks, palette images, or semantic cues to constrain chromatic output. Our goal is to make the same hex-conditioned prompt interface work across generation and editing with a single finetuned model.

3 Method

3.1 Preliminaries

We build on a rectified-flow text-to-image model (Lipman et al., 2023; Esser et al., 2024) with a text encoder, a VAE (Kingma & Welling, 2014; Rombach et al., 2022), and a denoising transformer (Esser et al., 2024). Let y denote the text prompt and $I^\star$ the target image. For editing, I^c denotes the source image. The frozen text encoder maps y into embeddings $\mathbf{c}$. The VAE encoder maps the target image into a latent sequence $\mathbf{z}_0 = E_{\text{vae}}(I^\star)$, and for editing the same VAE encodes the optional source image into a clean condition latent $\mathbf{z}^c = E_{\text{vae}}(I^c)$.

Training follows the flow-matching objective. We sample $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ and $t \sim \mathcal{U}(0, 1)$, where $t = 0$ corresponds to the clean target latent and $t = 1$ corresponds to pure noise, and form

$$\mathbf{z}_t = (1 - t)\mathbf{z}_0 + t\epsilon, \qquad \mathbf{v}^\star = \epsilon - \mathbf{z}_0. \tag{1}$$

The denoiser predicts a velocity from the noisy target latent, the timestep, the text tokens, and optional image-condition tokens:

$$\widehat{\mathbf{v}}_\theta = D_\theta([\mathbf{z}_t; \mathbf{z}^c], t, \mathbf{c}), \tag{2}$$

where $[\mathbf{z}_t; \mathbf{z}^c]$ denotes sequence concatenation and reduces to $\mathbf{z}_t$ for T2I samples. The base loss is

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{\mathbf{z}_0, \epsilon, t, y}\left[\|\widehat{\mathbf{v}}_\theta - \mathbf{v}^\star\|_2^2\right]. \tag{3}$$

Thus generation and editing share the same denoising objective and differ only in whether the clean condition-image latent is concatenated to the noisy target latent.

3.2 Training pipeline

Figure 3 summarizes our training pipeline for grounding a shared hex-prompt interface through object-level supervision and timestep-gated pure-color anchors. The model follows ordinary text prompts for generation and takes a source image for editing.

Explicit color-tag wrapping. We mark each 24-bit sRGB hex specification as a color attribute by enclosing it in `<color>` and `</color>` tags directly in the prompt. Because the text encoder remains frozen during finetuning, we make the color span explicit in its input before encoding. For example, prompts can write "a photo of a `<color>` *#CFEFFB* `</color>` colored car" or "change the bag to `<color>` *#FFF8C4* `</color>`." Empirically, this wrapping improves object-level color fidelity and compositional color binding, with gains on ACBench-T2I, ACBench-Edit, and CompColor (Table 2).

Unified generation and editing format. We jointly train generation and editing so that one model learns color perception, object-color binding, and color-conditioned editing through the same hex-prompt interface. Every sample contains a target image and a hex-conditioned prompt, and each editing sample also contains a source image encoded as a clean condition-latent branch.

Pure-color anchor supervision. Real images provide object semantics, but their measured colors are noisy: shadows can map the same object to many plausible RGB values, leaving only an approximate color. We therefore add **pure-color anchors**: we sample 24-bit RGB values, render each value as a solid-color image, and pair it with a prompt containing the corresponding `<color>#HEX</color>` token. These anchors provide a clean low-level signal for hex-to-RGB grounding, so we use them to focus training on color-token alignment rather than on object semantics. Motivated by early

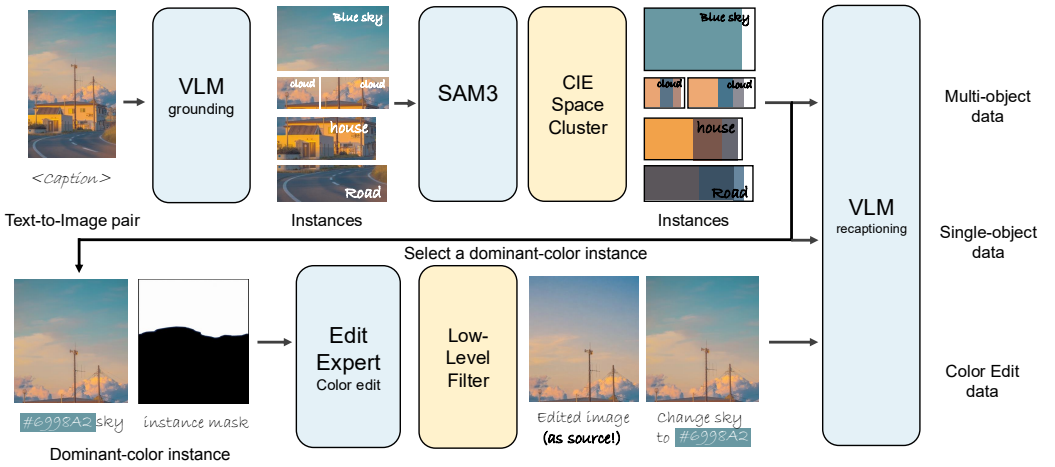


Figure 4: **Paint-500K data pipeline.** Real image-caption pairs are converted into hex-conditioned T2I and editing data through VLM grounding, SAM3 masks, CIE-space color extraction, edit synthesis, filtering, and recaptioning.

color stabilization in pure-color generation trajectories (Appendix E), we apply a **high-noise gate**: pure-color samples are trained only with $t \sim \mathcal{U}(t_{\text{gate}}, 1.0)$, while real-image T2I and editing samples use $t \sim \mathcal{U}(0, 1)$. For $t_{\text{gate}} = 1$, pure-color samples are trained at $t = 1$. Table 3 shows that high-noise gating improves generation and editing over ungated anchors.

Training loss. The final training objective combines the three streams:

$$\mathcal{L} = \lambda_{\text{t2i}}\mathcal{L}_{\text{t2i}} + \lambda_{\text{edit}}\mathcal{L}_{\text{edit}} + \lambda_{\text{rgb}}\mathcal{L}_{\text{rgb}}, \tag{4}$$

where each term uses the flow-matching loss in Eq. 3. $\mathcal{L}_{\text{t2i}}$ is computed on real-image color-generation samples, $\mathcal{L}_{\text{edit}}$ on paired image-editing samples, and $\mathcal{L}_{\text{rgb}}$ on pure-color anchor samples under the high-noise gate. We finetune the denoising transformer with the VAE and text encoder frozen. Additional mixture and optimization details are provided in Appendix D.

3.3 Data pipeline

Extracting object colors. Extracting a representative color from an object is crucial for reliable color supervision: lighting and shadows can make a single-colored object contain many pixel values. A common approach is RGB-space clustering (Yang et al., 2026; Pylette Developers, 2026; Kachlon et al., 2026), typically using k-means. This has two limitations: (I) RGB distances do not reflect human perception (Commission Internationale de l'Eclairage, 2018; Yang et al., 2026), potentially separating perceptually similar colors or merging distinct ones, and (II) objects have varying numbers of dominant colors, so a fixed K can split shading and noise into redundant labels. Inspired by ColorBind (Shomer-Chai et al., 2026), we use the perceptual CIELAB space (Commission Internationale de l'Eclairage, 2018). We further introduce MeanShift (Comaniciu & Meer, 2002) into object-color annotation to adapt the cluster count to each object's color distribution. Empirically, MeanShift reduces redundant color splits relative to fixed-K k-means in CIELAB, yielding a more coherent dominant-color region (Figure 5).

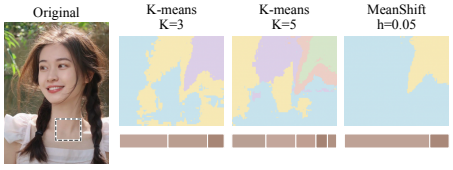


Figure 5: **MeanShift reduces redundant color splits compared with fixed-K k-means.** Both methods cluster the outlined skin patch in CIELAB before filtering. Maps show cluster membership, and bars show mean-RGB palettes weighted by pixel share.

Binding colors to objects and captions. Paint-500K uses high-quality real image-caption pairs from an internal collection. A VLM identifies caption-relevant objects and their bounding boxes, and SAM3 (Carion et al., 2026) produces the object masks used for color extraction. We filter clusters by pixel coverage and discard instances without sufficient retained coverage. The largest retained cluster provides an sRGB hex label for its object. The VLM then rewrites the caption to bind each label to the corresponding noun phrase using explicit `<color>#HEX</color>` spans. Figure 4



Figure 6: **Paint-500K training examples.** Top: T2I samples whose captions carry the target hex colors (shortened with ellipses). Bottom: editing pairs, each with the source (left), the target (right) and the instruction. Further examples are in Appendix M.

shows the full pipeline. Its filtering details and VLM templates are given in Appendices D and F.

Generation and editing streams. For T2I supervision, multi-object captions teach compositional color binding, while single-object crops emphasize direct object-color perception. We collect 400K T2I samples: 100K single-object and 300K multi-object examples. For editing, a pretrained editing model (Appendix D) recolors each single-object image to form the source, and the original photograph serves as the target, paired with an instruction to restore its hex color. Only the source is generated, so the targets carry no generative artifacts. Filtering for scene preservation, meaningful recoloring, and instruction consistency yields 100K editing samples that use the same prompt-native color syntax. Figure 6 shows examples from both streams.

4 Experiments

4.1 Benchmarks and metrics

CompColor benchmark. CompColor (Shomer-Chai et al., 2026) is a compositional color-binding benchmark that measures whether a generator can faithfully bind two distinct colors to two objects in the same prompt. Each prompt has the form "a {color} colored {object} and a {color} colored {object}", with color words drawn from a fixed *named-color* palette, e.g., "a *paleturquoise* colored shirt and a *ivory* colored bench". The released baseline table covers a wide range of color-binding methods evaluated under this named-color setting. Our task instead specifies prompt colors as 24-bit hex strings. We therefore evaluate CompColor under an additional hex-translated transfer protocol that replaces each color word with its canonical hex value while keeping the original object compositions intact, e.g., "a *#AFEEEE* colored shirt and a *#FFFFF0* colored bench". More details about this benchmark and our hex-translated protocol are in Appendix C.

Any Color Benchmark (ACBench). To evaluate object-level color fidelity under 24-bit hex prompts in both generation and editing, we introduce **ACBench**. *ACBench-T2I* contains 1000 generation prompts over common object categories, and *ACBench-Edit* contains 500 real-image recoloring prompts. Following prior object-based benchmarks (Ghosh et al., 2023), we pair randomly sampled hex colors with everyday objects, animals, and plants. ACBench covers three settings:

- **Single-object (Single)** (500 prompts): one object with one target color, e.g., "a photo of a *#CFEFFB* car."

- **Two-object (Two)** (500 prompts): two objects, each assigned a distinct target color, e.g., "a photo of a `#FFF8C4` dog and a `#CFEFFB` chair."
- **Edit** (500 prompts): one object with one target color, e.g., "change the bag to `#CFEFFB`."

Protocol and metric. We use SAM3 (Carion et al., 2026) to obtain the target masks. For T2I, we segment the prompted object in the generated image. For editing, we segment the target object in the source image, inspect the mask manually, and reuse the same source mask across the compared edited outputs. For a target mask M, let $\bar{\mathbf{c}} = |M|^{-1} \sum_{p \in M} I(p)$ be the mean sRGB vector on the 0 to 255 scale. Following prior work (Shomer-Chai et al., 2026; Butt et al., 2026a), we compare the estimated object color with the target. Given target RGB $\mathbf{c}^\star$, we compute $\mathrm{MAE} = \frac{1}{3} \sum_{k \in \{R,G,B\}} |\bar{c}_k - c_k^\star|$, and failed localization receives a score of 0. CIELAB estimators and CIEDE2000 evaluation appear in Appendices H and A.

We convert MAE into a normalized score using a piecewise linear function:

$$s = 100 \cdot \max\left(0,\ 1 - \frac{\max(0,\ \mathrm{MAE} - 16)}{48}\right). \tag{5}$$

The score measures region-level color fidelity while allowing natural appearance variation. MAE of at most 16, measured as the average absolute channel error, receives full score. MAE between 16 and 64 is linearly penalized, and MAE of at least 64 receives 0. The linear interval retains graded credit for imperfect color matches, preserving distinctions that a tighter cutoff would collapse to zero. We generate one image per T2I prompt with one seed per prompt and report scores separately for the equally sized Single and Two splits, with Overall as their arithmetic mean. Beyond these automatic scores, we also conduct a user study to examine whether people prefer our color results to those of the base model (Appendix K).

4.2 Experimental setup

Baseline and implementation details. We finetune `FLUX.2-klein-base-4B` (Black Forest Labs, 2026a) and `Z-Image` Base (Z-Image Team et al., 2025) with the same training recipe. Both models are trained for 4000 steps on 4 GPUs, using Adam with a global batch size of 72 and a learning rate of 2×10^{-5}. Throughout the paper, `FLUX.2-4B` and `FLUX.2-9B` denote the undistilled klein base checkpoints.[3] The FLUX ablations use the 4B backbone with the same optimization settings.

Evaluation details. Table 1 compares complete systems, whereas same-backbone ablations assess our training recipe. The open-source baselines are SD1.5 (Rombach et al., 2022), FLUX.1 (Black Forest Labs, 2024), Z-Image-Turbo (Z-Image Team et al., 2025), Qwen-Image and Qwen-Image-Edit (Wu et al., 2025), FLUX.2-4B (Black Forest Labs, 2026a), FLUX.2-9B (Black Forest Labs, 2026b) and FLUX.2-dev (Black Forest Labs, 2025). The color-specialized baselines are ColorBind/Edit (Shomer-Chai et al., 2026), CtrlColor (Liang et al., 2024), ColorPeel (Butt et al., 2024) and ColorWave (Laria et al., 2026). NumColor inference code and checkpoints were unavailable at evaluation time, so we use its published NCU results (Appendix A). CompColor includes named-color and hex prompts (Appendix C).

4.3 Qualitative results

Paint-Anything better matches the requested hex values across object categories and color families, whereas `FLUX.2-4B` often produces plausible but numerically inaccurate colors under the same prompts (Figure 7). Figure 8 shows generation for the same object under different hex specifications, while Figure 9 shows recoloring a target object in a source image according to a hex-color instruction.

4.4 Quantitative results

Table 1 shows that our finetuned `FLUX.2-4B` improves over its base checkpoint across three complementary aspects of color control: object-level hex color fidelity for text-to-image generation

[3]The "4B" and "9B" suffixes refer to the DiT parameter count, whereas the Total params. column in Table 1 reports the total parameter count, including the bundled Qwen3 text encoder.

Model	Total params.	ACBench-T2I ↑			ACBench-Edit ↑	CompColor ↑		
		Single	Two	Overall	Edit	Single	Close	Distant
SD1.5	1.0B	15.00	8.00	11.50	—	0.53†	0.36†	0.30†
FLUX.1	17B	25.00	21.00	23.00	—	0.56†	0.54†	0.49†
Z-Image-Turbo	10B	32.11	35.00	33.56	—	0.65	0.69	0.66
Qwen-Image / Edit	27B	26.07	24.34	25.21	54.40	0.59	0.59	0.63
FLUX.2-9B	17B	36.24	40.07	38.15	64.53	0.75	0.67	0.68
FLUX.2-dev	56B	51.32	52.08	51.70	68.87	0.73	0.72	**0.79**
FLUX.2-4B	8B	37.45	36.60	37.02	58.90	0.74	0.70	0.73
FLUX.2-4B (Hex)						0.34	0.41	0.38
Z-Image Base	10B	32.15	34.74	33.45	—	—	—	—
Color-specialized methods								
ColorBind/Edit	var.	30.42	34.17	32.30	60.38	0.72	0.71	0.73
CtrlColor	SD1.5-based	—	—	—	57.46	—	—	—
ColorPeel	SD1.4-based	45.28	34.63	39.96	—	0.68	0.62	0.64
ColorWave*	SDXL-based	50.36	42.71	46.54	—	0.72	0.68	0.70
FLUX.2-4B + **Ours**	8B	**72.67**	**64.49**	**68.58**	**75.57**	**0.81**	**0.80**	0.76
FLUX.2-4B + **Ours** (Hex)						0.78	**0.80**	**0.79**
Z-Image Base + **Ours**	10B	56.76	50.78	53.77	—	0.77	0.77	0.76

Table 1: Main results: ACBench (0–100) and CompColor (0–1). CompColor uses named colors unless marked *(Hex)*, and paired rows share ACBench results. —: unreported. †: quoted from ColorBind (Shomer-Chai et al., 2026). *: our reproduction. Reruns use official defaults (FLUX.2-4B Edit CFG 2.0). NCU and additional CompColor results: Appendices A and C.

on ACBench-T2I, recoloring fidelity for image editing on ACBench-Edit, and compositional color binding on CompColor.

Hex supervision enables an 8B model to outperform a 56B model. Within the FLUX.2 family, ACBench-T2I Overall and ACBench-Edit scores increase across the off-the-shelf models with 8B, 17B, and 56B total parameters. Yet our finetuned 8B model exceeds the 56B model by 16.88 points on ACBench-T2I and 6.70 points on ACBench-Edit. The same training recipe also produces a large gain on Z-Image Base.

Finetuning closes the word-to-hex gap. On CompColor, replacing color names with raw hex strings reduces the FLUX.2-4B base model's average score from 0.72 to 0.38. After localized hex supervision, the hex-prompt average more than doubles and exceeds the base model's named-color



Figure 7: **Qualitative comparison with the base model.** Each pair shows FLUX.2-4B (left) and *Paint-Anything* (right) on the same hex-conditioned prompt. The baseline often produces colors that are semantically plausible but numerically distant from the requested hex value.

average. The named-color average also improves from 0.72 to 0.79, showing that finetuning preserves and improves the model's existing compositional color-binding ability.

***Paint-Anything* outperforms the evaluated specialized systems.** On ACBench, *Paint-Anything* exceeds the strongest specialized baseline in each task: ColorWave by 22.04 points on ACBench-T2I and ColorBind/Edit by 15.19 points on ACBench-Edit. The independent GenColorBench NCU evaluation also ranks *Paint-Anything* highest among the compared methods, while a human preference study favors its color results over those of the base model (Appendices A and K).

4.5 Ablation study

We ablate four choices in the final recipe: color-token wrapping, pure-color anchors, high-noise gating, and full-model finetuning. Table 2 shows that the complete recipe is best across ACBench generation, ACBench editing, and CompColor, while Table 3 isolates the gate threshold.

Full finetuning and LoRA. With the complete recipe, a shared learning rate and 4000 steps, full finetuning exceeds rank-256 LoRA: 68.58 vs. 48.45 (T2I), 75.57 vs. 58.71 (Edit), and 0.79 vs. 0.54 (CompColor).

LoRA	Wrap	Pure Img	Gate	ACB-T2I ↑	ACB-Edit ↑	CompC. ↑
×	×	×	×	44.06	65.81	0.52
×	✓	✓	×	63.83	70.39	0.71
×	✓	×	×	57.16	73.89	0.71
×	×	✓	✓	48.01	64.08	0.59
✓	✓	✓	✓	48.45	58.71	0.54
×	✓	✓	✓	**68.58**	**75.57**	**0.79**

Table 2: Module ablations on `FLUX.2-4B`. LoRA × denotes full finetuning and ✓ denotes rank-256 LoRA. Pure Img denotes pure-color anchors. ACB-T2I and ACB-Edit are the ACBench scores, and CompC. is the hex-prompt CompColor average.

Wrapping clarifies hex syntax, while anchors provide a clean color reference. Without pure-color anchors, bare-hex finetuning reaches only 44.06 and 65.81 on ACBench-T2I and ACBench-Edit. Color-token wrapping raises these scores to 57.16 and 73.89, giving the model a clearer textual handle for raw hex strings. With wrapping enabled, adding pure-color anchors provides a clean hex-to-RGB reference. Without gating, however, generation improves to 63.83 while editing decreases to 70.39. The ungated anchor objective therefore helps generation at the cost of editing fidelity.

Metric	0.0	0.7	0.8	0.9	1.0
ACBench-T2I ↑	63.83	65.93	**68.58**	67.54	67.62
ACBench-Edit ↑	70.39	72.68	**75.57**	73.68	71.49
CompColor ↑	0.71	**0.79**	**0.79**	0.77	0.76

Table 3: Timestep-gate ablation on `FLUX.2-4B`, with wrapping and anchors enabled. $t_{\text{gate}} = 0$ matches the ungated anchor row in Table 2.

High-noise gating makes anchor supervision more effective. Restricting pure-color supervision to high-noise timesteps matches where color emerges in pure-color trajectories (Appendix E) and leaves lower-noise training to real-image generation and editing. With wrapping enabled, our gated recipe improves ACBench-T2I by 4.75 points and ACBench-Edit by 5.18 points over ungated anchors, while also improving CompColor. It also outperforms training without anchors on both generation and editing. Thresholds 0.7, 0.8, and 0.9 all improve both tasks over ungated anchors (Table 3), showing low sensitivity. We use $t_{\text{gate}} = 0.8$.

5 Limitations and Future Work

Our current training data do not include palette-specific supervision. Future work can add palette-specific supervision and extend the same unified interface to a broader range of color-control tasks.

6 Conclusion

We presented *Paint-Anything*, showing that pretrained image models can learn any-color control as a unified prompt-native capability for generation and editing. Paint-500K supplies object-level hex supervision through perceptual color clustering, with our empirical analysis motivating MeanShift over fixed-K clustering. Overall, *Paint-Anything* improves ACBench color fidelity by 85.3% for generation and 28.3% for editing relative to the base model.



Figure 8: **Hex-conditioned generation.** *Paint-Anything* follows a range of requested hex colors for the same object. All results use the same random seed.

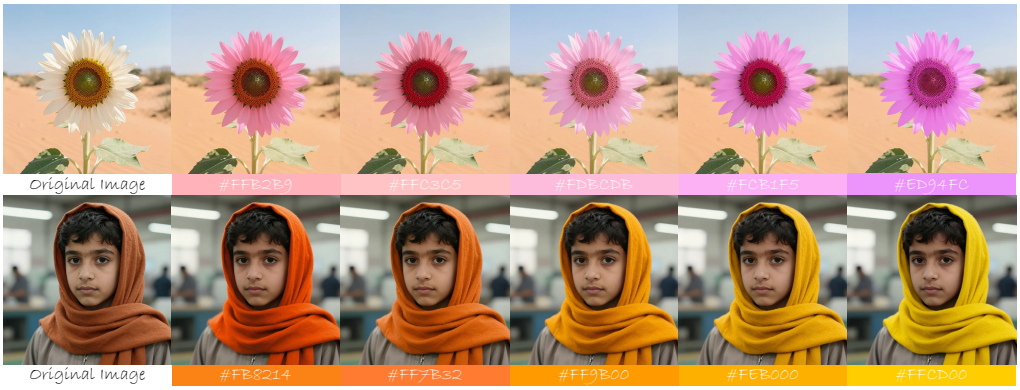


Figure 9: **Hex-conditioned editing.** Given a source image and a hex-color instruction, *Paint-Anything* follows the requested hex color when recoloring the target object.

References

Mahmoud Afifi, Marcus A. Brubaker, and Michael S. Brown. HistoGAN: Controlling colors of gan-generated and real images via color histograms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7941–7950, 2021. URL https://openaccess.thecvf.com/content/CVPR2021/html/Afifi_HistoGAN_Controlling_Colors_of_GAN-Generated_and_Real_Images_via_Color_CVPR_2021_paper.html.

Aishwarya Agarwal, Srikrishna Karanam, and Balaji Vasan Srinivasan. Training-free color-style disentanglement for constrained text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 6291–6300, 2025. URL https://openaccess.thecvf.com/content/CVPR2025W/CVEU/html/Agarwal_Training-free_Color-Style_Disentanglement_for_Constrained_Text-to-Image_Synthesis_CVPRW_2025_paper.html.

Elad Aharoni, Noy Porat, Dani Lischinski, and Ariel Shamir. Palette aligned image diffusion. *Computer Graphics Forum*, pp. e70384, 2026. doi: 10.1111/cgf.70384. URL https://doi.org/10.1111/cgf.70384.

Yanru An, Ling Gui, Qiang Hu, Chunlei Cai, Tianxiao Ye, Xiaoyun Zhang, and Yanfeng Wang. Controllable image colorization with instance-aware texts and masks, 2025. URL https://arxiv.org/abs/2505.08705.

Hyojin Bahng, Seungjoo Yoo, Wonwoong Cho, David K. Park, Ziming Wu, Xiaojuan Ma, and Jaegul Choo. Coloring with words: Guiding image colorization through text-based palette generation. In *Proceedings of the European Conference on Computer Vision*, 2018.

Black Forest Labs. FLUX. https://github.com/black-forest-labs/flux, 2024.

Black Forest Labs. FLUX.2 [dev] model card. https://huggingface.co/black-forest-labs/FLUX.2-dev, 2025. Accessed: 2026-04-16.

Black Forest Labs. FLUX.2 [klein] Base 4B model card. https://huggingface.co/black-forest-labs/FLUX.2-klein-base-4B, 2026a. Accessed: 2026-04-16.

Black Forest Labs. FLUX.2 [klein] Base 9B model card. https://huggingface.co/black-forest-labs/FLUX.2-klein-base-9B, 2026b. Accessed: 2026-04-16.

Tim Brooks, Aleksander Holynski, and Alexei A. Efros. InstructPix2Pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18392–18402, 2023.

Muhammad Atif Butt, Kai Wang, Javier Vazquez-Corral, and Joost van de Weijer. ColorPeel: Color prompt learning with diffusion models via color and shape disentanglement. In *Proceedings of the European Conference on Computer Vision*, 2024.

Muhammad Atif Butt, Alexandra Gomez-Villa, Tao Wu, Javier Vazquez-Corral, Joost van de Weijer, and Kai Wang. GenColorBench: A color evaluation benchmark for text-to-image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 36638–36648, 2026a. URL https://openaccess.thecvf.com/CVPR2026?day=2026-06-07.

Muhammad Atif Butt, Diego Hernández, Alexandra Gomez-Villa, Kai Wang, Javier Vazquez-Corral, and Joost Van De Weijer. NumColor: Precise numeric color control in text-to-image generation. In *European Conference on Computer Vision*, 2026b. URL https://arxiv.org/abs/2603.13547. Accepted for publication.

ByteDance Seed Team. Seed1.8 model card. https://github.com/ByteDance-Seed/Seed-1.8, 2026. Accessed: 2026-05-04.

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. SAM 3: Segment anything with concepts. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/e0982cbc81401df3430ee1ff780dc7a2-Abstract-Conference.html.

Huiwen Chang, Ohad Fried, Yiming Liu, Stephen DiVerdi, and Adam Finkelstein. Palette-based photo recoloring. *ACM Transactions on Graphics*, 34(4):139:1–139:11, 2015. doi: 10.1145/2766978. URL https://doi.org/10.1145/2766978.

Zheng Chang, Shuchen Weng, Peixuan Zhang, Yu Li, Si Li, and Boxin Shi. L-CAD: Language-based colorization with any-level descriptions using diffusion priors. In *Advances in Neural Information Processing Systems*, volume 36, pp. 77174–77186, 2023a. URL https://arxiv.org/abs/2305.15217.

Zheng Chang, Shuchen Weng, Peixuan Zhang, Yu Li, Si Li, and Boxin Shi. L-CoIns: Language-based colorization with instance awareness. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19221–19230, 2023b. doi: 10.1109/CVPR52729.2023.01842. URL https://doi.org/10.1109/CVPR52729.2023.01842.

Cheng-Kang Ted Chao, Jason Klein, Jianchao Tan, Jose Echevarria, and Yotam Gingold. ColorfulCurves: Palette-aware lightness control and color editing via sparse optimization. *ACM Transactions on Graphics (TOG)*, 42(4):1–12, 2023.

Hila Chefer, Yuval Alaluf, Yael Vinker, Lior Wolf, and Daniel Cohen-Or. Attend-and-excite: Attention-based semantic guidance for text-to-image diffusion models. *ACM Transactions on Graphics*, 2023. doi: 10.1145/3592116. URL https://doi.org/10.1145/3592116.

Zhennan Chen, Junwei Zhu, Xu Chen, Jiangning Zhang, Jiawei Chen, Zhuoqi Zeng, Wei Zhang, Chengjie Wang, Jian Yang, and Ying Tai. L2p: Unlocking latent potential for pixel generation. *arXiv preprint arXiv:2605.12013*, 2026.

Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002. doi: 10.1109/34.1000236. URL https://doi.org/10.1109/34.1000236.

Commission Internationale de l'Eclairage. Colorimetry, 4th edition. Technical Report CIE 015:2018, CIE, 2018. URL https://www.cie.co.at/publications/colorimetry-4th-edition.

Xiaoyan Cong, Yue Wu, Qifeng Chen, and Chenyang Lei. Automatic controllable colorization via imagination. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2609–2619, 2024.

Omer Dahary, Or Patashnik, Kfir Aberman, and Daniel Cohen-Or. Be yourself: Bounded attention for multi-subject text-to-image generation. In *European Conference on Computer Vision*, 2024. URL https://www.ecva.net/papers/eccv_2024/papers_ECCV/papers/02245.pdf.

Chaorui Deng, Deyao Zhu, Kunchang Li, Chenhui Gou, Feng Li, Zeyu Wang, Shu Zhong, Weihao Yu, Xiaonan Nie, Ziang Song, et al. Emerging properties in unified multimodal pretraining. *arXiv preprint arXiv:2505.14683*, 2025. URL https://arxiv.org/abs/2505.14683.

Haiwen Diao, Penghao Wu, Hanming Deng, Jiahao Wang, Shihao Bai, Silei Wu, Weichen Fan, Wenjie Ye, Wenwen Tong, Xiangyu Fan, et al. SenseNova-U1: Unifying multimodal understanding and generation with NEO-unify architecture. *arXiv preprint arXiv:2605.12500*, 2026.

Yi Dong, Yuxi Wang, Ruoxi Fan, Wenqi Ouyang, Zhiqi Shen, Peiran Ren, and Xuansong Xie. ChromaFusionNet (CFNet): Natural fusion of fine-grained color editing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 1591–1599, 2024. doi: 10.1609/AAAI.V38I2.27925. URL https://doi.org/10.1609/AAAI.V38I2.27925.

Nikai Du, Zhennan Chen, Zhizhou Chen, Shan Gao, Xi Chen, Zhengkai Jiang, Jian Yang, and Ying Tai. Textcrafter: Accurately rendering multiple texts in complex visual scenes. *arXiv preprint arXiv:2503.23461*, 2025.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *Proceedings of the International Conference on Machine Learning*, 2024. URL https://arxiv.org/abs/2403.03206.

Yijia Fan, Ziqi Huang, Zhongang Cai, Yan Li, Zimo Wen, Wanqi Yin, Haiwen Diao, and Ziwei Liu. Learning native reflection in unified models with interleaved reinforcement learning. *arXiv preprint arXiv:2609.35767*, 2026.

Zhen Fang, Ruiyan Han, XinYu Sun, Yuchen Ma, Ziheng Wang, Yu Zeng, Zehui Chen, Lin Chen, Wenxuan Huang, Wei-Jie Xu, et al. Unicorn: Towards self-improving unified multimodal models through self-generated supervision. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13638–13669, 2026.

Weixi Feng, Xuehai He, Tsu-Jui Fu, Varun Jampani, Arjun Akula, Pradyumna Narayana, Sugato Basu, Xin Eric Wang, and William Yang Wang. Training-free structured diffusion guidance for compositional text-to-image synthesis. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PUIqjT4rzq7.

Songwei Ge, Taesung Park, Jun-Yan Zhu, and Jia-Bin Huang. Expressive text-to-image generation with rich text. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7545–7556, 2023.

Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. GenEval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems*, 36:52132–52152, 2023.

Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross-attention control. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2208.01626.

Kaiyi Huang, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. T2I-CompBench: A comprehensive benchmark for open-world compositional text-to-image generation. In *Advances in Neural Information Processing Systems*, 2023.

Guanlong Jiao, Biqing Huang, Kuan-Chieh Wang, and Renjie Liao. UniEdit-Flow: Unleashing inversion and editing in the era of flow models. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/023d94f44110b9a3c62329beec739772-Abstract-Conference.html.

Weiyang Jin, Yuwei Niu, Jiaqi Liao, Chengqi Duan, Aoxue Li, Shenghua Gao, and Xihui Liu. SRUM: Fine-grained self-rewarding for unified multimodal models. *arXiv preprint arXiv:2510.12784*, 2025.

Eliran Kachlon, Alexander Visheratin, Nimrod Sarid, Tal Hacham, Eyal Gutflaish, Saar Huberman, Hezi Zisman, David Ruppin, and Ron Mokady. BBQ-to-Image: Numeric bounding box and qolor control in large-scale text-to-image models, 2026. URL https://arxiv.org/abs/2602.20672.

Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. Imagic: Text-based real image editing with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6007–6017, 2023. URL https://openaccess.thecvf.com/content/CVPR2023/html/Kawar_Imagic_Text-Based_Real_Image_Editing_With_Diffusion_Models_CVPR_2023_paper.html.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014. URL https://arxiv.org/abs/1312.6114.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4015–4026, 2023.

Vladimir Kulikov, Matan Kleiner, Inbar Huberman-Spiegelglas, and Tomer Michaeli. FlowEdit: Inversion-free text-based editing using pre-trained flow models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 19721–19730, October 2025.

Héctor Laria, Alexandra Gomez-Villa, Jiang Qin, Muhammad Atif Butt, Bogdan Raducanu, Javier Vazquez-Corral, Joost van de Weijer, and Kai Wang. Leveraging semantic attribute binding for free-lunch color control in diffusion models. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 7689–7698, 2026. doi: 10.1109/WACV61042.2026.00742. URL https://doi.org/10.1109/WACV61042.2026.00742.

Haodong Li, Chunmei Qing, Huanyu Zhang, Dongzhi Jiang, Yihang Zou, Hongbo Peng, Dingming Li, Yuhong Dai, ZePeng Lin, Juanxi Tian, et al. Coco: Code as cot for text-to-image preview and rare concept generation. In *European Conference on Computer Vision*, pp. 57–75. Springer, 2026a.

Haodong Li, Tianfei Ren, Xiaoxiao Ma, Chunmei Qing, Zhen Fang, Sipeng He, Ziyu Guo, Haoyu Wu, Juanxi Tian, Yihang Zou, et al. Videococo: Code-as-cot for physically-consistent video generation via an agentic dual-engine system. *arXiv preprint arXiv:2607.27380*, 2026b.

Yifan Li, Yuhang Bai, Shuai Yang, and Jiaying Liu. COCO-LC: Colorfulness controllable language-based colorization. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 10939–10947, 2024.

Zhexin Liang, Zhaochen Li, Shangchen Zhou, Chongyi Li, and Chen Change Loy. Control color: Multimodal diffusion-based interactive image colorization, 2024. URL https://arxiv.org/abs/2402.10855.

Jiaqi Liao, Zhengyuan Yang, Linjie Li, Dianqi Li, Kevin Lin, Yu Cheng, and Lijuan Wang. ImageGen-CoT: Enhancing text-to-image in-context learning with chain-of-thought reasoning. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 17214–17223. IEEE, 2025.

LightX2V Team. Qwen-Image-Edit-2511-Lightning model card. https://huggingface.co/lightx2v/Qwen-Image-Edit-2511-Lightning, 2026. Accessed: 2026-05-04.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.02747.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection. In *European Conference on Computer Vision*, 2024. URL https://www.ecva.net/papers/eccv_2024/papers_ECCV/papers/06319.pdf.

Yichen Liu, Donghao Zhou, Jie Wang, Xin Gao, Guisheng Liu, Jiatong Li, Quanwei Zhang, Qiang Lyu, Lanqing Guo, Shilei Wen, Weiqiang Wang, and Pheng-Ann Heng. HiFi-Inpaint: Towards high-fidelity reference-based inpainting for generating detail-preserving human-product images. *arXiv preprint arXiv:2603.02210*, 2026.

Alexander Lobashev, Maria Larchenko, and Dmitry Guskov. Color conditional generation with sliced wasserstein guidance. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2503.19034.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/081b08068e4733ae3e7ad019fe8d172f-Abstract-Conference.html.

Pylette Developers. Pylette: A python library for extracting color palettes from images. https://pypi.org/project/pylette/, 2026. Accessed: 2026-05-05.

Qianru Qiu, Jiafeng Mao, and Xueting Wang. Exploring palette based color guidance in diffusion models. In *Proceedings of the 33rd ACM International Conference on Multimedia*, 2025. URL https://arxiv.org/abs/2508.08754.

Qwen Team. Qwen-Image-Edit-2511 model card. https://huggingface.co/Qwen/Qwen-Image-Edit-2511, 2025. Accessed: 2026-05-04.

Royi Rassin, Eran Hirsch, Daniel Glickman, Shauli Ravfogel, Yoav Goldberg, and Gal Chechik. Linguistic binding in diffusion models: Enhancing attribute correspondence through attention map alignment. In *Advances in Neural Information Processing Systems*, volume 36, pp. 3536–3559, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/0b08d733a5d45a547344c4e9d88bb8bc-Paper-Conference.pdf.

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

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*, pp. 10684–10695, 2022.

Chenxi Ruan, Yihan Hou, Yu Xiao, Guosheng Hu, and Wei Zeng. ColorConceptBench: A benchmark for probabilistic color-concept understanding in text-to-image models, 2026. URL https://arxiv.org/abs/2601.16836.

Shay Shomer-Chai, Wenxuan Peng, Bharath Hariharan, and Hadar Averbuch-Elor. Color bind: Exploring color perception in text-to-image models. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1916–1925, 2026. doi: 10.1109/WACV61042.2026.00191. URL https://doi.org/10.1109/WACV61042.2026.00191.

Tripti Shukla, Srikrishna Karanam, and Balaji Vasan Srinivasan. Test-time conditional text-to-image synthesis using diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 4559–4568, 2026. URL https://openaccess.thecvf.com/content/CVPR2026W/A4VM/html/Shukla_Test-time_Conditional_Text-to-Image_Synthesis_Using_Diffusion_Models_CVPRW_2026_paper.html.

Ka Chun Shum, Binh-Son Hua, Duc Thanh Nguyen, and Sai-Kit Yeung. Color alignment in diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

Sung-Lin Tsai, Bo-Lun Huang, Yu-Ting Shen, Cheng Yu Yeo, Chiang Tseng, Bo-Kai Ruan, Wen-Sheng Lien, and Hong-Han Shuai. Color me correctly: Bridging perceptual color spaces and text embeddings for improved diffusion generation. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 10074–10082, 2025.

Zhenwei Wang, Nanxuan Zhao, Gerhard Hancke, and Rynson W. H. Lau. Language-based photo color adjustment for graphic designs. *ACM Transactions on Graphics*, 42(4):101:1–101:16, 2023.

Zirui Wang, Zhizhou Sha, Zheng Ding, Yilin Wang, and Zhuowen Tu. TokenCompose: Text-to-image diffusion with token-level supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 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. URL https://arxiv.org/abs/2508.02324.

Ji Xie, Luke Zettlemoyer, Xudong Wang, et al. Reconstruction alignment improves unified multimodal models. In *International Conference on Learning Representations*, volume 2026, pp. 120095–120137, 2026.

Ruihang Xu, Dewei Zhou, Fan Ma, and Yi Yang. Contextgen: Contextual layout anchoring for identity-consistent multi-instance generation. In *International Conference on Learning Representations*, volume 2026, pp. 95978–96007, 2026a.

Ruihang Xu, Dewei Zhou, Xiaolong Shen, Fan Ma, and Yi Yang. PhyEdit: Towards real-world object manipulation via physically-grounded image editing. *arXiv preprint arXiv:2604.07230*, 2026b.

Sihan Xu, Yidong Huang, Jiayi Pan, Ziqiao Ma, and Joyce Chai. Inversion-free image editing with natural language. *arXiv preprint arXiv:2312.04965*, 2023a.

Sihan Xu, Ziqiao Ma, Yidong Huang, Honglak Lee, and Joyce Chai. Cyclenet: Rethinking cycle consistency in text-guided diffusion for image manipulation. *Advances in Neural Information Processing Systems*, 36: 10359–10384, 2023b.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, et al. Qwen3 technical report, 2025a. URL https://arxiv.org/abs/2505.09388.

Yuqi Yang, Dongliang Chang, Yuanchen Fang, Yi-Zhe Song, Zhanyu Ma, and Jun Guo. Controllable-continuous color editing in diffusion model via color mapping, 2025b. URL https://arxiv.org/abs/2509.13756.

Yuqi Yang, Dongliang Chang, Yijia Ling, Ruoyi Du, and Zhanyu Ma. Recolour what matters: Region-aware colour editing via token-level diffusion. In *European Conference on Computer Vision*, 2026. URL https://eccv.ecva.net/Conferences/2026/AcceptedPapers. Accepted for publication.

Hu Ye, Jun Zhang, Sibo Liu, Xiao Han, and Wei Yang. IP-Adapter: Text compatible image prompt adapter for text-to-image diffusion models. *arXiv preprint arXiv:2308.06721*, 2023. URL https://arxiv.org/abs/2308.06721.

Junyan Ye, Jun He, Zilong Huang, Dongzhi Jiang, Xuan Yang, Rui Chen, and Weijia Li. Genclaw: Code-driven agentic image generation. *arXiv preprint arXiv:2605.30248*, 2026a.

Junyan Ye, Wei Liu, Dongzhi Jiang, Zichen Wen, HaoDong Li, Zhutao Lv, Jiaxin Lin, Jinhua Yu, Jun He, Zilong Huang, et al. Editable visual design. *arXiv preprint arXiv:2609.04034*, 2026b.

Xingxi Yin, Zhi Li, Jingfeng Zhang, Chenglin Li, and Yin Zhang. ColorEdit: Training-free image-guided color editing with diffusion model, 2024. URL https://arxiv.org/abs/2411.10232.

Zixin Yin, Xili Dai, Ling-Hao Chen, Deyu Zhou, Jianan Wang, Duomin Wang, Gang Yu, Lionel M. Ni, Lei Zhang, and Heung-Yeung Shum. Training-free text-guided color editing with multi-modal diffusion transformer. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/59f278de1619bdb6b53fd04e8e0976e0-Abstract-Conference.html.

Z-Image Team, Huanqia Cai, Sihan Cao, Ruoyi Du, Peng Gao, Aiming Hao, Steven Hoi, Zhaohui Hou, Shijie Huang, Dengyang Jiang, Yuming Jiang, Xin Jin, Liangchen Li, Zhen Li, Zhong-Yu Li, David Liu, Dongyang Liu, Qilong Wu, Feng Yu, Zechao Zhan, Chi Zhang, Shifeng Zhang, Ruikai Zhou, and Shilin Zhou. Z-Image: An efficient image generation foundation model with single-stream diffusion transformer, 2025. URL https://arxiv.org/abs/2511.22699.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3836–3847, 2023.

Yabo Zhang, Kunchang Li, Dewei Zhou, Xinyu Huang, and Xun Wang. Images in sentences: Scaling interleaved instructions for unified visual generation. *arXiv preprint arXiv:2605.12305*, 2026.

Zechuan Zhang, Ji Xie, Yu Lu, Zongxin Yang, and Yi Yang. Enabling instructional image editing with in-context generation in large scale diffusion transformer. In *Advances in Neural Information Processing Systems*, volume 38, pp. 154372–154404, 2025. doi: 10.52202/085713-4646. URL https://papers.nips.cc/paper_files/paper/2025/hash/cb7b55efe349d67ff3ac244aa1ae49d7-Abstract-Conference.html.

Dewei Zhou, Ji Xie, Zongxin Yang, and Yi Yang. 3dis: Depth-driven decoupled instance synthesis for text-to-image generation. *arXiv preprint arXiv:2410.12669*, 2024a.

Dewei Zhou, Mingwei Li, Zongxin Yang, and Yi Yang. Dreamrenderer: Taming multi-instance attribute control in large-scale text-to-image models. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 16712–16722. IEEE, 2025.

Dewei Zhou, Xinyu Huang, Xun Wang, Ji Xie, Yabo Zhang, Liang Li, Kunchang Li, Zongxin Yang, and Yi Yang. Metapoint: Unlocking precise spatial control in agentic visual generation. In *European Conference on Computer Vision*, pp. 373–390. Springer, 2026a.

Dewei Zhou, You Li, Zongxin Yang, and Yi Yang. Refineanything: Multimodal region-specific refinement for perfect local details, 2026b. URL https://arxiv.org/abs/2604.06870.

Donghao Zhou, Jiancheng Huang, Jinbin Bai, Jiaze Wang, Hao Chen, Guangyong Chen, Xiaowei Hu, and Pheng-Ann Heng. MagicTailor: Component-controllable personalization in text-to-image diffusion models. *arXiv preprint arXiv:2410.13370*, 2024b.

Donghao Zhou, Guisheng Liu, Hao Yang, Jiatong Li, Jingyu Lin, Xiaohu Huang, Yichen Liu, Xin Gao, Cunjian Chen, Shilei Wen, Chi-Wing Fu, and Pheng-Ann Heng. OmniShow: Unifying multimodal conditions for human-object interaction video generation. *arXiv preprint arXiv:2604.11804*, 2026c.

Supplementary Material Overview

We provide additional information in the supplementary material, as outlined below:

- Sec. A: How ACBench relates to existing color benchmarks, with the GenColorBench NCU transfer check.
- Sec. B: ACBench construction, prompt sampling and baseline settings.
- Sec. C: The hex-translated CompColor protocol and the full original baseline roster.
- Sec. D: Paint-500K composition, the training mixture and the tools used to build it.
- Sec. E: Pure-color generation trajectories motivating the high-noise gate.
- Sec. F: Prompt templates for grounding, filtering and recaptioning.
- Sec. G: Evaluator settings used for every reported score.
- Sec. H: ACBench reliability under alternative region statistics, SAM3 thresholds and localization failures.
- Sec. I: Pure-color anchor resolution ablation.
- Sec. J: Editing performance as a function of source-to-target color distance and on unnatural target colors.
- Sec. K: Human preference study protocol and results.
- Sec. L: Training and evaluation separation check.
- Sec. M: Additional training-data examples.

A Relation to Existing Color Benchmarks

Existing benchmarks expose related weaknesses in color-controllable generation. ColorBind introduces CompColor for compositional color binding (Shomer-Chai et al., 2026), GenColorBench broadens evaluation to many color-control settings including numerical colors (Butt et al., 2026a), and ColorConceptBench targets implicit color-concept understanding (Ruan et al., 2026). ACBench complements these efforts by focusing on explicit 24-bit hex-to-object fidelity for both generation and editing.

GenColorBench NCU. We also evaluate under the official GenColorBench Numerical Color Understanding (NCU) protocol (Butt et al., 2026a): four images per RGB/HEX prompt at each model's default sampling steps and resolution, GroundingDINO+SAM2 localization, OneHue dominant-color extraction, and CIEDE2000 scoring with the released color-neighborhood tolerance. Table 4 reports the result alongside the per-category scores published by NumColor (Butt et al., 2026b), whose evaluation setting we follow.

Model	L1	L3	CSS3/X11	Avg. ↑
`FLUX.2-4B` Base	34.71	34.64	32.64	34.00
Paint-Anything	**58.15**	**58.54**	**56.98**	**57.89**
FLUX + NumColor†	55.71	48.02	51.96	51.90
SD3.5 + NumColor†	51.72	43.35	46.87	47.31
SD3 + NumColor†	49.23	39.46	47.19	45.29
PixArt-Σ + NumColor†	48.56	40.83	43.90	44.43
PixArt-α + NumColor†	44.93	36.17	38.51	39.87

Table 4: GenColorBench NCU, following the evaluation setting of NumColor (Butt et al., 2026b). L1 and L3 denote ISCC-NBS levels, and Avg. is their arithmetic mean with CSS3/X11. †: scores quoted from NumColor (Butt et al., 2026b), Table 1, not rerun here.

On the same FLUX.2-4B backbone, finetuning improves mean NCU from 34.00 to 57.89, a gain of 23.89 points, with improvements in all three categories. Our mean also exceeds the published FLUX + NumColor result (51.90) by 5.99 points. GenColorBench NCU remains a single-object T2I numerical-color test. ACBench also covers two-object binding and image editing, while CompColor provides an independent compositional-binding diagnostic.

B ACBench Construction and Evaluation

Task composition and prompts. ACBench-T2I contains 1000 prompts over common object categories: 500 specify one object and one target hex color, and 500 specify two objects with distinct target hex colors. ACBench-Edit contains 500 real-image recoloring prompts, each specifying one target object and its requested hex color. The prompt forms are illustrated in Section 4. We check these prompts for exact-string overlap with Paint-500K in Appendix L.

Editing images and masks. The 500 editing source images come from an internal dataset of high-quality real images that does not overlap with the Paint-500K source images. Masking follows Section 4, where we reuse one manually inspected source mask so that the evaluated region stays fixed across methods.

Color targets and score interpretation. We sample target colors at random from the 24-bit RGB space and attach each target hex to its corresponding object, with distinct target colors in two-object prompts. Scoring uses the MAE-to-score mapping of Section 4, and we analyze its sensitivity to the color estimator and to localization in Appendix H.

Relation to prior evaluation protocols. Following the object-centric evaluation approach of CompColor (Shomer-Chai et al., 2026) and GenColorBench (Butt et al., 2026a), we localize the requested object before assessing its color. ACBench uses its own SAM3 and sRGB-MAE protocol. For independent evaluation, we use the released CompColor evaluator without modification and the official GenColorBench NCU protocol, including its dominant-color extraction and CIEDE2000-based scoring (Appendices C and A).

B.1 Baseline Settings and Result Sources

For reproduced baselines, we follow the default configurations of their official open-source implementations and Hugging Face releases, except for the matched FLUX.2-4B editing CFG of 2.0 stated in the main paper.

Prompt interfaces and result provenance. ACBench specifies numerical target colors, while CompColor uses named-color prompts unless a row is marked "Hex." The latter replaces the color names with their canonical hex values while preserving object compositions. Color-specialized methods retain their respective model backbones and control interfaces. Among them, ColorWave is our reproduction, as marked in Table 1. We quote the SD1.5 and FLUX.1 CompColor scores in Table 1 from ColorBind, as marked on those cells, but evaluate both models on ACBench ourselves. Additional quoted CompColor baselines appear in Appendix C. NumColor NCU results are arithmetic means of its three published category scores, rather than ACBench reruns (Appendix A). Cross-model results compare complete systems, whereas same-backbone finetuning and ablation comparisons assess our training recipe.

C CompColor Hex Protocol

Benchmark background. CompColor is the compositional color-binding benchmark released by ColorBind (Shomer-Chai et al., 2026), originally designed to expose a failure mode of T2I models: when a prompt mentions *multiple* colored objects, generators often leak color across objects, swap the two assignments, or collapse both objects to a shared dominant hue. CompColor isolates this failure mode by holding the textual structure fixed, varying only the (color, object) bindings, and measuring per-object color fidelity. Each prompt follows the template "a {color1} colored {object1} and a {color2} colored {object2}".

Color palette and pair construction. The released benchmark draws color words from a fixed *named-color* palette (e.g., *tomato*, *royal-blue*, *light-cyan*, *hot-pink*) whose canonical RGB triplets are known. For each pair of color names, the perceptual distance between their canonical sRGB values is computed in CIELAB space (denoted ΔE^*_{ab} in the original paper). Color pairs are then partitioned by this distance into a *Close* split, where the two colors are perceptually similar and easy to confuse (e.g., *sky-blue* vs. *light-cyan*), and a *Distant* split, where the two colors are perceptually well-separated (e.g., *sky-blue* vs. *hot-pink*). The benchmark also includes two *Single* subsets (one color $\times$ one object), used to verify that single-object color rendering still works in isolation.

Object compositions. Object pairs are sampled from a curated list of common everyday categories (animals, vehicles, garments, household items, food) so that the requested colors are physically plausible and segmentable. The original benchmark fixes both color words and object words *up front* and ships the prompts as a closed set, so all baselines are evaluated on exactly the same prompts and seeds. We respect this convention: our hex protocol modifies only the color tokens, never the object tokens or pair structure.

Evaluation metric. The original CompColor metric estimates, for each generated image, whether each requested object was rendered in the requested color. In practice this combines an object localizer (a segmenter applied to each object phrase) with a color-similarity check between the segmented region and the target RGB value. The released scores are normalized to a 0 to 1 scale, where higher means better object-level color binding. For CompColor results that we evaluate, including our named-color and hex-translated rows, we run the ColorBind open-source evaluation codebase (Shomer-Chai et al., 2026) without modification. Concretely, the pipeline uses LangSAM, which combines Grounding DINO (Liu et al., 2024) with SAM (Kirillov et al., 2023), to locate each object from its text label and produce a segmentation mask, and then scores color with the original k-means-plus-ΔE_{CMC} protocol.

Hex translation. We keep the original object compositions and the *Single*/*Close*/*Distant* splits untouched, and replace only the color words in each prompt with explicit 24-bit hex strings derived from the canonical RGB value of the original color name (e.g., *tomato* → *#FF6347*, *royal-blue* → *#4169E1*). Named-color baselines and our hex-conditioned rows therefore describe the same perceptual targets, so within-model differences isolate the change from color words to hex specifications. This is a stricter test than CompColor's named-color setting: hex strings remove the explicit named-color cue ("*tomato*" suggests a reddish hue), while retaining any numerical-color knowledge already present in the pretrained text encoder.

Subset aggregation. The released benchmark provides two single-object subsets evaluated separately in the original paper. To match the score format used by the most recent baseline tables, we average them into a single *Single* column in Table 1 and report the *Close* and *Distant* columns as is. In Table 6 we further report the unweighted mean of Single, Close, and Distant as an *Avg* column, matching the original paper's averaging convention.

Three-object extension. CompColor was originally proposed as a two-object compositional benchmark, and the released baseline scores cover only this setting, which is the one reported in the main paper. We also evaluate the same wrapped-hex interface on CompColor's three-object extension (Table 5).

Protocol	*Paint-Anything*	`FLUX.2-4B` Base
Valid disjoint masks	0.763	0.362
Missing/overlapping masks score zero	0.506	0.210

Table 5: CompColor three-object extension evaluated under the wrapped-hex interface.

Paint-Anything remains ahead under both protocols. The evaluator scores the designated target object, so this experiment measures color control in scenes containing three colored objects rather than strict three-way joint correctness.

Score format. All CompColor hex-protocol scores follow the original 0 to 1 convention, while ACBench scores follow our 0 to 100 percentage-point scale. The two benchmarks are not numerically comparable in Table 1: CompColor measures *compositional binding* on a closed prompt set, while ACBench measures *object-level hex color fidelity* on broader generation and editing protocols.

Full original baseline roster. The main paper Table 1 keeps the modern T2I/edit baselines that overlap with ACBench so that the same backbone family can be tracked across both benchmarks. Table 6 reports the remaining original CompColor baselines released by ColorBind (Shomer-Chai et al., 2026), namely Attend-and-Excite (Chefer et al., 2023), Structured Diffusion (Feng et al., 2023), SynGen (Rassin et al., 2023), RichText (Ge et al., 2023) and Bounded Attention (Dahary et al., 2024), with the same (Single, Close, Distant, Avg) format as the original paper. We copy these rows directly from the release, which uses named-color prompts only.

Method	Single ↑	Close ↑	Distant ↑	Avg ↑
SD 1.4	0.51	0.38	0.28	0.39
SD 2.1	0.61	0.33	0.26	0.40
Attend-and-Excite	0.55	0.46	0.38	0.46
Structured Diffusion	0.58	0.39	0.39	0.45
SynGen	0.45	0.49	0.56	0.50
RichText	0.29	0.35	0.30	0.31
Bounded-Attention	0.43	0.55	0.29	0.42

Table 6: Remaining original CompColor baselines from ColorBind (Shomer-Chai et al., 2026), omitted from the main paper Table 1. Scores follow the released 0 to 1 convention.

D Training Data and Mixture

Paint-500K contains 500K color-control samples built from an internal, non-public collection of high-quality real images. It is organized into two streams: 400K T2I generation samples and 100K editing samples. The T2I stream contains 100K single-object color generation samples and 300K multi-object color generation samples. Rows include target image paths, object prompts, source image paths for editing, object masks, target hex colors, and dominant-color metadata. We use Seed1.8 (ByteDance Seed Team, 2026) as the VLM for grounding, filtering, and recaptioning, and use Qwen-Image-Edit-2511 (Wu et al., 2025; Qwen Team, 2025) with the 4-step Qwen-Image-Edit-2511-Lightning LoRA (LightX2V Team, 2026) as the editing model.

Final training uses three streams in total: Paint-500K T2I generation, Paint-500K editing, and auxiliary pure-color grounding. Each training batch contains 30 real-image T2I samples, 12 pure-color samples, and 30 editing samples, giving a total batch size of 72. We set $\lambda_{\text{t2i}} = \lambda_{\text{edit}} = \lambda_{\text{rgb}} = 1$ in all experiments. The auxiliary pure-color stream contains 10K solid-color samples at 512×512 resolution, which we train only at timesteps $t \in [0.8, 1.0]$ in the final recipe. All 4B finetuning and ablation rows use Adam, learning rate 2×10^{-5}, and 4000 steps. T2I generation with `FLUX.2-klein-base-4B` uses the default classifier-free guidance of 4.0 unless otherwise noted, and all FLUX.2-4B editing comparisons, including the base model and finetuned variants, use the same CFG of 2.0.

Construction details. For instance grounding, each VLM proposal contains a text label, a short region description, and a bounding box. SAM3 then segments each grounded box so that we extract colors from object pixels. We normalize the CIELAB channels to $[0, 1]$ and cluster masked pixels with MeanShift using bandwidth 0.05. Two filters then decide which colors are retained: each retained cluster must cover at least 15% of the object pixels (`COLOR_MIN_RATIO` $= 0.15$), and the retained clusters must jointly cover at least 90% of the object pixels (`INSTANCE_KEEP_RATIO` $= 0.90$), or we discard the instance. Small highlight, shadow, or decoration regions are therefore removed, while multiple substantial colors on one object can be retained. When a single dominant color is required, we take the retained cluster with the largest pixel mass and convert it to a 24-bit sRGB hex label. In a spot check, we randomly inspected 100 Paint-500K samples, and 96 of their dominant-color labels agreed with the human-perceived dominant object color. In a second check, five annotators independently assessed target-object colors across the color gamut, and the MAE computed from their annotations differed from the MAE of the automatic extraction by 15.37%.

When several objects share the same semantic label, VLM grounding returns separate bounding boxes for each instance, including same-category objects with different colors (Figure 4), and we extract colors independently inside each box. During recaptioning, we feed the original caption, grounded layout, text labels, and extracted hex colors back to the VLM, which aggregates the localized instances, colors, and spatial relations into a complete caption and inserts exact `<color>#HEX</color>` spans into the relevant noun phrases. These captions are the multi-object T2I prompts and can describe, for example, a red object on the left and a green object of the same category on the right. For single-object T2I samples, we start from instances whose dominant color covers a large fraction of the mask, randomly expand the instance box, crop a local image region, and recaption the crop around the selected object and its dominant hex color.

For editing data, we keep a synthesized source-target pair only when the mask-outside change is small enough to preserve the surrounding scene and the mask-inside change is large enough to verify a meaningful color edit. VLM verification and recaptioning further require that the target object remains identifiable, the requested color change is visually grounded, and any visible texture, material, pattern, or shape changes are reflected in the final instruction while preserving the exact target hex string.

E Pure-Color Generation Trajectories

Figure 10 explores when color emerges in two pure-color generation trajectories, giving a qualitative motivation for high-noise anchor supervision. The downstream evidence comes from the gate ablation in Table 3, where $t_{\text{gate}} = 0.8$ improves both generation and editing over ungated anchors.

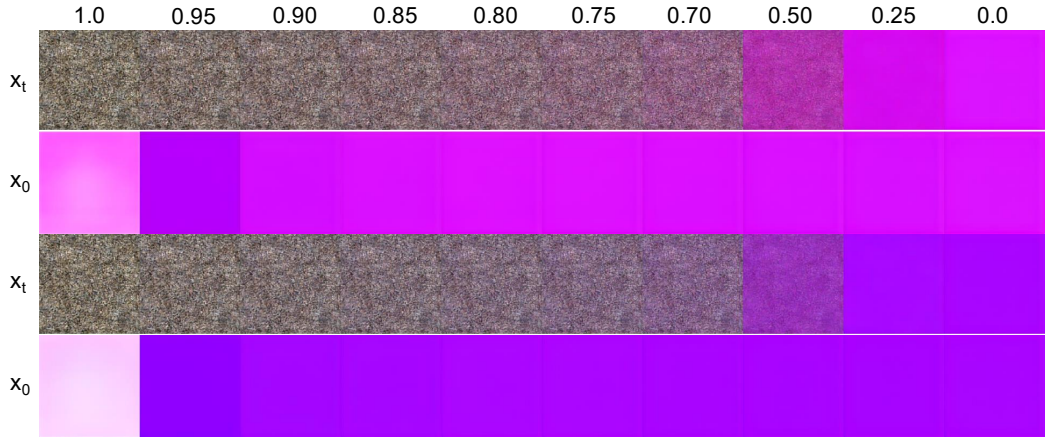


Figure 10: **Early color stabilization in pure-color generation.** Two trajectories visualize noisy states (rows labeled x_t) and corresponding single-step denoised predictions (rows labeled x_0) at decreasing noise levels. In these examples, the predicted color is already visually stable near $t \approx 0.8$.

F Prompt Templates and Data-Generation Tools

We used fixed prompt templates for VLM-based caption refinement, instance grounding, and edit-instruction refinement. The color-caption refinement system prompt was:

```
Role: You are a visual data refinement expert specializing in integrating
precise color attributes into image captions for VLM training.

Task: Integrate detected objects and their specific HEX color codes into a
global caption. Use the provided bounding box coordinates only to determine
spatial relationships and distinguish between multiple objects, but do not
include the numerical coordinates in the final output.

Constraints & Rules:

No Bbox in Output: Do not include any coordinate values (x1, y1, x2, y2)
or bbox IDs in the final caption. Use them only to infer positions
(e.g., "on the left," "in the background").
Preserve Context: Maintain the original global caption's narrative flow
and structure.
Disambiguation: If multiple similar objects exist, use their relative
spatial positioning to make the description unique
(e.g., "The <color>#HEX</color> bottle on the right").
Natural Integration: Insert the color tags directly before or after the
object they describe so the sentence remains grammatically fluent.
Return Format: "[Final enhanced caption text with <color>HEX code</color>
tags and no coordinates]"
Strict HEX Usage: You must use the EXACT HEX color codes provided in the
input. Do not invent or hallucinate colors that are not in the input
(e.g., #000000, #FFFFFF).

Example for your reference:
```

```
User Input:
Global Caption: A cat sitting on a sofa near a lamp.
Bounding Boxes:
id: 1 | label: cat | color: #4A4A4A | region_caption: a cat |
coords: [100, 200, 300, 400]
id: 2 | label: sofa | color: #F5F5DC | region_caption: a sofa |
coords: [0, 150, 800, 900]
id: 3 | label: lamp | color: #FFD700 | region_caption: a lamp |
coords: [700, 50, 850, 500]

Desired Output:
"A <color>#4A4A4A</color> cat is sitting on a <color>#F5F5DC</color>
sofa, positioned next to a <color>#FFD700</color> lamp on the right."
```

For instance grounding, we used the following prompt template:

```
Role: You are a visual instance grounding expert for precise color-control
data construction.

Task: Given an image and its global caption, identify caption-relevant
object instances that can support reliable object-level color labeling.
Return only objects that are visually present, localizable, and useful for
color extraction. For each instance, return a normalized bounding box.

Constraints & Rules:

Caption Faithfulness: Prefer objects explicitly mentioned in the caption.
You may include visually salient objects not mentioned in the caption only
when they are necessary to preserve the scene context.
Instance Separability: Split multiple similar objects into separate
instances when they can be distinguished by position, size, or appearance.
Spatial Disambiguation: Use relative positions such as "left", "right",
"front", "background", "upper", or "lower" to disambiguate instances.
No Hallucination: Do not output objects that are not clearly visible.
Color Suitability: Prefer objects with a coherent dominant surface color.
Avoid transparent, reflective, heavily shadowed, tiny, or highly textured
objects when their color cannot be reliably measured.
Normalized Bbox: Return each bounding box as normalized coordinates on a
0 to 1000 scale, in the order x1, y1, x2, y2.
Bbox Tags: The bbox value must be wrapped with <bbox></bbox>, for example
"bbox": "<bbox>[120, 85, 640, 730]</bbox>".
No Coordinates in Captions: Coordinate values must appear only in the
"bbox" field, not inside natural language captions or descriptions.
Return Format: Return only a valid JSON list. Each item must contain
"label", "instance_caption", "visual_description",
"spatial_description", and "bbox".
Example:
[
  {
    "label": "sky",
    "instance_caption": "the blue sky in the background",
    "visual_description": "a large coherent sky region",
    "spatial_description": "upper background",
    "bbox": "<bbox>[0, 0, 1000, 420]</bbox>"
  }
]
```

For edit-instruction refinement, we used the following prompt:

```
You are an image edit prompt enhancement assistant.

You are given:
- an original edit prompt,
- a source image,
- an edited image,
- and the target object label.

Your task is to enhance the original edit prompt, not rewrite it from scratch.

Rules:
1. Preserve the original editing intent.
2. Always preserve the original hex color value exactly as written in the
original prompt.
3. Do not replace the hex color with a natural language color name.
4. Improve wording so the prompt is natural, concise, and fully in English.
5. Keep the enhanced prompt under 30 words.
6. Mention texture, material, pattern, or shape changes only if they are
clearly visible in the edited image compared with the source image.
7. Do not add extra changes unless they are visually clear.
```

```
8. If the original prompt already matches the edit well, make only minimal
improvements.
9. If image quality is too poor to judge reliably, return exactly: error
10. Return only the enhanced prompt as one short sentence, or exactly: error
11. The output must contain the same hex color string from the original
prompt exactly once.
12. Do not output explanations.
```

G Evaluator Settings

For both ACBench-T2I and ACBench-Edit, SAM3 (Carion et al., 2026) localizes the target object, and we compute scores from the masked pixels: in T2I we segment the generated target object from the prompt, and in editing we segment the source object and reuse the source mask on the edited image without dilation. The predicted color is the mean RGB value inside the mask, and the final error is its channel-wise MAE to the target hex color. Samples with failed target localization receive zero score. Like CompColor, whose evaluator uses LangSAM and its released color-scoring protocol (Appendix C), the evaluation is object-level, but the main paper metric replaces its thresholded LAB accuracy with the continuous hex-fidelity score defined in Section 4. Object-centric evaluation with pretrained detectors or segmenters has established precedents in GenEval (Ghosh et al., 2023), T2I-CompBench (Huang et al., 2023), Grounded SAM (Ren et al., 2024), and TokenCompose (Wang et al., 2024).

H ACBench Reliability

ACBench's primary score uses mean RGB inside the target mask followed by channel-wise MAE. Region averages can be affected by shading, material, and mask boundaries, so we test whether the paper's ranking depends on this particular estimator, on SAM3 thresholds, or on localization failures.

Region-statistic robustness. We recompute seven T2I model/configuration comparisons and three editing models over 4,656 object instances with five estimators: Mean RGB (paper metric), MeanShift dominant RGB, Median RGB, Median Lab, and Dominant Lab (largest CIELAB k-means cluster). The alternatives cover the color-quantization and RGB/CIELAB evaluation used by ColorBind (Shomer-Chai et al., 2026) and the dominant-hue Lab evaluation used by GenColorBench (Butt et al., 2026a). Table 7 reports T2I scores on the 0 to 1 scale used by the robustness script, which is the paper's 0 to 100 convention divided by 100.

Model / Configuration	Mean RGB	MeanShift	Median RGB	Median Lab	Dominant Lab
Paint-Anything, $t_{\text{gate}} = 0.8$	0.6858	0.6954	0.7341	0.7254	0.7228
Paint-Anything, $t_{\text{gate}} = 0.9$	0.6754	0.6800	0.7122	0.7176	0.7228
Paint-Anything, $t_{\text{gate}} = 0.0$	0.6383	0.6744	0.7053	0.6965	0.6931
`FLUX.2-9B` Base	0.3815	0.3734	0.4082	0.3930	0.4118
`FLUX.2-4B` Base	0.3702	0.3713	0.3886	0.3853	0.3943
Z-Image-Turbo	0.3356	0.3395	0.3703	0.3546	0.3568
Qwen-Image	0.2521	0.2860	0.2975	0.2945	0.3003

Table 7: ACBench-T2I under alternative region-color estimators (0 to 1). The ordering is preserved for MeanShift, Median RGB, and Median Lab. Dominant Lab ties the two strongest variants at the displayed precision, but every estimator preserves their advantage over the base models.

The same ranking holds for ACBench-Edit, where *Paint-Anything*, Base and Qwen-Image-Edit score 0.7557, 0.5890 and 0.5440 under Mean RGB and every alternative estimator preserves this order (Spearman ρ=1.0). The T2I gap between *Paint-Anything* and Base remains large under every estimator: $+0.3156$ (Mean RGB), $+0.3241$ (MeanShift), $+0.3455$ (Median RGB), $+0.3401$ (Median Lab), and $+0.3284$ (Dominant Lab).

SAM3 threshold sensitivity. With all other evaluator settings fixed, we sweep the SAM3 detection threshold from 0.1 to 0.9 and four coverage gates $(\text{min cluster}, \text{min total}) \in \{(0.15, 0.70), (0.10, 0.60), (0.05, 0.30), (0, 0)\}$, yielding $9 \times 4 = 36$ combinations. Table 8 reports the default-coverage slice. *Paint-Anything* ranks above `FLUX.2-4B` Base at every threshold, with a margin from $+0.2482$ to $+0.3156$. Across the full 36-setting grid the gap ranges from $+0.2482$ to $+0.3308$.

SAM3 detection threshold	*Paint-Anything*	`FLUX.2-4B` Base	Difference
0.1 to 0.5	0.6858	0.3702	+0.3156
0.6	0.6837	0.3696	+0.3141
0.7	0.6760	0.3642	+0.3118
0.8	0.6623	0.3526	+0.3097
0.9	0.5659	0.3177	+0.2482

Table 8: ACBench-T2I versus SAM3 detection threshold at default coverage $15\%/70\%$ (0 to 1).

Localization-failure audit. To measure how often the target object cannot be localized, we set a SAM3 confidence threshold and score failures as zero. After verifying that high-confidence masks correctly cover the target on 20 random samples per model, we draw 600 images per model from the 1,000 ACBench-T2I outputs and measure failure rates (Table 9).

Model	Failures / 600	Failure rate
`FLUX.2-4B` Base	25	4.17%
`FLUX.2-9B` Base	14	2.33%
Qwen-Image	10	1.67%
Z-Image-Turbo	1	0.17%
Paint-Anything	33	5.50%

Table 9: Localization failure rates over 600 ACBench-T2I images per model.

Manual inspection attributes only 11% of failures to SAM3, and most arise from the generator not producing a localizable target object. *Paint-Anything* has the highest failure rate among the listed models, so the zero score for failures does not favor it in this audit. Excluding all failures raises *Paint-Anything* from 68.58 to 70.93 without changing the model ranking. For ACBench-Edit, all models share the same human-verified source mask, so inter-model mask quality is matched by construction.

I Pure-Color Anchor Resolution

Holding the remaining training recipe fixed and training for 4,000 steps, we vary only the solid-color anchor resolution (Table 10).

Anchor resolution	ACBench-T2I	ACBench-Edit
256×256	63.10	73.82
512×512 (default)	68.58	75.57
1024×1024	69.52	76.03

Table 10: Pure-color anchor resolution, with the rest of the recipe held fixed.

Dropping to 256×256 costs 5.48 points on ACBench-T2I and 1.75 on ACBench-Edit. Raising the resolution to 1024×1024 adds only 0.94 and 0.46 points at a higher training cost, so we keep 512×512 as the default.

J Editing under Large Color Shifts and Unnatural Colors

To examine editing under different source-to-target color shifts, we analyze a subset of ACBench-Edit. We divide this subset into four equal-sized quartiles by the CIEDE2000 distance between the estimated source-object color and the requested target, and separately isolate near-complementary cases where both colors are chromatic and the hue rotation exceeds $120°$. Table 11 reports scores (0 to 100) on this subset, and Table 1 reports the full-benchmark scores.

Model	Q1 (nearest)	Q2	Q3	Q4 (farthest)	Near-comp.
Paint-Anything	78.8	77.1	79.2	88.9	86.9
`FLUX.2-4B`	67.6	64.2	62.9	56.3	50.8

Table 11: ACBench-Edit scores (0 to 100) by source-to-target CIEDE2000 distance.

Paint-Anything remains effective in every bucket and is strongest on the farthest quartile and the near-complementary subset, while the base model degrades as the required color shift grows. CompColor Distant under the hex-translated protocol shows the same pattern for target-pair contrast (Section 4).

Unnatural target colors. Among the 500 ACBench-Edit instructions, we identified 33 that recolor an object to an unnatural color. *Paint-Anything* scores 72.98 on this subset, close to its overall ACBench-Edit score of 75.57. The subset covers object recoloring only and contains no large scene regions such as a green sky.

K User Study

We collect color-preference judgments from 15 participants on 80 randomly sampled ACBench-T2I prompts. Each trial shows a target swatch and two anonymized images from *Paint-Anything* and `FLUX.2-4B`. Across $15 \times 80 = 1200$ judgments, *Paint-Anything* is preferred in 55%, tied in 32%, and loses in 13%, a non-tie win rate of $55/(55 + 13) = 80.9\%$. Across 200 paired image-quality judgments, *Paint-Anything* wins 22%, ties 58%, and loses 20%, giving a 52.4% non-tie win rate. The color preferences agree in direction with the ACBench comparison.

L Training and Evaluation Separation

We construct evaluation prompts independently, in formats different from the training captions. For the exact overlap check, we normalize the benchmark prompt strings and the object prompt, T2I prompt, and edit instruction fields of Paint-500K in the same way, by trimming whitespace, lowercasing, and collapsing internal whitespace. Under this normalization, no benchmark prompt exactly matches a training prompt.

M Additional Training-Data Examples

We show 20 T2I examples and 20 editing pairs selected for visual diversity from the available training-data sample pools. For display, we translate the text from the original Chinese annotations and shorten the editing captions for readability, keeping the target hex values unchanged.



A nearly black #020806 background fills most of the frame. A little left of center, a woman in a red #961E0A long dress holds several red roses in both hands and looks down at them. Red fabric spreads out below and in front of her, creating a strongly atmospheric composition.



A #E1E9EB sky stretches above distant #485152 mountains. Dark green conifers and warm yellow shrubs stand in front of the mountains. A small, pale beige calf with curly hair stands at the center, looking toward the camera. In the foreground, grass is #292E23. A soft, warm light creates a peaceful, soothing atmosphere.



A winter forest with #BCBEBD snow in the foreground and snow-covered fallen branches scattered across the ground. #4D4A4A tree trunks fill the middle and rear of the scene, with warm sunlit highlights on some trunks. Darkness fills the deeper forest. Beams of sunlight cast shadows of varying intensity across the snow, creating a quiet, chilly woodland scene.

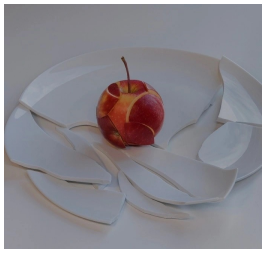


A naturalistic wildlife photograph in green tones. A brown rabbit sits near the lower center of the foreground, ears upright and gaze turned left. It rests on #A8B869 grass. A background of green grass and blurred vegetation surrounds the rabbit. A shallow depth of field and soft natural light create a calm, leisurely atmosphere.



A rustic still life on a wooden tabletop in warm sunlight. A pair of old sheets of paper lies in the foreground, with two yellow ears of corn near their upper center. Behind the corn, an off-white teapot sits on the left and a jar with a red fu character sticker sits on the right. A small black #050609 container stands at the lower right, beside the papers.

Figure 11: **T2I training-data examples (1/4).** Five image–caption examples with target hex colors.



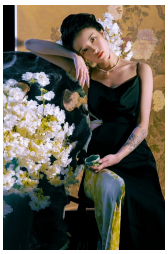
An artistic composition on a plain, neutral #898A92 background. A broken white plate in #838792 sits at the center, with fragments scattered around an apple. A red #571217 apple with yellow-orange stripes rests in the middle of the broken plate. At the center, the apple has been cut into several pieces and reassembled, with its stem still attached at the top.



Rice cake slices in #DECFC0 are stacked on a wooden plate at the center, topped with black sesame seeds, chopped nuts, and dried red fruit. At the center, the plate rests on coarse, dark brown fabric in #462A26 and #1E100E, which fills the background. An edge of a woven basket appears at the upper left. Red chilies decorate the upper and lower areas of the fabric, creating a rustic Chinese snack still life.



Outdoors, a young woman with two braids stands at the center, smiling and turning her face toward the left. Facial skin has #C6A397 and #825F4D tones. A pale pink top, visible in the lower frame, has small flutter sleeves and includes #B6B1B8 and #F6F1F2. Blurred green plants, scattered sunlight, and fine strands of hair lifted by the wind create a fresh, soft atmosphere.



A contemporary Chinese-style portrait. On the right, a woman leans against a dark textured counter on the left, resting her left arm on it and holding a pale cyan teacup in her right hand, and she wears a black #010607 strappy long dress with a yellow-and-white printed lower garment, a fine necklace, and a tattoo on her left arm. Clusters of white flowers fill the left foreground. A beige painting with meticulous floral patterns hangs behind her in soft, elegant light.



A close-up of a white lotus in full bloom at the center, with yellow stamens, broad open petals, and several supporting green stems. A large lotus leaf in #A2CDA5 and #B9E0C4 extends over the flower at the top of the frame. More leaves and stems fill the blurred lotus-pond background, whose main tone is #ABD2A0. Overall, the scene feels fresh and soft.

Figure 12: **T2I training-data examples (2/4).** Five image–caption examples with target hex colors.



A nostalgic Mid-Autumn food close-up. Against a blurred background in #798275 and #535F56, a chipped green table supports a black #080403 tray at the center. At the center, the tray holds four golden Cantonese mooncakes with baked reddish-brown patterns and stamped text such as double yolk. An edge of a dark woven food basket appears at the upper right.



A traditional Chinese portrait with low saturation and a shallow depth of field. Against a white #F5F7FA background, a woman sits sideways at the center, looking slightly downward with her hands folded in front. Black #0C0E08 hair forms a high bun with gold ornaments. An outer hanfu is dark green in #192215 and #063F43, with subtle sleeve patterns and an orange inner layer. An elegant, quiet atmosphere fills the scene.



A painted portrait of a serious, contemplative woman at the center, looking toward the left. A mass of long golden hair in #778271 covers her shoulders. A dark top combines #403D2F and #322020 and fills the lower part of the frame. Abstract red and gray-green brushstrokes surround her in the background, emphasizing the figure and the painting's handmade texture.

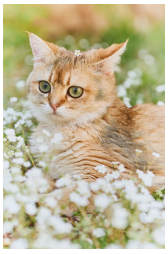


At dusk, a soft pink-purple sky in #C47B7C and #876879 fills the upper frame. Himeji Castle stands at the center, with a white main structure, several tiers of dark gray roofs with upturned eaves, and a stone foundation. Dense #141612 trees fill the foreground and surround the lower part of the castle, creating a quiet, beautiful scene with a traditional atmosphere. A low-angle view of an ancient stone fortress in a desert landscape



beneath a broad #CEDBEC sky that fills the upper half of the frame. Its weathered walls are mainly #816B5F with areas of #5C3D2C. Round towers rise above the walls, flags hang from them, and fires burn in some areas. A palm tree stands at the lower right. A sandy foreground contains stones and ruined building fragments.

Figure 13: **T2I training-data examples (3/4).** Five image–caption examples with target hex colors.



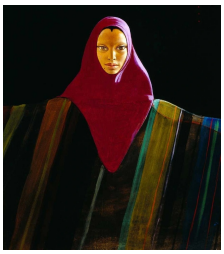
A front-facing photograph of a fluffy cat at the center. Its fur is mainly #DAC0A6 with #BE9263 markings. It has large bright green eyes, a pink nose, and white whiskers. A small #E0E0E1 flower rests near the top of its head. At the center, the cat lies in grass among many small white flowers, against a blurred natural background mixing green and warm tones. A soft, fresh atmosphere fills the scene.



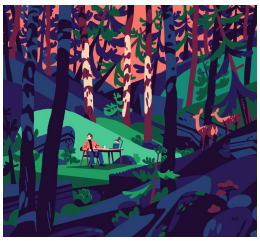
A medium close-up of an East Asian toddler with small braids at the center, looking curiously at the camera with round eyes. A softly colored painting hangs high on the #DCD6DC wall behind her, and she wears a yellow dress with small floral and cartoon bear patterns. Its broad white collar has ruffled edges and includes #C2AFA3 and #F9F1EF in the foreground.



A surrealist, ultra-wide-angle close-up in cool blue-gray tones. A woman's portrait occupies the right side, with black eyes and curly brown hair in #211E1A and #090807 covering much of that side. On the left, clouds cross a sky in #2C3B4C and #21557E. Blurred city silhouettes appear in the distant lower left, creating a mysterious atmosphere.

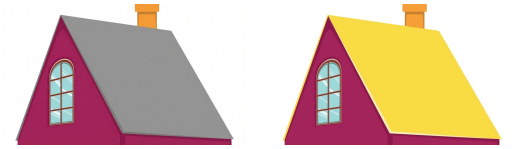


An oil painting with abstract expressionist and vintage influences. A woman appears in a centered, front-facing half-length portrait against a simple black #040A08 background in the upper frame, and she wears a magenta #680310 headscarf and clothing with intricate textures and colorful vertical stripes. A soft light, warm tones, and heavy yet gentle brushwork create a calm, mysterious expression and strong color contrast.



A flat forest illustration with vivid colors. At the top, the sky is #F77666, a dark #1F0C30 tree trunk stands on the left, and the green slope at the center is #4CC795. A person in an orange top sits at a table on the slope using a laptop. A pair of deer among the trees behind and to the right look toward the person. Dense trees in varied shades, low plants, and dark rocks surround the scene.

Figure 14: **T2I training-data examples (4/4).** Five image–caption examples with target hex colors.



Original Image #F9DB45

Change the roof to #F9DB45.



Original Image #28361B

Change the leaves in the basket to #28361B.



Original Image #E7E6EF

Change the cube to #E7E6EF.



Original Image #C58F94

Change the sweatshirt to #C58F94.



Original Image #81252B

Change the wallet to #81252B.



Original Image #34645F

Change the book cover to #34645F.



Original Image #CBABA4

Change the headboard to #CBABA4.



Original Image #655252

Change the skirt to #655252.



Original Image #ABC1B9

Change the plate to #ABC1B9.



Original Image #DB8548

Change the chest of drawers to #DB8548.

Figure 15: **Editing training-data examples (1/2).** Ten source–target pairs with short instructions. Each pair places the source on the left and the target on the right.



Figure 16: **Editing training-data examples (2/2).** Ten source–target pairs with short instructions. Each pair places the source on the left and the target on the right.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes] .

   Justification: The abstract and introduction state the intended scope: prompt-native 24-bit hex color control for generation and editing. The quantitative claims are supported by Table 1 and the ablations in Tables 2 and 3; limitations are discussed in Section 5.

   Guidelines:

   - The answer NA 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 NA 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: Section 5 states that the training data contain no palette-specific supervision. Appendix H notes that region averages are affected by shading, material and mask boundaries, and tests the metric under alternative estimators, SAM3 thresholds and localization failures. Appendix K states what the human study does not establish, and Appendix D states that the Paint-500K source images are not publicly available.

   Guidelines:

   - The answer NA 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: [NA] .

   Justification: The paper does not present theoretical theorems or formal proofs; it is an empirical study of precise hex-conditioned color generation and editing.

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes] .

   Justification: The method, data construction, training mixture, optimizer settings, benchmark construction, scoring protocol, prompt templates, and evaluator settings are described in Sections 3.3–4 and Appendices D–G.

   Guidelines:

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

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [No] .

   Justification: The paper describes the data construction, training recipe and evaluation protocols in detail (Section 3 and Appendices D–G), but no code or data are released with the paper. The Paint-500K source images come from an internal collection that is not publicly available.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.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://nips.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, etc.) necessary to understand the results?

   Answer: [Yes] .

   Justification: Section 4 specifies the evaluated models, metrics, training hyperparameters, and inference settings; Appendix D gives dataset and mixture details, Appendix F gives prompt templates, and Appendix G gives evaluator settings.

   Guidelines:

   - The answer NA 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: Scores are computed from one generated image per prompt over 1000 ACBench-T2I and 500 ACBench-Edit prompts, without error bars. Appendix H instead tests whether the conclusions depend on the region-color estimator, the SAM3 threshold and coverage gates, or localization failures, and Appendix K reports the human study descriptively.

Guidelines:

- The answer NA 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: Section 4 reports the number of GPUs, global batch size, optimizer, learning rate and training steps, but the paper does not report the GPU type, wall-clock time or total project compute.

   Guidelines:

   - The answer NA 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: The work uses existing pretrained models and image data processing pipelines for a generative modeling study, and we do not identify any deviation from the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA 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: [No] .

    Justification: The paper focuses on the technical problem of precise color control and discusses technical limitations, but it does not include a dedicated broader-impact discussion of possible positive and negative societal uses of improved image generation and editing.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA 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., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA] .

    Justification: The paper does not release a model checkpoint or dataset.

    Guidelines:

    - The answer NA 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: Existing models, benchmarks, and tools used by the paper are cited where introduced, including FLUX, Qwen-Image, Z-Image, SAM3, CompColor, and the VLM/editing tools used in the data pipeline.

Guidelines:

- The answer NA 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: The paper introduces Paint-500K and ACBench and documents their construction, fields, benchmark splits, evaluation protocol, prompt templates and evaluator settings in Sections 3.3 and 4 and Appendices B, D, F and G. The assets are documented but not released with the paper.

Guidelines:

- The answer NA 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: [No] .

Justification: Appendix K describes the human preference study (15 participants, 80 ACBench-T2I prompts, a target swatch beside two anonymized images per trial), but the paper does not include the full instruction text, screenshots or compensation details.

Guidelines:

- The answer NA 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: [No] .

    Justification: Participants in the preference study in Appendix K only compare generated images against target color swatches. The paper does not describe participant risks, their disclosure, or IRB review.

    Guidelines:

    - The answer NA 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 rigorousness, or originality of the research, declaration is not required.

    Answer: [Yes] .

    Justification: The paper describes the use of VLM/LLM-based components in the data pipeline, including instance grounding, caption rewriting, filtering, and recaptioning; these uses are part of dataset construction rather than unreported writing assistance.

    Guidelines:

    - The answer NA 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 (https://neurips.cc/Conferences/2026/MainTrackHandbook) for what should or should not be described.