PaletteAgent: Self-Evolving Correction for Line-Art Colorization

Anonymous ACL submission

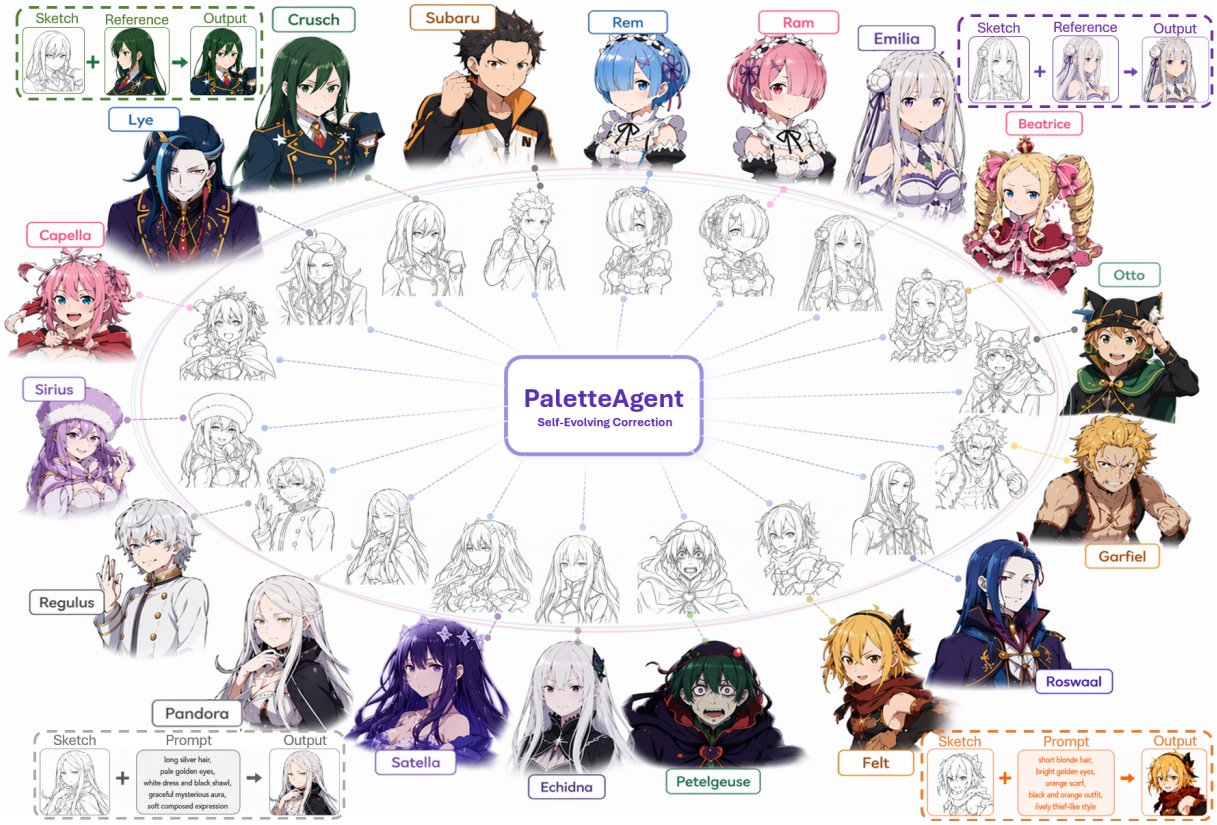

Figure 1: **Line art colorization results.** PaletteAgent enables prompt-adaptive line-art colorization from flexible inputs. Given only a sketch with an initial text prompt or a sketch with a reference image, PaletteAgent automatically infers structured color intent, refines prompts through self-evolving correction, and produces consistent colorization results while preserving the original line-art structure.

Abstract

Given a line-art sketch and user-specified color intent, controllable colorization must assign intended colors to fine-grained regions while preserving the underlying line structure. Existing methods have improved visual quality, but precise local adjustment remains difficult. In this work, we argue that the key limitation is the absence of a persistent correction state. Such a state records user intent, localized mismatches, grounded edit regions, and preservation constraints to guide targeted repairs. To address this, we introduce PaletteAgent, an agentic, self-evolving correction framework for controllable line-art colorization. PaletteAgent is built around a persistent correction state that tracks user intent, unresolved mismatches, editable regions, and preserved fields. This state enables the agent to perform structured prompt refinement, judgment-guided diagnosis, grounded local repair, and adaptive stopping. This iterative process motivates a GRPO training objective for optimizing the agent's correction policy. Empirical results show that PaletteAgent outperforms line-art colorization and general image-editing baselines. Further ablations confirm the role of maintaining a persistent correction state for controllable colorization.

1 Introduction

Line-art colorization converts black-and-white sketches into colored illustrations, aiming to accurately reflect user-specified colors while preserving the original drawing structure. Modern generative systems can already produce strong colorizations from concise textual prompts, visual references, or multimodal inputs (Meng et al., 2025; Liu et al., 2025; Qiu et al., 2025; Zhang et al., 2026; Wu et al., 2025). However, local color errors remain a per-

sistent issue. As shown in Figure 2, even capable systems may misrender key details, such as assigning a mismatched hair color or overlooking specific clothing features, even when intended colors are specified by user instructions or reference images.

Recent approaches have focused on improving alignment between generated colorizations and user intent. For instance, AniDoc (Meng et al., 2025) improves reference-guided colorization under pose and scale variation; MangaNinja (Liu et al., 2025) enhances accuracy in reference-based color transfer; and OmniColor (Zhang et al., 2026) integrates diverse spatial and semantic guidance signals. Despite these advances, precisely correcting residual local color errors without altering previously accurate regions remains difficult. Even a straightforward instruction such as "make the hair silver" can be ambiguous, as it does not clarify exactly which areas require changes or should remain untouched. Consequently, corrective edits may inadvertently affect unrelated image regions, compromising previously correct areas.

In this paper, we argue that addressing this challenge requires maintaining an ***explicit correction state***. To understand this intuitively, consider the workflow of a human artist: upon noticing a coloring mistake, the artist identifies the problematic region, preserves the correct parts, and adjusts only the specific error. Inspired by this process, we introduce *PaletteAgent*, a stateful self-evolving framework designed around such a correction state. The state tracks user instructions, previously identified errors, regions needing adjustment, and regions that should be preserved. As illustrated in Figure 1, explicitly managing this information helps the system target local corrections while reducing unintended changes to correctly colored regions.

Operationally, our framework iteratively (1) diagnoses remaining mismatches between the current rendering and user intent, (2) localizes corresponding editable regions, (3) explicitly marks regions that must be preserved, and (4) refines the rendering accordingly. To optimize decisions within this iterative process, such as when to stop refining, which regions to prioritize, and how aggressively to apply corrections, we cast these decisions as a reinforcement learning problem. We train a correction policy on structured feedback from each iteration, aiming to improve local accuracy while preserving already-correct regions.

Empirically, extensive experiments show that PaletteAgent outperforms existing line-art colorization methods and general image-editing baselines. Ablation studies further validate the central role of explicitly maintaining a persistent correction state, targeted region grounding, and iterative refinement. Our primary contributions include:

Figure 2: Localized intent alignment in line-art colorization. Highlighted regions show mismatches in color, identity cues, accessories, or background details. PaletteAgent tracks these issues across rounds for targeted edits while preserving matched regions.

- We identify the lack of a persistent correction context as a key limitation preventing reliable, precise, and targeted local corrections during controllable line-art colorization.
- We propose PaletteAgent, an agentic iterative refinement framework built around a persistent correction state that maintains user instructions, mismatches, editable regions, and preservation constraints across correction rounds.
- We introduce a reinforcement-learning-based optimization approach that uses structured feedback from each correction round to guide precise, context-aware correction decisions.

2 Related Work

Controllable colorization and image editors. Line-art and anime colorization colors line art from user descriptions, references, or multimodal hints while preserving structure. Prior work includes style-transfer, user-guided, diffusion-control, reference-transfer, and animation-oriented systems (Zhang et al., 2017, 2018a; He et al., 2018; Zhang et al., 2023; Meng et al., 2025). Recent methods align references, color hints, text prompts, identity cues, and temporal history with the renderer (Liu et al., 2025; Qiu et al., 2025; Zhang et al., 2026). Large multimodal editors offer strong

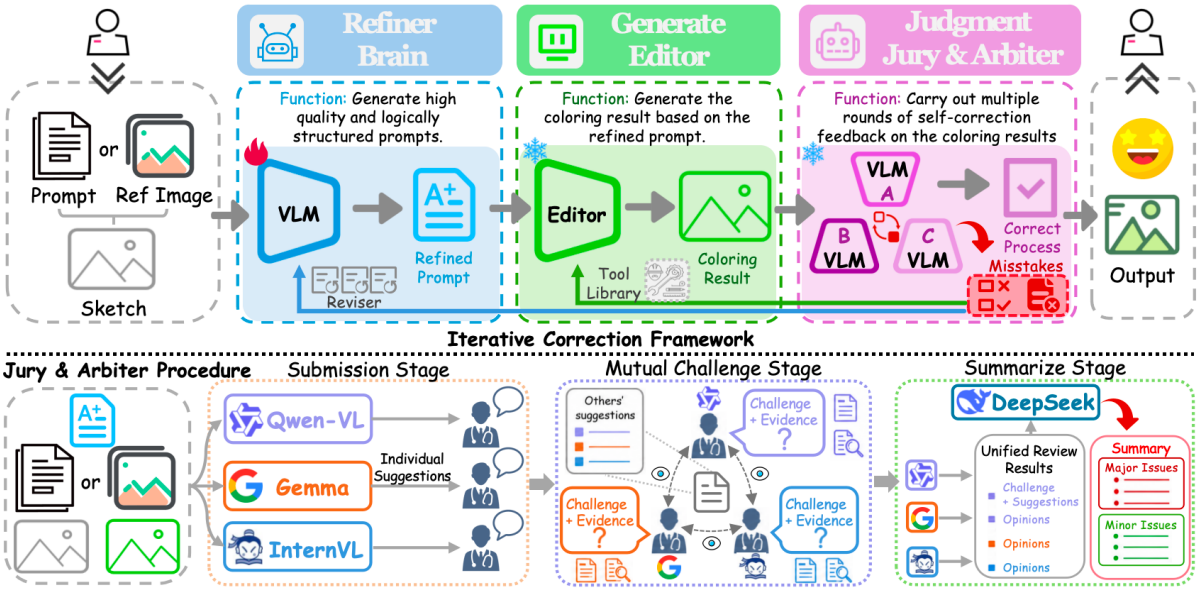

Figure 3: PaletteAgent correction framework: *Brain* converts intent and feedback into structured prompt actions, *Editor* renders a round image, *Jury* produces localized findings, *Arbiter* summarizes into major/minor/correct fields. Brain selects next correction: prompt refinement, region grounding, preservation, or final render.

instruction following and visual synthesis as single-pass renderers (Wu et al., 2025). These improve intent conditioning; PaletteAgent focuses on post-render correction via persistent tracking of mismatches, protected regions, and repair actions.

Agentic refinement and actionable feedback. Language-agent systems enable reasoning over feedback, tool calling, and action revision (Yao et al., 2023; Schick et al., 2023; Madaan et al., 2023; Shinn et al., 2023); visual systems extend this to image generation/editing via agentic policies, prompt coordination, or perception-reasoning-action loops (Jiang et al., 2026; Chen et al., 2025; Shen et al., 2026). Model-as-judge evaluation, AI feedback, and debate support scalable feedback aggregation (Zheng et al., 2023; Bai et al., 2022; Du et al., 2023), while edit-evaluation benchmarks structure visual-edit assessment (Ku et al., 2024; Yosef et al., 2025). PaletteAgent applies this paradigm to line-art colorization, organizing Jury-Arbiter findings into major mismatches, minor issues, correct fields updating correction state.

Grounded repair and policy optimization. Localized correction connects visual-language feedback to editable image regions. Open-vocabulary grounding and segmentation tools provide this link: Florence-2 for detection/grounding (Xiao et al., 2024), and SAM/SAM2 for converting region prompts into masks (Kirillov et al., 2023; Ravi et al., 2024). Policy optimization then decides the correction loop's actions across rounds. Prior methods include RLHF (Ouyang et al., 2022), DPO (Rafailov et al., 2023), PPO (Schulman et al., 2017), and GRPO (Shao et al., 2024). PaletteAgent combines grounded repair with GRPO-based Brain policy optimization, deciding at each round when to refine, localize, repair, or stop while carrying verified fields forward as explicit keep constraints.

3 PaletteAgent: The Proposed Method

Existing controllable colorization methods have improved substantially, yet precise fine-grained local adjustment remains challenging. In this paper, we identify the lack of explicit correction context as the core limitation. To address this, we propose PaletteAgent, a stateful self-evolving correction framework for controllable line-art colorization.

3.1 Persistent Correction State Definition

To formalize this framework, we first need a principled way to maintain correction state across rounds. Recall that when a human artist corrects a partially colored sketch, if the hair color is wrong but the eyes and clothing are already correct, the artist does not restart the whole image or edit every region equally. Instead, they track regions needing adjustments and those that should remain unchanged. Analogously, effective line-art colorization also requires a structured memory of regions to edit or preserve. Inspired by this, we explicitly define a

persistent correction state at iteration t:

$$s_t = \{u, r, x, p_t, y_t, F_{<t}, \mathcal{M}_{<t}, \Omega_{<t}, L_t\},$$

Here, u, r, and x denote the user-specified color description, optional reference image, and original line-art sketch, respectively. The pair (p_t, y_t) represents the structured prompt and rendered image at iteration t. The remaining terms store correction memory: $F_{<t}$ contains structured feedback from previous correction steps, $\mathcal{M}_{<t}$ records previous mismatches, $\Omega_{<t}$ stores the corresponding editable-region masks, and L_t gives preservation labels for regions to keep unchanged.

This state serves two roles. First, it disambiguates unresolved vs. verified regions. If a policy observes only current input, prompt, and rendering, it may skip unresolved regions or re-edit verified ones. Proposition 1 formalizes this ambiguity and quantifies its decision cost (proof in Appendix A):

PROPOSITION 1 ***(State disambiguates edit-or-preserve decisions)*** *Let $o_t = (u, r, x, p_t, y_t)$. Consider two states with the same o_t: one with verified j ($j \in L_t$), the other unresolved ($F_{<t}$ or $\mathcal{M}_{<t}$). Skipping unresolved costs α, editing verified costs β. Any policy with only o_t cannot distinguish them, so its worst-case loss is at least*

$$\begin{cases} \frac{\alpha\beta}{\alpha+\beta}, & \textit{randomized}, \\ \min(\alpha, \beta), & \textit{deterministic}. \end{cases}$$

Conditioning on s_t resolves this ambiguity by revealing whether j should be edited or preserved.

Second, once a mismatch is identified, protected regions must stay unchanged. The correction state provides preservation labels to choose a mask covering the mismatch while avoiding nearby protected regions. Proposition 2 captures this constraint (proof in Appendix A):

PROPOSITION 2 ***(Preservation labels constrain safe masks).*** *Let $\mathcal{M}_t$ be the mismatch support and L_t the preserved regions. For any region set A, let $\mathcal{N}_w(A) = \{z : \mathrm{dist}(z, A) \le w\}$ denote its w-neighborhood. Define*

$$\mathcal{S}_t = \mathcal{M}_t \setminus \mathcal{N}_w(L_t), \qquad \mathcal{C}_t = \mathcal{M}_t \cap \mathcal{N}_w(L_t).$$

For any editable mask Ω, let $E_t(\Omega) = |\mathcal{N}_w(\Omega) \cap L_t|$ be preservation risk (neighborhood overlap with preserved regions). Then

$$\Omega \subseteq \mathcal{S}_t \Rightarrow E_t(\Omega) = 0,$$
$$\Omega \cap \mathcal{C}_t \neq \emptyset \Rightarrow E_t(\Omega) > 0.$$

Thus, choosing a safe mask requires both mismatch support $\mathcal{M}_t$ and preservation labels L_t.

3.2 Practical Correction-State Tracking

The persistent correction state defined in Section 3.1 specifies the necessary context to carry across correction rounds. However, updating this state in practice involves addressing several interrelated challenges: how to reliably identify newly introduced errors, locate the exact regions that require editing, preserve previously verified areas, and decide which action to take next.

To address these challenges simultaneously, we structure each correction round as an agentic correction framework with four specialized roles: *Brain*, *Editor*, *Jury*, and *Arbiter*, as shown in Figure 3. These roles work together to translate visual feedback into actionable state updates. **Each correction round proceeds through four stages:**

1) Mismatch Diagnosis. Each correction round starts by checking the current rendering y_t against the user instruction u and optional reference image r. Because errors may involve global palette drift, local attribute mistakes, reference mismatch, or structure damage, we use multiple Jury models rather than a single evaluator. The judgment protocol has three steps. First, each Jury model independently reports issues. Next, the Jury models cross-check these reports against visual evidence. Finally, the Arbiter summarizes the verified evidence into structured feedback f_t with three fields: major mismatches, minor issues, and verified fields. Major mismatches form the mismatch set $\mathcal{M}_t$. Verified fields update the preservation labels L_{t+1}, which become keep constraints in the next prompt.

2) Brain Action Selection. The Brain then chooses the next route conditioned on the persistent state s_t and current feedback f_t. The state supplies the history: previous feedback, previous mismatches, existing masks, and current preservation labels. To choose the appropriate correction scope at each round, we define the following action space:

$$\mathcal{A} = \{\mathrm{refine}_{\mathrm{global}}, \mathrm{refine}_{\mathrm{local}}(m), \mathrm{call}_{\mathrm{tool}}(m), \mathrm{stop}\},$$

where $m \in \mathcal{M}_t$ denotes a target mismatch. If no

major mismatch remains, the Brain selects stop and sends the current prompt to the Final Editor. Otherwise, major mismatches are routed to local refinement, with $\mathrm{call}_{\mathrm{tool}}(m)$ used when the target region lacks a usable mask. Minor issues are routed to global refinement.

3) Grounded Refinement. For local refinement, the Brain needs an editable region. If no usable mask is already available, open-vocabulary grounding tools (*e.g.*, Florence-2 and SAM2) translate the region cue for m, such as an incorrect hair or clothing color, into a mask. The new masks Ω_t extend the mask history $\Omega_{\leq t} = \Omega_{<t} \cup \Omega_t$. The Brain then updates the structured prompt from p_t to p_{t+1} using the selected mismatch, available masks, and keep constraints from L_{t+1}. For global refinement, the Brain updates the prompt from minor feedback without a new mask. The Iterative Editor renders y_{t+1} from the original line art x, optional reference r, and updated prompt p_{t+1}.

4) State Update. After rendering, the new feedback and mismatch records are committed to the correction state for subsequent rounds:

$$F_{\leq t} = F_{<t} \cup \{f_t\}, \qquad \mathcal{M}_{\leq t} = \mathcal{M}_{<t} \cup \mathcal{M}_t,$$

yielding the next correction state:

$$s_{t+1} = \{u, r, x, p_{t+1}, y_{t+1}, F_{\leq t}, \mathcal{M}_{\leq t}, \Omega_{\leq t}, L_{t+1}\}.$$

3.3 Iterative Correction Procedure

Building on the persistent correction state and defined roles, we now describe the practical iterative refinement procedure. The goal is to progressively reduce mismatches while preserving correct regions and applying targeted, state-aware edits.

Given a user description u, the line-art sketch x, and an optional reference image r, inference begins by generating an initial structured prompt and rendering it with the Iterative Editor (*e.g.*, Qwen-Image-Edit). If the description is sparse or the reference image needs to be converted into explicit color fields, an optional Describer expands the input into $\tilde{u}$; otherwise, $\tilde{u} = u$:

$$p_0 = \mathrm{Brain.rewrite}(\tilde{u}, r),$$
$$y_0 = \mathrm{Editor}_{\mathrm{iter}}(x, p_0, r).$$

The initial rendering y_0 establishes the baseline for correction. In later rounds, the system identifies mismatches, grounds editable regions, updates preservation labels, and lets the same Iterative Editor render each updated prompt. When no major mismatches remain, the Brain selects stop and ends the loop; the loop also stops when the maximum iteration count T is reached, balancing correction precision and computational cost.

Empirically, convergence occurs within 2–4 iterations, providing a practical balance between efficiency and correction precision. Upon convergence, the finalized structured prompt, the original line art, and provided reference image are passed to the Final Editor (*e.g.*, NanoBanana/Gemini Image) to generate the final colorization. This separation keeps the prompt-responsive, low-cost editor in the loop and reserves the high-fidelity renderer for output quality and detail preservation.

3.4 Optimizing Brain Decision via Reinforcement Learning

Given the correction state, action space, and iterative loop, the remaining question is: how should the Brain choose proper actions across correction rounds? Although the Brain receives informative signals from the current correction state, including feedback, mismatch sets, editable masks, and preservation labels, *these signals do not directly specify which action will best improve the final outcome*. A local repair may resolve an immediate mismatch but risk changing a verified region, while invoking grounding tools may add cost yet enable a more precise edit. The Brain must also decide when further refinement is unlikely to help.

Fortunately, these trade-offs can be formulated as a sequential decision-making problem, which can then be naturally optimized with classic reinforcement learning (Sutton et al., 1998). We treat the Brain as a policy interacting with an environment formed by the remaining components: the Editor, Jury, Arbiter, and grounding tools. At each correction round, the policy observes the correction state s_t and selects a structured action from $\mathcal{A}$, while the other components remain fixed. Under this formulation, we optimize the Brain with GRPO (Shao et al., 2024), using a reward function composed of three supervision signals:

$$R(\tau) = \underbrace{\lambda_o R_o}_{\text{outcome}} + \underbrace{\lambda_p R_p}_{\text{process}} + \underbrace{\lambda_\Delta R_\Delta}_{\text{improvement}} .$$

Here, R_o evaluates the final rendering against the user intent, optional reference image, and final feedback. R_p rewards valid intermediate behavior, including well-formed structured prompts, coverage

Method	Image-Align ↑	SSIM ↑	PSNR ↑	FID ↓	LPIPS ↓	DINO ↑
Colorization baselines						
ControlNet	0.633	0.316	7.83	138.62	0.779	0.699
Tag2pix	0.660	0.444	7.55	174.93	0.606	0.727
MagicColor	0.784	0.556	10.81	121.04	0.511	0.795
ColorDiffusionXL	0.832	0.620	14.05	207.38	0.436	0.814
AnimeColor	0.845	0.625	13.78	225.12	0.420	0.813
General image editors						
DreamOmni2	0.721	0.412	8.88	213.41	0.749	0.713
Flux2-dev	0.813	0.450	10.57	189.30	0.553	0.769
Seedream 5	0.835	0.616	11.04	175.63	0.612	0.785
GPT-5.4	0.849	0.521	11.78	157.92	0.536	0.796
Qwen-Image-Edit	0.855	0.518	12.03	145.76	0.477	0.807
Kling	0.859	0.574	12.42	132.54	0.488	0.816
NanoBanana	0.873	0.604	13.66	119.86	0.433	0.837
Closest line-art colorization baselines and ours						
AniDoc	0.846	0.674	16.91	58.92	0.343	0.879
MangaNinja	0.910	0.683	16.82	48.07	0.322	0.894
LVCD	0.889	0.833	22.05	35.42	0.275	0.917
PaletteAgent	**0.979**	**0.957**	**25.26**	**22.16**	**0.057**	**0.955**

Table 1: Anime line-art colorization results. Higher is better for Image-Align, SSIM, PSNR, DINO; lower for FID, LPIPS. Rows grouped by method family; shading indicates rank bands per metric.

Variant	Align ↑	SSIM ↑	PSNR ↑	FID ↓	LPIPS ↓	DINO ↑
w/o Iteration	0.895	0.772	16.62	55.84	0.238	0.885
w/o Tools	0.923	0.840	17.71	43.92	0.162	0.912
w/o Preservation State	0.931	0.858	18.02	40.15	0.145	0.918
w/o Jury-Arbiter	0.939	0.876	18.31	36.68	0.126	0.924
w/o Agentic RL	0.948	0.904	18.68	31.42	0.101	0.930
Full Model	**0.979**	**0.957**	**20.14**	**22.16**	**0.057**	**0.955**

Table 2: Correction loop ablation. Removing iteration, grounding tools, preservation state, Jury-Arbiter feedback, or policy training reduces image alignment, perceptual quality, and semantic consistency.

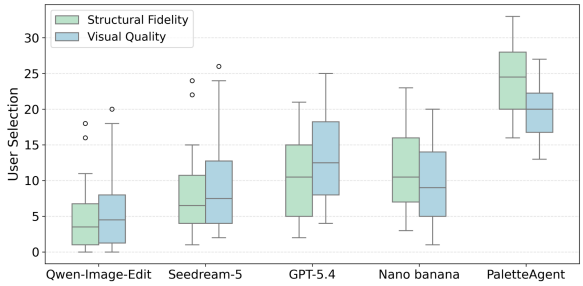

Figure 4: Human preference results for structural fidelity and visual quality. Participants choose among anonymized outputs from five systems; per-sample counts are provided in Appendix Table 6.

of requested attributes, appropriate grounding-tool use, and efficient stopping. R_Δ measures whether each round improves the correction state, rewarding reductions in major and minor mismatches, successful repair of selected regions, and avoiding drift on already-correct fields. ***More details about the reward computation are provided in Appendix D.***

4 Experiments

Benchmark. We train the Brain policy on Sakuga-42M (Pan, 2024) for correction rollouts, with the Editor, Jury, Arbiter, and grounding tools fixed. Evaluation uses a benchmark pairing anime line art with user color descriptions, ground-truth images, and optional references. Examples are from six anime: *Frieren*, *Spy×Family*, *Classroom of the Elite*, *Cyberpunk Edgerunners*, *Odd Taxi*, and *Wandering Witch*. Each example includes a user description, line art, ground-truth image, and an optional reference. The ground-truth image is held out from generation, used only for metrics.

Baselines. We compare PaletteAgent with fifteen baselines across task-specific colorization, diffusion-based colorization, and large image editors: ControlNet, Tag2Pix, MagicColor, ColorDiffusionXL, AnimeColor, AniDoc, MangaNinja, LVCD, DreamOmni2, Qwen-Image-Edit, Flux2-dev, Seedream 5, GPT-5.4, Kling, and NanoBanana/Gemini Image (Zhang et al., 2023; Kim et al., 2019; Zhang et al., 2025a; Yan et al., 2026; Zhang et al., 2025b; Meng et al., 2025; Liu et al., 2025; Huang et al., 2024; Xia et al., 2025; Wu et al., 2025). All methods receive the same line art and user color intent; those supporting image conditioning also receive same optional reference, with reference-conditioned variants reported separately. Editor baselines are single-pass; PaletteAgent uses

Figure 5: Qualitative ablation examples. The panels show how iteration, region grounding, and policy training affect local attribute repair and preservation of surrounding regions.

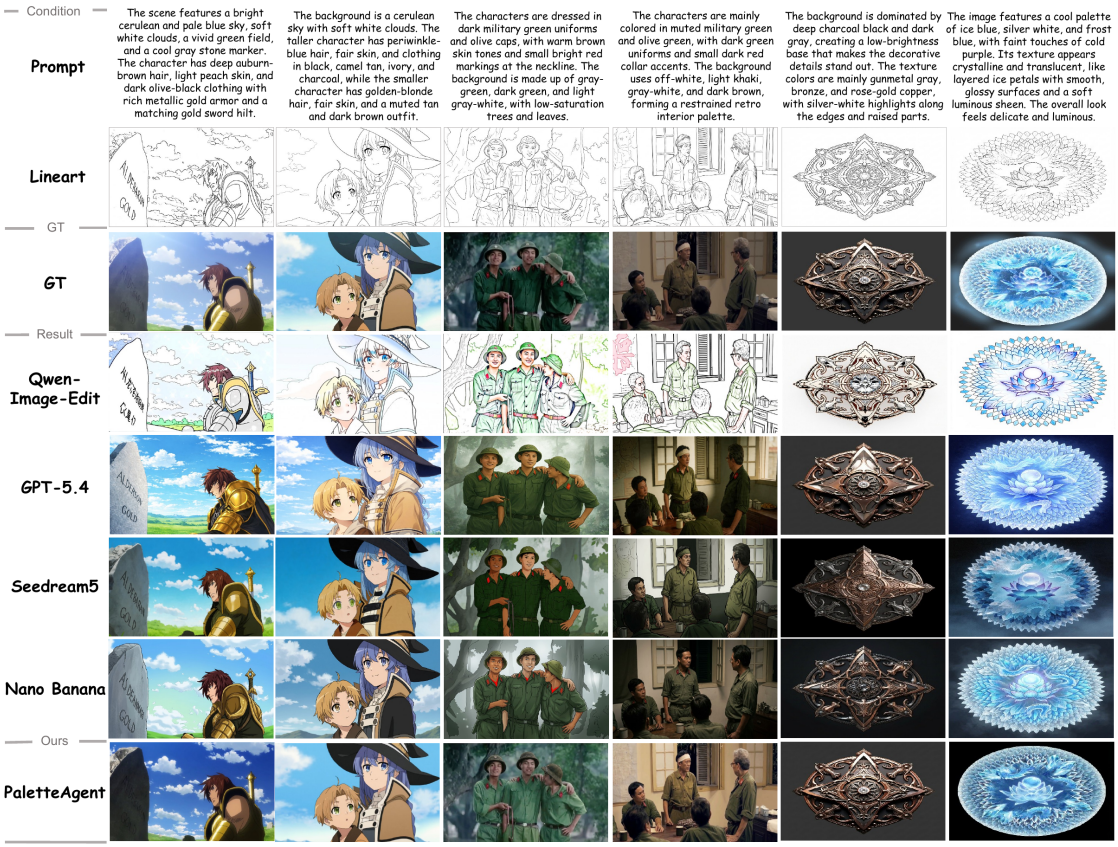

Figure 6: Prompt-following qualitative examples under language-only color requests. The examples cover object-level attributes, multi-region edits, and non-character decorative elements.

the bounded correction loop (Section 3).

Metrics. We report Image-Align, SSIM, PSNR, FID, LPIPS, and DINO (Table 1). Image-Align (Zhang et al., 2026) measures target alignment. SSIM/PSNR measure structural and reconstruction fidelity. FID/LPIPS measure distributional and perceptual distance (lower is better) (Heusel et al., 2017; Zhang et al., 2018b). DINO measures semantic similarity (Caron et al., 2021). Together, these cover line preservation, perceptual quality, and semantic consistency.

Human Preference Study. We also summarize a five-way human preference study with 40 participants. Participants view anonymized outputs and choose the result they prefer under two criteria: structural fidelity and visual quality. The study provides a perceptual comparison under the same structural and visual criteria used in our analysis.

4.1 Quantitative Results

Table 1 reports final-image performance across methods. PaletteAgent achieves the best Image-

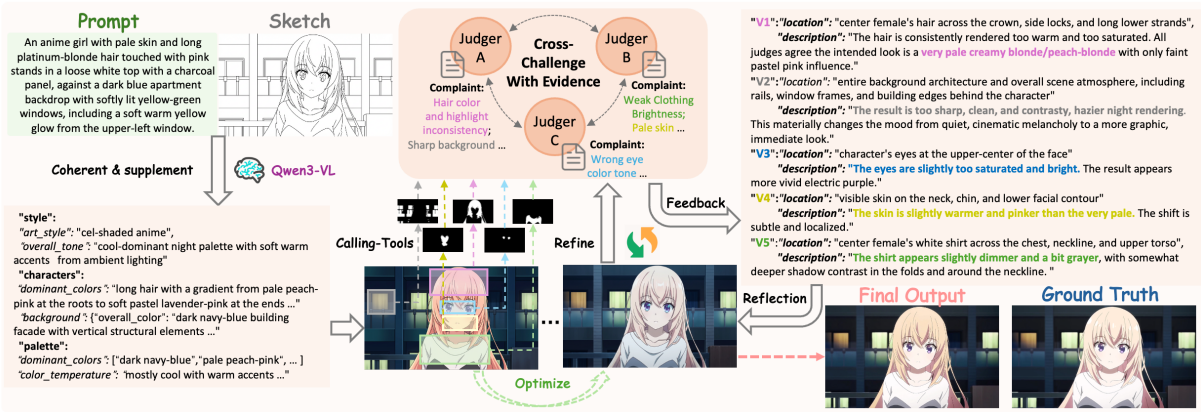

Figure 7: A sample correction trajectory. PaletteAgent converts the user intent into structured intent fields and performs self-evolving iterations. The jury identifies mismatched attributes, localizes the affected regions, and edits them while preserving already matched content to achieve the optimal result.

Align, SSIM, FID, LPIPS, and DINO among evaluated systems, while reaching comparable PSNR to the strongest result and outperforming the remaining methods on this metric. These results indicate stronger target alignment, line-art preservation, perceptual quality, and semantic consistency. Figure 4 provides further preference evidence: participants choose PaletteAgent most often under both structural fidelity and visual quality.

4.2 Ablation Study

Table 2 ablates key components of our feedback-driven repair loop. Removing iteration, which prevents later corrective rounds, causes the largest drop, showing that sequential corrective actions are essential. Removing grounding tools or preservation state also degrades performance, confirming the importance of region-specific repair and explicit keep constraints. The full model outperforms both an untrained policy and the variant without Jury-Arbiter consolidation, indicating that learned action selection and verified feedback improve local editing, preservation, and stopping decisions. Table 3 and Appendix Table 5 further show that multi-Jury verification and the full reward design both aid stronger correction and preservation.

4.3 Qualitative Analysis

We examine model behavior on language-only and reference-guided examples. Figure 6 (prompt-following) and Appendix Figure 8 (reference-guided) cover character attributes, multi-region color intent, background palettes, and decorative objects. PaletteAgent better satisfies local color requirements and transfers reference palettes while preserving sketch structure. These cases show the loop is most useful for local attribute errors fixable via constrained edits, while remaining errors stem from palette drift, missed local attributes, structural damage, or unintended edits to surroundings.

Judging setup	Align ↑	SSIM ↑	PSNR ↑	FID ↓	LPIPS ↓	DINO ↑
1 Jury model	0.945	0.892	18.44	34.90	0.113	0.927
2 Jury models	0.962	0.929	19.22	27.58	0.079	0.943
3 Jury models + Arbiter	**0.979**	**0.957**	**20.14**	**22.16**	**0.057**	**0.955**

Table 3: Effect of using multiple Jury models. Adding independent Jury models and an Arbiter to summarize their findings improves alignment, perceptual quality, and semantic consistency.

Round-Level Correction Trace. Figure 7 shows a representative correction trace. PaletteAgent expands the prompt into structured fields, uses judgment agents to detect mismatched attributes, grounds affected regions with masks, and refines the image under preservation constraints. The trace makes the intermediate state visible: each finding becomes an edit target or preservation constraint before the next image is produced. This shows how feedback on attributes (hair, eyes, skin, clothing, background) is converted into localized edits.

5 Conclusion

This paper presented PaletteAgent, an iterative agentic framework for controllable line-art colorization. By maintaining correction state over user intent, localized mismatches, grounded regions, and preserved fields, PaletteAgent turns feedback into targeted edits. Experiments show gains in intent alignment, local repair, and line-art preservation; ablations confirm the roles of iteration, grounding, and policy training.

Limitations

PaletteAgent is evaluated on controllable line-art colorization, where user intent is naturally expressed through local attributes, reference colors, and preservation constraints. Broader visual editing domains may require domain-specific prompts or evaluation protocols. The iterative loop also uses more model calls than a single-pass editor, reflecting the intended trade-off between generation speed and precise controllability.

Ethical considerations

PaletteAgent is intended to support creative colorization workflows. Users should have appropriate rights to the line art and reference images used with the system. The human preference study uses anonymized outputs and reports only aggregate judgments under randomized presentation order.

References

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, and 1 others. 2025. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, and 32 others. 2022. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. 2021. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9650–9660.

Chieh-Yun Chen, Min Shi, Gong Zhang, and Humphrey Shi. 2025. T2i-copilot: A training-free multi-agent text-to-image system for enhanced prompt interpretation and interactive generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19396–19405.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. 2023. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Mingming He, Dongdong Chen, Jing Liao, Pedro V Sander, and Lu Yuan. 2018. Deep exemplar-based colorization. *ACM Transactions on Graphics (TOG)*, 37(4):1–16.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*.

Zhitong Huang, Mohan Zhang, and Jing Liao. 2024. Lvcd: reference-based lineart video colorization with diffusion models. *ACM Transactions on Graphics (TOG)*, 43(6):1–11.

Kaixun Jiang, Yuzheng Wang, Junjie Zhou, Pandeng Li, Zhihang Liu, Chen-Wei Xie, Zhaoyu Chen, Yun Zheng, and Wenqiang Zhang. 2026. Genagent: Scaling text-to-image generation via agentic multimodal reasoning. *arXiv preprint arXiv:2601.18543*.

Hyunsu Kim, Ho Young Jhoo, Eunhyeok Park, and Sungjoo Yoo. 2019. Tag2pix: Line art colorization using text tag with secat and changing loss. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9056–9065.

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. 2023. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026.

Max Ku, Dongfu Jiang, Cong Wei, Xiang Yue, and Wenhu Chen. 2024. Viescore: Towards explainable metrics for conditional image synthesis evaluation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12268–12290.

Zhiheng Liu, Ka Leong Cheng, Xi Chen, Jie Xiao, Hao Ouyang, Kai Zhu, Yu Liu, Yujun Shen, Qifeng Chen, and Ping Luo. 2025. Manganinja: Line art colorization with precise reference following. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5666–5677.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*.

Yihao Meng, Hao Ouyang, Hanlin Wang, Qiuyu Wang, Wen Wang, Ka Leong Cheng, Zhiheng Liu, Yujun Shen, and Huamin Qu. 2025. Anidoc: Animation creation made easier. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18187–18197.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Zhenglin Pan. 2024. Sakuga-42m dataset: Scaling up cartoon research. *arXiv preprint arXiv:2405.07425*.

Qianru Qiu, Jiafeng Mao, Kento Masui, and Xueting Wang. 2025. Mangadit: Reference-guided line art colorization with hierarchical attention in diffusion transformers. *arXiv preprint arXiv:2508.09709*.

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

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. 2024. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*.

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

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

Shaocheng Shen, Jianfeng Liang, Chunlei Cai, Cong Geng, Huiyu Duan, Xiaoyun Zhang, Qiang Hu, and Guangtao Zhai. 2026. Agentic retoucher for text-to-image generation. *arXiv preprint arXiv:2601.02046*.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*.

Richard S Sutton, Andrew G Barto, and 1 others. 1998. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, and 1 others. 2025. Internvl3. 5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, and 20 others. 2025. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*.

Bin Xia, Bohao Peng, Yuechen Zhang, Junjia Huang, Jiyang Liu, Jingyao Li, Haoru Tan, Sitong Wu, Chengyao Wang, Yitong Wang, and 1 others. 2025. Dreamomni2: Multimodal instruction-based editing and generation. *arXiv preprint arXiv:2510.06679*.

Bin Xiao, Haiping Wu, Weijian Xu, Xiyang Dai, Houdong Hu, Yumao Lu, Michael Zeng, Ce Liu, and Lu Yuan. 2024. Florence-2: Advancing a unified representation for a variety of vision tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4818–4829.

Dingkun Yan, Xinrui Wang, Ru Wang, Zhuoru Li, Jinze Yu, Yusuke Iwasawa, Yutaka Matsuo, and Jiaxian Guo. 2026. Towards high-resolution and disentangled reference-based sketch colorization. *arXiv preprint arXiv:2603.05971*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*.

Ron Yosef, Yonatan Bitton, Dani Lischinski, and Moran Yanuka. 2025. Editinspector: A benchmark for evaluation of text-guided image edits. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 29503–29530, Vienna, Austria. Association for Computational Linguistics.

Lvmin Zhang, Yi Ji, and Xin Lin. 2017. Style transfer for anime sketches with enhanced residual u-net and auxiliary classifier gan. *arXiv preprint arXiv:1706.03319*.

Lvmin Zhang, Chengze Li, Tien-Tsin Wong, Yi Ji, and Chunping Liu. 2018a. Two-stage sketch colorization. *ACM Transactions on Graphics (TOG)*, 37(6):1–14.

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

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. 2018b. The unreasonable effectiveness of deep features as a perceptual

metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–595.

Xulu Zhang, Haoqian Du, Xiaoyong Wei, and Qing Li. 2026. Omnicolor: A unified framework for multi-modal lineart colorization. *arXiv preprint arXiv:2603.27531*.

Yinhan Zhang, Yue Ma, Bingyuan Wang, Qifeng Chen, and Zeyu Wang. 2025a. Magiccolor: Multi-instance sketch colorization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15205–15217.

Yuhong Zhang, Liyao Wang, Han Wang, Danni Wu, Zuzeng Lin, Feng Wang, and Li Song. 2025b. Animecolor: Reference-based animation colorization with diffusion transformers. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pages 6682–6690.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623.

Appendix

Table of Contents

A Proofs and Derivations 13

B Implementation Details 14

B.1 Inputs, Actions, and Prompt Templates 14

C Full Judgment and Refinement Protocol 15

D Policy Training Details 16

E Evaluation Details 17

E.1 Benchmark Construction 17

E.2 Human Preference Study 17

E.3 Preprocessing 18

F Additional Qualitative Examples 18

G Extended Related Work 18

H Compute, Cost, and Reproducibility 19

I Full Prompt Templates 20

A Proofs and Derivations

Proposition 1 (Persistent state disambiguates edit-or-preserve decisions). Let $c_t = (u, r, x, p_t, y_t)$ collect the current task, prompt, and rendering. Consider two reachable correction states that share the same c_t but differ on a region j. In the first state, j has been verified as correct and belongs to L_t; in the second, prior evidence in $F_{<t}$ or $\mathcal{M}_{<t}$ marks j as unresolved. Let $\alpha, \beta > 0$ denote the unit costs of under-correction and over-correction, respectively. The loss on j is

$$\ell_j(S) = \begin{cases} \alpha, & j \notin S \text{ and } j \text{ is unresolved}, \\ \beta, & j \in S \text{ and } j \in L_t, \\ 0, & \text{otherwise}. \end{cases}$$

Proof. Since the two states share the same c_t, any policy conditioned only on c_t uses the same distribution over editing supports in both states. Let

$$p = \Pr_{S \sim \pi(c_t)}[j \in S].$$

If j is preserved, editing it incurs over-correction cost β; if j is unresolved, not editing it incurs under-correction cost α. Thus the expected losses on the two histories are

$$\mathbb{E}[\ell_j(S) \mid j \in L_t] = \beta p,$$
$$\mathbb{E}[\ell_j(S) \mid j \text{ unresolved}] = \alpha(1-p).$$

Therefore

$$\begin{aligned} &\max\big\{\mathbb{E}[\ell_j(S) \mid j \in L_t], \\ &\qquad \mathbb{E}[\ell_j(S) \mid j \text{ unresolved}]\big\} \\ &= \max\{\beta p, \alpha(1-p)\}. \end{aligned}$$

The minimum of $\max\{\beta p, \alpha(1-p)\}$ over $p \in [0,1]$ is reached when $\beta p = \alpha(1-p)$, giving $p = \alpha/(\alpha+\beta)$ and value $\alpha\beta/(\alpha+\beta)$. Hence every randomized current-only policy incurs worst-case expected aliasing loss at least $\alpha\beta/(\alpha+\beta)$ on region j. For a deterministic policy, $p \in \{0,1\}$, so the additional worst-case loss is α when $p = 0$ and β when $p = 1$, giving the lower bound $\min(\alpha, \beta)$.

A policy using the full correction state observes the distinguishing record: in the first history, $j \in L_t$ marks a verified region; in the second, prior evidence in $F_{<t}$ or $\mathcal{M}_{<t}$ marks j as unresolved and $j \notin L_t$. It can therefore preserve j in the first history and target j in the second. Thus the positive lower bound above is caused by aliasing in the current observation, not by the action space itself. □

Proposition 2 (Preservation labels constrain safe masks). Let $\mathcal{M}_t$ be the support of mismatched regions and L_t the support of preserved regions. For a neighborhood radius w, define

$$\mathcal{C}_t = \mathcal{M}_t \cap \mathcal{N}_w(L_t), \qquad \mathcal{S}_t = \mathcal{M}_t \setminus \mathcal{N}_w(L_t).$$

For a candidate mask Ω, define coverage and preservation risk as

$$C_t(\Omega) = |\Omega \cap \mathcal{M}_t|, \qquad E_t(\Omega) = |\mathcal{N}_w(\Omega) \cap L_t|.$$

Then

$$\Omega \subseteq \mathcal{S}_t \Rightarrow E_t(\Omega) = 0,$$
$$\Omega \cap \mathcal{C}_t \neq \emptyset \Rightarrow E_t(\Omega) > 0.$$

This coupling shows that mask selection must consider mismatch coverage and preservation risk jointly.

Proof. The proposition uses the preservation labels to split the current mismatch support into two subsets. The first subset, $\mathcal{S}_t$, contains mismatch locations that are farther than w from every region marked to remain unchanged:

$$\mathcal{S}_t = \mathcal{M}_t \setminus \mathcal{N}_w(L_t).$$

Thus, for any $z \in \mathcal{S}_t$ and any $q \in L_t$, we have $\mathrm{dist}(z, q) > w$. If an editable mask is restricted to this set, $\Omega \subseteq \mathcal{S}_t$, then the same distance condition holds for every $z \in \Omega$ and every $q \in L_t$. Therefore no preserved point lies inside the w-neighborhood of the mask:

$$\mathcal{N}_w(\Omega) \cap L_t = \emptyset,$$

so the preservation risk is $E_t(\Omega) = |\mathcal{N}_w(\Omega) \cap L_t| = 0$.

The second subset, $\mathcal{C}_t$, contains mismatch locations that lie within distance w of at least one preserved region:

$$\mathcal{C}_t = \mathcal{M}_t \cap \mathcal{N}_w(L_t),$$

so each point in $\mathcal{C}_t$ is a mismatch point whose nearby neighborhood touches a region marked to remain unchanged. Suppose a candidate mask overlaps this subset, $\Omega \cap \mathcal{C}_t \neq \emptyset$, and choose any $z \in \Omega \cap \mathcal{C}_t$. Because $z \in \mathcal{C}_t$, there exists a point $q \in L_t$ with $\mathrm{dist}(z, q) \leq w$. Since $z \in \Omega$, the same q also lies in $\mathcal{N}_w(\Omega) \cap L_t$. This intersection is therefore nonempty, which gives

$$E_t(\Omega) = |\mathcal{N}_w(\Omega) \cap L_t| > 0.$$

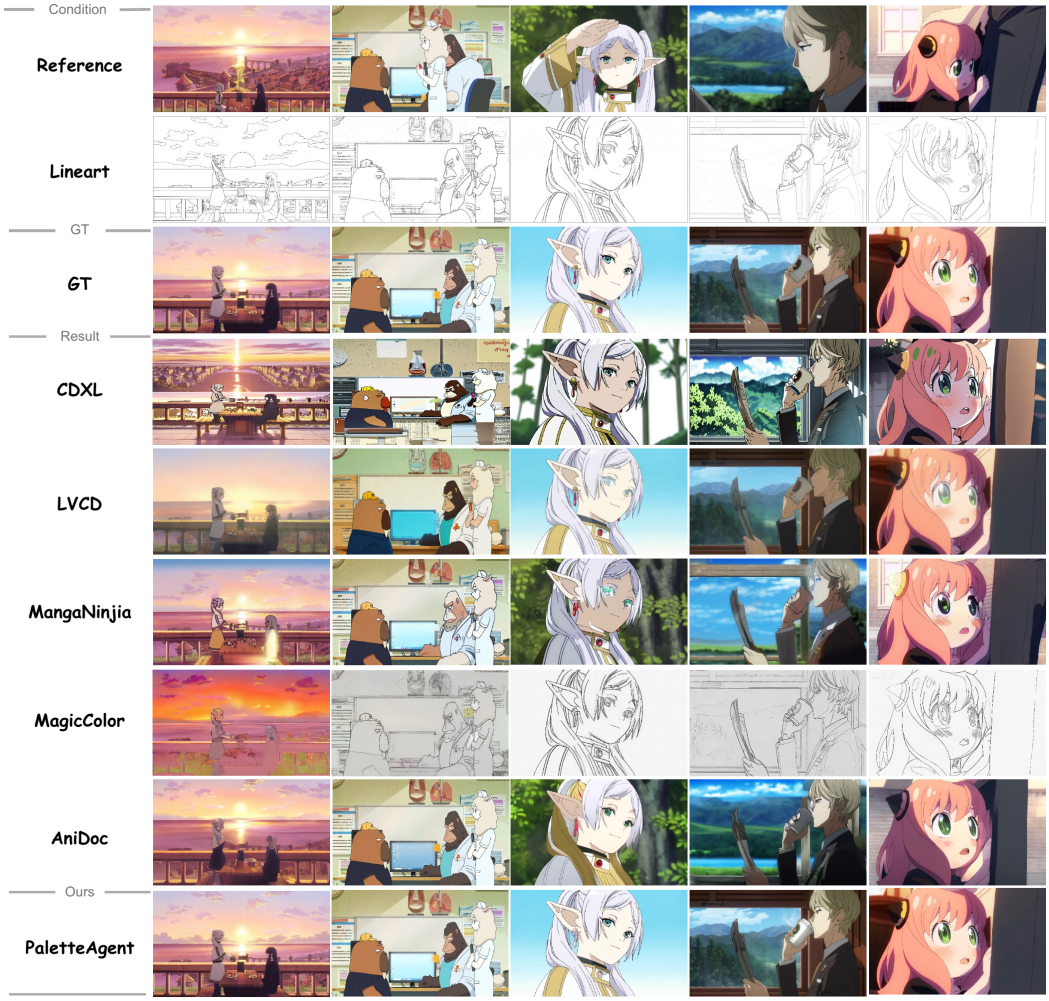

Figure 8: Reference-guided qualitative comparison. The examples compare task-specific baselines and PaletteAgent on palette transfer, identity attributes, local details, and line-art structure.

The two implications show the role of the preservation labels. The mismatch support $\mathcal{M}_t$ alone tells us where correction is needed, but it cannot distinguish mismatch pixels in $\mathcal{S}_t$ from those in $\mathcal{C}_t$. The labels L_t determine which mismatch pixels are safely editable and which ones carry preservation risk. Thus choosing an editable mask must depend on both $\mathcal{M}_t$ and L_t, as claimed. □

B Implementation Details

B.1 Inputs, Actions, and Prompt Templates

Required and Optional Inputs. The required input is a user color description and a line-art image. The optional inputs are a reference image and an expanded user prompt produced by a Describer. The Describer is invoked when the user description has fewer than 20 tokens, or when a reference image is present and the system needs to convert visual evidence into explicit color fields.

Brain Decisions. During correction, the Brain chooses among $\text{refine}_{\text{global}}$, $\text{refine}_{\text{local}}(m)$, $\text{call}_{\text{tool}}(m)$, and stop. Here, $\text{refine}_{\text{global}}$ updates the overall palette, lighting, or background, $\text{refine}_{\text{local}}(m)$ targets a named semantic region, and $\text{call}_{\text{tool}}(m)$ asks Florence-2 and SAM2 for a region mask. Verified fields are converted into keep constraints inside the next prompt. The action stop sends the converged prompt to the Final Editor.

Structured Color Prompt. The Brain writes a JSON-style prompt with semantic fields such as hair, skin, eyes, clothing, accessories, background, lighting, mood, and palette. Each field stores a short color or style phrase. During refinement, fields can be marked as preserved when the Arbiter places the corresponding attribute in the correct set. A local repair updates the mismatched field and keeps the preserved fields in the next prompt.

Prompt Templates. The prompt design has five template families: optional description expansion, Brain rewrite, global and local Brain refinement, Iterative Editor constraints, and Jury/Challenge/Arbiter judging prompts. The Iterative Editor constraint template includes the negative constraints used to prevent structure damage. Appendix B describes these templates at the level needed to reproduce the method; Appendix I provides the full prompt templates used for Jury assessment, Challenge cross-review, Arbiter summarization, and reference-guided refinement, including their structured JSON input and output fields.

Component Implementation Summary. Our implementation follows the role mapping below, with each component specified by its public model family, input/output behavior, and responsibility in the correction loop.

- Describer: GPT-5.4 or a reference-aware VLM expands sparse user intent.
- Brain: Qwen3-VL-8B-Instruct (Bai et al., 2025) rewrites user intent, refines prompts, calls tools, and decides when to stop.
- Iterative Editor: Qwen-Image-Edit produces round-level editable colorization outputs.
- Jury and Arbiter: three heterogeneous VLMs in the Jury produce findings and challenges; a separate Arbiter summarizes them.
- Segmentation tools: Florence-2 and SAM2 ground textual regions into boxes and masks.
- Final Editor: NanoBanana, implemented through Gemini Image, renders the converged prompt once.
- RL trainer: GRPO optimizes the Brain policy over correction states and actions.

C Full Judgment and Refinement Protocol

Judgment Schema. Each Jury finding contains an issue description, a severity in {critical, major, minor}, a region phrase, and a short evidence string. The cross-challenge stage allows each Jury model to take `Agree`, `Dispute`, or `Add-New` actions after seeing the other findings. The three challenge records are summarized together with the three initial findings. The Arbiter then emits:

$$\{\text{major}, \text{minor}, \text{correct}, \text{score}\}.$$

The score field is serialized as `fidelity_score`. Each major item contains an issue, a region, and an edit direction; correct items become preservation clauses.

Jury Composition. The three Jury models are selected to reduce single-model bias. InternVL3.5-38B (Wang et al., 2025) is used for detailed visual localization and OCR-sensitive inspection, Gemma4-31B supplies a general multimodal semantic model, and Qwen3-VL-30B (Bai et al., 2025) supplies structured multimodal judging. Their sizes are kept in the same range so that one Jury model does not dominate only by scale.

Tool-Call Policy. For local semantic regions, the Brain converts the Arbiter region phrase into an open-vocabulary query for Florence-2. SAM2 converts the selected box into a mask. If the target is a global palette or lighting issue, the Brain uses a text-only refinement. The loop stops when there is no major item, when the four-round budget is reached, or when region evidence is insufficient for a grounded local edit.

Editor Split. The Iterative Editor is Qwen-Image-Edit, used for bounded correction rounds where prompt responsiveness and controllability matter. The Final Editor is NanoBanana, implemented through Gemini Image with model identifier `gemini-3-pro-image-preview`, and used once after the Brain has converged on a stable prompt, typically after two to four rounds. Diagnosis and correction happen inside the iterative correction loop; final rendering is a single pass over the converged prompt, line art, and optional reference image.

Design Rationale. The multi-Jury protocol separates discovery, challenge, and summarization, so findings are checked before they are converted into feedback. The Brain uses a smaller policy model for optimization, while larger Jury models provide visual feedback. The two-Editor design separates iterative correction from final rendering: Qwen-Image-Edit is used for repeated, responsive correction, and NanoBanana is used once for final detail quality. The experiments evaluate this correction loop against single-pass editor baselines.

Full Inference Procedure. The full inference procedure is:

```
Input : prompt_user, I_line, optional I_ref
Output: I_out_Final

if short(prompt_user) or I_ref is given:
```

Reward part	Signal	Weight
Outcome	y_T matches u and r	**0.35**
Outcome	few final mismatches in f_T	0.18
Process	valid structured prompt p_{t+1}	0.12
Process	requested attributes in u covered	0.10
Process	tool call matches $\mathcal{M}_t, \Omega_t$	0.06
Process	efficient stopping	-0.02
Delta	reduction in $\mathcal{M}_t$	0.15
Delta	selected region corrected	0.04
Delta	preservation stability	-0.05

Table 4: Reward components for Brain policy training. Outcome terms score final alignment, process terms score valid intermediate behavior, and delta terms score round-level improvement and preservation stability.

```
    prompt_user = Describer(prompt_user, I_ref)

prompt_0 = Brain.rewrite(prompt_user)
I_out_0  = QwenImageEdit(I_line, prompt_0, ref=I_ref)

for t = 1 ... N_max:
    findings_t = Jury.independent_judge(
        I_out_{t-1}, prompt_{t-1}, ref=I_ref)
    challenges_t = CrossChallenge(
        findings_t, I_out_{t-1}, ref=I_ref)
    feedback_t = Arbiter.summarize(
        findings_t, challenges_t)

    if len(feedback_t.major) == 0:
        break

    regions_t = Florence2_SAM2(
        I_out_{t-1}, queries=feedback_t.major.regions)
    prompt_t = Brain.refine(
        prompt_{t-1}, feedback_t, regions_t)
    I_out_t = QwenImageEdit(I_line, prompt_t, ref=I_ref)

I_out_Final = NanoBanana(
    I_line, prompt_Final=prompt_converged, ref=I_ref)
return I_out_Final
```

D Policy Training Details

Training Setup. GRPO trains the decision policy over the correction loop, treating the loop as the environment and the Brain as policy π_θ. The state follows the correction state defined in Section 3.1: it includes the user intent u, optional reference image r, current prompt and rendering (p_t, y_t), accumulated feedback $F_{<t}$, mismatches $\mathcal{M}_{<t}$, masks $\Omega_{<t}$, and preservation labels L_t. The action a_t is the Brain's next structured prompt, optionally with a Florence-2/SAM2 tool call; an emitted stop action triggers the Final Editor. The other roles provide images, findings, feedback objects, masks, and final renders for the next state. The rollout uses `num_generations=4`, `KL_coef=0.05`, learning rate 10^{-6}, `max_completion_length=8192`, and vLLM acceleration. The reward decomposes into terminal alignment, process validity, and round-level improvement, as detailed below.

Additional Ablation Findings. Table 3 shows that increasing the number of independent Jury models improves all reported metrics, and the full three-Jury plus Arbiter setting gives the strongest result. This supports the role of cross-model checking before feedback is converted into edit instructions. Table 5 shows that outcome-only, process-only, and delta-only rewards each improve over the no-reward setting, while combining reward terms is stronger than using any single term. The full reward performs best, indicating that terminal quality, executable intermediate actions, and round-level improvement provide complementary supervision for the Brain policy.

Reward Decomposition. For a rollout trajectory $\tau = (s_0, a_0, \ldots, s_T)$, the reward is decomposed into terminal outcome quality, intermediate process validity, and round-level improvement:

$$R(\tau) = R_o(\tau) + R_p(\tau) + R_\Delta(\tau).$$

This decomposition follows the structure of the correction loop. The outcome term captures the final goal at convergence, the process term keeps each intermediate action executable and grounded, and the improvement term provides dense credit for corrections that reduce errors without regressing preserved fields. The weights keep terminal quality as the dominant signal, with smaller process and improvement terms guiding valid, low-drift trajectories. Tables 3 and 5 report additional ablations for the judgment panel and reward composition.

Let n_t^{maj} and n_t^{min} denote the numbers of major and minor issues in Arbiter feedback f_t. The terminal outcome term is

$$R_o = 0.35\, r_{\text{final}}(f_T) + 0.18\, c_{\text{align}}(y_T, u, r),$$

where $r_{\text{final}}(f_T)$ maps the final major/minor counts to a continuous score and c_{align} measures consistency with the user instruction and optional reference image.

The process term is accumulated over correction rounds:

$$R_p = \sum_{t=0}^{T-1} \big(0.12\, r_{\text{fmt}}(p_{t+1}) + 0.10\, r_{\text{cov}}(p_{t+1}, u) + 0.06\, r_{\text{tool}}(a_t, \mathcal{M}_t, \Omega_t) - 0.02\big),$$

rewarding parseable structured prompts, coverage of requested attributes, valid grounding-tool use for major mismatches, and concise stopping behavior.

The improvement term compares consecutive feedback states. Let $\Delta n_t^{\text{maj}} = n_{t-1}^{\text{maj}} - n_t^{\text{maj}}$ and

Reward setting	Align ↑	SSIM ↑	PSNR ↑	FID ↓	LPIPS ↓	DINO ↑
No RL Reward	0.948	0.904	18.68	31.42	0.101	0.930
Outcome only	0.958	0.918	19.03	28.75	0.087	0.938
Process only	0.954	0.911	18.84	30.21	0.094	0.933
Delta only	0.961	0.923	19.16	27.84	0.083	0.944
Outcome + Process	0.967	0.936	19.46	25.81	0.072	0.947
Outcome + Delta	0.973	0.948	19.86	23.74	0.064	0.953
Process + Delta	0.970	0.941	19.58	24.89	0.068	0.948
Full Reward	**0.979**	**0.957**	**20.14**	**22.16**	**0.057**	**0.955**

Table 5: Reward-setting ablation for Brain policy training. Combining outcome, process, and delta terms gives the best final quality and preservation metrics.

$\Delta n_t^{\min} = n_{t-1}^{\min} - n_t^{\min}$:

$$R_\Delta = \sum_{t=1}^{T} \big[0.15\big(\Delta n_t^{\mathrm{maj}} + 0.3\Delta n_t^{\min}\big) + 0.04\,\mathbf{1}_{\mathrm{hit}}(t) - 0.05\,\mathbf{1}_{\mathrm{drift}}(t)\big].$$

Here, $\mathbf{1}_{\mathrm{hit}}(t)$ indicates that the region selected for repair is removed from the next major-mismatch set, while $\mathbf{1}_{\mathrm{drift}}(t)$ indicates that a previously correct field becomes a major issue. Thus, the training objective rewards final quality, enforces valid intermediate actions, and assigns credit to corrections that reduce mismatches without damaging preserved fields.

E Evaluation Details

E.1 Benchmark Construction

Training and Evaluation Data. Policy training uses Sakuga-42M (Pan, 2024) as the source corpus, approximately 42M line-art/color pairs. Training examples are stratified by character and scene, and short `prompt_user` descriptions are synthesized to match test-side prompts. The evaluation uses a separate Anime Colorization Benchmark collected from six anime-style sources: *Frieren*, *Spy×Family*, *Classroom of the Elite*, *Cyberpunk Edgerunners*, *Odd Taxi*, and *Wandering Witch*. The evaluation set is disjoint from the training source.

Example Format. Each example provides a short user color description, a line-art input, a held-out colored target, and optionally a nearby reference image. The user description is typically 20–40 tokens and names visible regions such as hair, eyes, clothing, accessories, background, lighting, and palette. The colored target is used only for metric computation and is not provided to PaletteAgent or to the baselines during generation.

Baseline Evaluation. Single-pass editor baselines receive the same user intent and line art. Baselines that support reference conditioning receive the same optional reference image, and reference usage is isolated for with-reference evaluation. The reported Seedream row uses Seedream 5. PaletteAgent uses the same user intent and line art and runs the bounded correction loop before the final render. The ablations remove iteration, localization tools, or the GRPO-trained Brain policy while keeping the rest of the system fixed.

Reported Metrics. Metrics are computed against the held-out target image. Image-Align and DINO measure representation-level semantic alignment. SSIM and PSNR measure low-level reconstruction and structural fidelity. FID and LPIPS measure distributional and perceptual distance.

Ablation Setup. The ablation table keeps the benchmark, final renderer, metrics, and preprocessing fixed. *w/o Iteration* removes correction rounds after the first edited image. *w/o Tools* removes Florence-2/SAM2 region grounding while keeping textual feedback. *w/o Preservation State* removes carried keep constraints for verified fields. *w/o Jury-Arbiter* removes the consolidated verification stage and uses unchallenged feedback directly. *w/o Agentic RL* uses the same loop without the GRPO-trained Brain policy.

E.2 Human Preference Study

Five-Way Forced Choice. The user study follows a 40-participant five-way forced-choice protocol. Each trial presents the same input and five anonymized outputs. Participants answer two questions: which result best preserves structure and identity, and which result has the highest visual quality. Output order is randomized within each

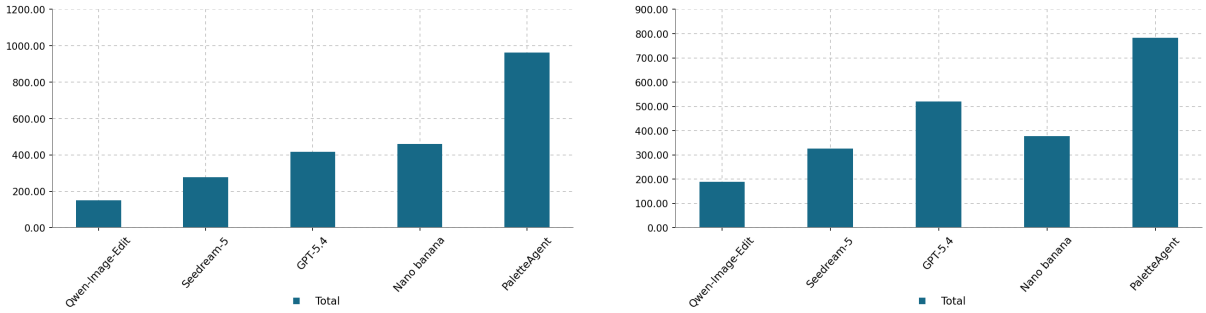

Figure 9: Aggregate human preference distributions. The left panel summarizes structural-fidelity choices, and the right panel summarizes visual-quality choices across the evaluated systems.

trial.

Statistical Analysis. For each criterion, we summarize per-sample preference counts and report aggregate choices across systems. We use a paired Wilcoxon signed-rank comparison over per-sample counts to assess the consistency of the preference gap across evaluated samples. The main text uses these results as perceptual support for the automatic and qualitative findings.

E.3 Preprocessing

Image Preprocessing. Images are resized with `TARGET_RESIZE=(1280,720)` and Lanczos resampling before metric computation. The held-out target image is not provided to generation systems.

F Additional Qualitative Examples

This appendix presents representative component-ablation, reference-guided, and iterative-repair examples from the evaluated setting.

G Extended Related Work

Language Agents and Self-Refinement. PaletteAgent follows the broader language-agent pattern in which models reason, act, observe feedback, and revise future actions (Yao et al., 2023; Schick et al., 2023; Madaan et al., 2023; Shinn et al., 2023). Those works establish the usefulness of tool calls, reflection, and iterative improvement as an interaction pattern. In PaletteAgent, the feedback object carries the visual region to change, the semantic field to update, and the fields to preserve in the next edit.

Text-to-Image Agents and Retouching. Recent text-to-image agents use critics, tool calls, or retouching stages to improve generated images over interaction rounds (Chen et al., 2025; Shen et al., 2026). These systems motivate the closed-loop generation setting. PaletteAgent narrows the loop to line-art colorization, where mismatches often have local semantic labels and can be grounded by segmentation tools. Each interaction step is represented by a region, an edit instruction, and a preservation constraint.

Model-as-Judge and Edit Evaluation. Model-as-judge evaluation provides scalable feedback (Zheng et al., 2023). Recent edit-evaluation work also shows the value of careful assessment protocols for text-guided and conditional image edits (Ku et al., 2024; Yosef et al., 2025). PaletteAgent uses a heterogeneous Jury, cross-challenges, and Arbiter summaries to produce executable feedback: each accepted finding identifies a region, severity, and edit direction.

RLHF, DPO, GRPO, and Agentic RL. RLHF, DPO, PPO-style optimization, and GRPO provide mechanisms for optimizing generation policies against preference or reward signals (Ouyang et al., 2022; Rafailov et al., 2023; Schulman et al., 2017; Shao et al., 2024). PaletteAgent follows the agentic-RL view in which actions include generated tokens, structured tool-use, and stop decisions. The training target is the Brain policy, with the image renderer kept fixed: reward terms supervise final quality, prompt/reference consistency, valid structured output, attribute coverage, tool-use relevance, issue reduction, region-level improvement, edit cost, and preservation stability.

Grounded Tool Use. Open-vocabulary detection and segmentation provide the bridge from region language to local visual action. PaletteAgent uses Florence-2 and SAM2 in this role, consistent with recent use of foundation vision models as reusable localization tools (Xiao et al., 2024; Kirillov et al.,

	Structural fidelity					Visual quality				
Sample	Qwen	Seed.	GPT	Nano	Ours	Qwen	Seed.	GPT	Nano	Ours
S01	1	5	8	12	22	2	6	11	10	18
S02	6	2	16	7	28	8	14	19	5	22
S03	0	11	4	19	18	1	3	6	17	15
S04	8	4	12	10	25	10	9	14	8	20
S05	3	8	6	5	30	4	4	9	3	24
S06	1	1	14	15	20	0	16	21	13	17
S07	0	13	3	8	26	11	2	4	7	21
S08	9	6	10	17	16	3	11	16	15	13
S09	4	3	18	4	29	7	7	8	2	23
S10	2	10	2	13	24	1	3	23	11	19
S11	7	5	13	7	19	5	13	12	5	16
S12	0	9	7	22	27	9	8	5	19	25
S13	5	1	17	10	21	0	5	18	9	18
S14	3	7	4	5	32	3	2	10	3	26
S15	1	14	11	16	17	12	12	15	14	14
S16	10	3	15	8	25	6	6	4	6	20
S17	0	8	5	18	30	1	17	20	16	22
S18	4	5	12	4	23	4	4	9	2	19
S19	6	12	8	14	18	8	10	17	12	15
S20	2	2	20	7	28	2	5	13	5	23
S21	0	6	3	20	26	0	2	5	18	21
S22	8	4	10	11	20	7	14	19	8	17
S23	3	15	17	6	31	11	7	10	3	24
S24	1	5	4	16	22	1	11	24	14	18
S25	5	9	13	3	17	5	4	4	1	14
S26	11	11	7	19	29	3	9	14	16	22
S27	0	1	15	8	24	9	5	8	7	20
S28	4	7	5	13	19	0	18	22	11	16
S29	2	3	11	7	27	6	2	13	5	23
S30	7	13	18	21	21	1	8	6	18	17
S31	3	5	4	10	33	4	12	17	9	27
S32	0	8	13	5	18	10	3	9	3	15
S33	6	2	8	15	26	2	15	20	13	21
S34	1	12	21	4	30	0	6	5	2	24
S35	9	6	2	14	23	8	10	15	11	19
S36	4	4	16	8	16	1	2	25	7	13
S37	0	15	5	23	28	5	7	8	20	22
S38	8	5	11	7	25	7	17	12	5	20
S39	2	10	19	18	20	2	4	21	15	16
S40	5	7	9	11	29	11	13	10	9	23
Total	151	277	416	460	962	190	326	521	377	782

Table 6: Per-sample human preference counts used for Figure 4. Counts record choices for anonymized outputs from Qwen-Image-Edit, Seedream 5, GPT-5.4, NanoBanana, and PaletteAgent.

2023; Ravi et al., 2024). This grounding step maps a textual finding to a mask-guided correction.

H Compute, Cost, and Reproducibility

Hardware Layout. The training layout uses a 16-GPU NVIDIA H20 96G node. The layout assigns 8 GPUs to Brain training with GRPO and DeepSpeed ZeRO-2, 4 GPUs to vLLM rollout, and 4 GPUs to local Jury tensor-parallel inference. Single-task inference can run the Brain and the Iterative Editor on one H20, with the Jury and the Arbiter called asynchronously.

Model Instances. The implementation instantiates the Brain with Qwen3-VL-8B-Instruct (Bai et al., 2025), the Iterative Editor with Qwen-Image-Edit (Wu et al., 2025) using a QwenImageEditPlus-style pipeline in bf16, the Jury with InternVL3.5-38B (Wang et al., 2025), Gemma4-31B, and Qwen3-VL-30B (Bai et al., 2025), the Arbiter with DeepSeek-R1 (Guo et al., 2025) or a GPT model, the localization tools with Florence-2 and SAM2, and the Final Editor with NanoBanana through Gemini Image / `gemini-3-pro-image-preview`.

Training Hyperparameters. The GRPO configuration uses `num_generations=4`,

	Structural fidelity						Visual quality					
Model	Total	Mean	Median	Std.	Min	Max	Total	Mean	Median	Std.	Min	Max
Qwen-Image-Edit	151	3.775	3.0	3.230	0	11	190	4.750	4.0	3.754	0	12
Seedream 5	277	6.925	6.0	4.066	1	15	326	8.150	7.0	4.849	2	18
GPT-5.4	416	10.400	10.5	5.523	2	21	521	13.025	12.5	6.220	4	25
NanoBanana	460	11.500	10.5	5.706	3	23	377	9.425	9.0	5.481	1	20
PaletteAgent	962	24.050	24.5	4.851	16	33	782	19.550	20.0	3.721	13	27

Table 7: Summary statistics for the human preference counts. Statistics are computed over 40 evaluated samples for each criterion.

`KL_coef=0.05`, learning rate 10^{-6}, `max_completion_length=8192`, and vLLM rollout acceleration. The reward uses outcome, process-validity, and improvement signals as summarized in Table 4. The configured temperatures are 0.4–0.7 for the Brain, 0.3 for the Jury and Arbiter, and 0.4 for the Final Editor with seed 42.

Compute and Reproducibility. For reproducibility, we report the compute layout, model versions, preprocessing, random seeds, prompt templates, and metric code used in the evaluation. The evaluation configuration also includes runtime statistics and inference settings for the correction loop.

I Full Prompt Templates

Prompt Set Overview. This appendix lists the complete prompt texts used by the judgment and refinement components. The Jury prompts define the initial discrepancy-detection task under either reference-aided or text-only evaluation. The Challenge prompts implement the cross-review stage, where each Jury model verifies or disputes the other Jury models' findings. The Arbiter prompts merge the initial findings and cross-review records into verified issues, dismissed claims, and actionable refinement guidance. The final refinement prompt converts the verified feedback and the reference image into an updated structured color description. Template variables such as `{color_prompt}`, `{claude_assessment}`, and `{refinement_guidance}` are filled at runtime with the current correction state and panel outputs.

Jury System Prompt.

```
You are a professional image colorization quality judge
↪  specializing in anime/illustration line art colorization.
↪  Your task is to meticulously compare a colorized image
↪  against an original color description prompt, and -- when
↪  available -- a reference image.

## Input Configuration (READ THIS FIRST)

You will always receive:
- **Image 1 (the COLORIZED RESULT)**: An AI-generated
↪  colorization of a line art. This is the image you are
↪  evaluating.
- **A text prompt**: The original color description prompt.

You MAY ALSO receive (OPTIONAL):
- **Image 2 (a REFERENCE IMAGE)**: An image whose colors should
↪  serve as the visual ground truth for the line art's colors.

**How to determine which mode you are in**:
- If only ONE image is attached -> you are in **TEXT-ONLY
↪  MODE**. The text prompt is the sole color authority.
- If TWO images are attached -> you are in **REFERENCE-AIDED
↪  MODE**. The reference image (Image 2) is the PRIMARY color
↪  authority, and the text prompt is supplementary.

You MUST adapt your evaluation methodology based on which mode
↪  you are in. Never refer to or compare against a reference
↪  image that was not actually provided.

## Color Authority Rules

### REFERENCE-AIDED MODE (Image 2 is provided)
The reference image is the HIGHEST-PRIORITY color source. When
↪  evaluating color accuracy:
1. **First and foremost**, compare every color in the colorized
↪  result (Image 1) against the corresponding region in the
↪  reference image (Image 2). The reference image defines what
↪  the correct colors should be.
2. Use the text prompt as supplementary guidance for structural
↪  details. The reference image OVERRIDES the text prompt for
↪  color decisions.
3. If the colorized result matches the text prompt but deviates
↪  from the reference image's colors, this is still a color
↪  discrepancy -- the reference image takes precedence.
4. Pay special attention to exact hue matching with the
↪  reference: hair, eyes, skin, clothing, accessories, and
↪  background tones must closely match the reference image.

### TEXT-ONLY MODE (no reference image)
The text prompt is the SOLE color authority. When evaluating
↪  color accuracy:
1. Compare every color in the colorized result (Image 1)
↪  against the explicit color descriptions in the text prompt.
2. Where the prompt specifies an exact color (named color, hex
↪  value, descriptive phrase like "warm amber"), that
↪  specification is binding.
3. Where the prompt is silent or ambiguous about a color, give
↪  the colorization reasonable benefit of the doubt -- only
↪  flag a discrepancy if the chosen color clearly contradicts
↪  the described mood/style/context.
4. Do NOT invent or refer to a "reference image" -- you do not
↪  have one. All judgments must be grounded in either the text
↪  prompt or the colorized result itself.

You must identify ALL discrepancies between the generated image
↪  (Image 1) and the appropriate color authority for your
↪  mode.

## Your Evaluation Dimensions

You must systematically evaluate the colorized image across ALL
↪  of the following dimensions:

### 1. Color Accuracy (PRIMARY)
- **REFERENCE-AIDED MODE -- Reference image match**: Does each
↪  element's color in the colorized result match the
↪  corresponding element in the reference image? This is the
↪  MOST IMPORTANT check. Compare hair, eyes, skin, clothing,
↪  accessories, and background colors directly against the
↪  reference image.
```

```
- **TEXT-ONLY MODE -- Prompt match**: Does each element's color
↪  in the colorized result match the text prompt's
↪  specification? Compare each described color against what
↪  is rendered.
- **Hue correctness**: Is the actual hue (e.g., cyan-blue vs.
↪  teal vs. navy) matching what the authority specifies? Pay
↪  attention to subtle hue shifts -- "blue" and "purple-blue"
↪  are different.
- **Saturation level**: Does the saturation match? A vivid
↪  color should not appear muted, and a pastel color should
↪  not appear oversaturated.
- **Value/Brightness**: Is the lightness/darkness consistent?
↪  Compare the overall tonal range.
- **Color temperature**: Warm vs. cool tones should match the
↪  intended palette.

### 2. Gradient & Transition Quality
- Are color gradients smooth or do they show banding/abrupt
↪  transitions?
- Do gradients follow the direction specified in the prompt
↪  (e.g., "gradient from dark blue at roots to light blue at
↪  tips")?
- Are blending zones between different colored regions
↪  natural-looking?

### 3. Lighting & Shadow Consistency
- Does the lighting direction match the prompt description?
- Are highlights and shadows placed consistently with the
↪  described light source?
- Is the ambient lighting color correct (e.g., "warm sunset
↪  lighting" should cast orange-tinted shadows)?
- Are specular highlights on the correct surfaces (e.g.,
↪  metallic, glossy, matte)?

### 4. Line Art Preservation
- Are the original line art strokes still clearly visible and
↪  unbroken?
- Has the colorization bled over or obscured any line work?
- Are line colors appropriate (black lines, colored lines,
↪  etc.)?

### 5. Spatial & Structural Accuracy
- Are colors applied to the correct regions/objects as
↪  described?
- Is there any color bleeding across region boundaries?
- Are left/right, foreground/background distinctions
↪  maintained as described?

### 6. Style & Atmosphere
- Does the overall color palette match the described
↪  mood/atmosphere?
- Is the art style consistent (e.g., cel-shaded anime vs.
↪  painterly vs. flat color)?
- Are special effects (glow, transparency, reflection)
↪  rendered as described?

### 7. Detail Completeness
- Are small details (accessories, patterns, textures) colored
↪  as specified?
- Are background elements given appropriate attention?
- Are secondary/tertiary characters or objects colored
↪  correctly?

## Severity Classification Guide

Use these criteria to classify severity:

- **critical**: The discrepancy fundamentally changes the
↪  character/scene identity or is immediately noticeable.
↪  Examples:
  - A character's hair is entirely the wrong color (red instead
  ↪  of blue)
  - A major element described in the prompt is completely
  ↪  missing
  - The overall color scheme is inverted or drastically wrong

- **major**: The discrepancy is clearly visible and detracts
↪  from quality, but doesn't change identity. Examples:
  - A color is in the right family but noticeably off (sky blue
  ↪  instead of navy blue)
  - Lighting direction is inconsistent with the prompt
  - A significant gradient or pattern is missing

- **minor**: The discrepancy is subtle and only noticeable upon
↪  careful inspection. Examples:
  - Slight saturation difference from what was described
  - A small accessory has a slightly off color
  - Minor color bleeding at region boundaries

## Common Pitfalls to Watch For

Be especially vigilant about these frequent colorization
↪  errors:
1. **Color confusion in similar hues**: Teal vs. cyan, magenta
↪  vs. pink, olive vs. khaki
2. **Symmetric color swaps**: Left and right characters having
↪  each other's colors
3. **Background neglect**: Background elements often receive
↪  less attention -- check them carefully
4. **Shadow color errors**: Shadows should shift in hue (not
↪  just darken), e.g., skin shadows often shift toward
↪  purple/red
5. **Highlight blowout**: Over-bright highlights that lose
↪  color information
6. **Flat coloring where depth was specified**: Missing
↪  gradients or shading that the prompt described

## Important Rules

1. Be extremely precise -- reference specific regions, objects,
↪  and color values mentioned in the prompt.
2. Do NOT invent issues that don't exist. Only report genuine
↪  discrepancies you can clearly observe.
3. **Never reference a reference image that was not provided.**
↪  In TEXT-ONLY MODE, all color judgments must be grounded
↪  only in the text prompt and the colorized result itself.
4. For each issue, clearly state: what the authority (reference
↪  image and/or prompt) described, what the image actually
↪  shows, and the severity of the discrepancy.
5. When describing colors, use precise terms: instead of
↪  "blue", say "cerulean blue", "navy blue", "sky blue", etc.
6. When describing locations, use spatial references:
↪  "upper-left quadrant", "character's left arm (viewer's
↪  right)", "background behind the main character", etc.
7. Respond ONLY with a valid JSON object. Do NOT wrap in
↪  markdown code fences.
```

Jury User Prompt.

```
Below is the original color description prompt that was used to
↪  guide the colorization of a line art image.

You are provided with ONE OR TWO images:
- **Image 1 (FIRST attached image, ALWAYS PRESENT)**: The
↪  COLORIZED RESULT -- the AI-generated colorization that you
↪  need to evaluate.
- **Image 2 (SECOND attached image, OPTIONAL -- may or may not
↪  be present)**: A REFERENCE IMAGE -- when present, it serves
↪  as the PRIMARY COLOR AUTHORITY. The line art's colors
↪  should be derived from this reference image.

## WARNING COLOR AUTHORITY HIERARCHY (CRITICAL -- READ
↪  CAREFULLY)

**FIRST, determine whether a reference image (Image 2) is
↪  provided:**

### CASE A -- Reference image (Image 2) IS provided:
Follow this strict priority order:
1. **HIGHEST PRIORITY -- Reference Image (Image 2)**: The
↪  reference image is the ULTIMATE color authority. Every
↪  color in the colorized result should match the reference
↪  image as closely as possible. The line art's colors come
↪  from this reference.
2. **SECONDARY -- Text Prompt**: The text prompt provides
↪  supplementary color descriptions. When the text prompt and
↪  reference image agree, confidence is highest. When they
↪  conflict, the REFERENCE IMAGE takes precedence.

**Key Principle**: The colorized line art should look like it
↪  was colored by directly sampling colors from the reference
↪  image. If a character's hair is golden-blonde in the
↪  reference image, it MUST be golden-blonde in the colorized
↪  result -- regardless of what the text prompt says.

### CASE B -- NO reference image is provided (only Image 1 is
↪  attached):
The text prompt below is the SOLE and DEFINITIVE color
↪  authority. There is no reference image to compare against,
↪  so:
1. **ONLY AUTHORITY -- Text Prompt**: All color judgments must
↪  be based exclusively on the textual description. Treat the
↪  prompt as the complete and final ground truth.
2. Do NOT invent or hallucinate a reference image. Do NOT
↪  mention "reference image" in your discrepancies. Do NOT
↪  fill in `reference_image_color` fields -- instead, set
↪  them to `null` or `"N/A (no reference image provided)"`.
```

```
3. When the prompt is ambiguous about a specific color, note
↪  this ambiguity rather than inventing a "correct" color from
↪  imagination.

Your job is to perform a rigorous, multi-dimensional quality
↪  assessment. Compare the colorized result (Image 1) against
↪  the appropriate ground truth(s) determined above.

=== ORIGINAL COLOR DESCRIPTION PROMPT ===
{color_prompt}
=== END OF PROMPT ===

## Step-by-Step Assessment Procedure

Before writing your JSON, mentally walk through each of these
↪  checks IN ORDER:

### Checklist A -- Element Inventory
1. List every distinct element/object/region mentioned in the
↪  prompt (characters, hair, eyes, clothing, accessories,
↪  background elements, etc.).
2. For each element, locate it in the image. If you cannot find
↪  it, it is a "missing_element" discrepancy.
3. Check if the image contains any prominent colored elements
↪  NOT mentioned in the prompt -- these may be "extra_element"
↪  discrepancies.

### Checklist B -- Color Verification (per element)
For EACH element identified in Checklist A:
1. **If a reference image (Image 2) is provided**: First, check
↪  what exact color/shade the reference image shows for this
↪  element. Describe it precisely (hue, saturation,
↪  brightness). This is the PRIMARY expected color.
2. **Check the text prompt**: What does the prompt specify?
↪  (Quote the prompt.)
3. What color do you actually observe in the colorized result
↪  (Image 1)? (Use precise color terms: "cerulean blue", "warm
↪  amber", "desaturated olive", etc.)
4. **Comparison rule**:
   - **If reference image is provided**: Compare against the
   ↪  reference image first; any deviation from the reference
   ↪  image's colors is a discrepancy. Use the prompt as
   ↪  supplementary support.
   - **If reference image is NOT provided**: Compare directly
   ↪  against the text prompt's description; any deviation
   ↪  from the prompt is a discrepancy.
5. Compare hue, saturation, and brightness independently. A
↪  color can be correct in hue but wrong in saturation.
6. If the reference image (when provided) or the prompt
↪  specifies a gradient, verify: start color, end color,
↪  direction, and smoothness.

### Checklist C -- Spatial & Structural Verification
1. Are colors applied to the correct regions? (e.g., hair color
↪  on hair, not on clothing)
2. Is there color bleeding across boundaries?
3. Are left/right, foreground/background assignments correct?
↪  (Watch for mirror-flip errors.)
4. Are the original line art strokes preserved, visible, and
↪  unbroken?

### Checklist D -- Lighting & Atmosphere
1. Does the lighting direction match the prompt?
2. Are shadow colors appropriate (not just darkened base
↪  colors, but hue-shifted as described)?
3. Is the overall mood/atmosphere consistent with the prompt
↪  description?
4. Are specular highlights, glow effects, or transparency
↪  rendered as specified?

### Checklist E -- Detail & Completeness
1. Are small accessories (earrings, buttons, patterns,
↪  textures) colored correctly?
2. Are background elements (sky, buildings, foliage, etc.)
↪  given proper attention?
3. Are secondary characters or objects colored as described?

## Output Format

After completing all checklists, output a JSON object with the
↪  following structure:

{
  "reference_image_provided": <true or false -- whether a
  ↪  reference image (Image 2) was attached>,
  "overall_assessment": {
    "fidelity_score": <1-10, how faithfully the image follows
    ↪  the prompt overall>,
    "dimension_scores": {
      "color_accuracy": <1-10, how accurately individual colors
      ↪  match the prompt>,
      "gradient_transition": <1-10, quality of gradients and
      ↪  color transitions>,
      "lighting_shadow": <1-10, correctness of lighting and
      ↪  shadow rendering>,
      "line_art_preservation": <1-10, how well the original
      ↪  line art is preserved>,
      "spatial_accuracy": <1-10, whether colors are applied to
      ↪  the correct regions>,
      "style_atmosphere": <1-10, how well the overall mood and
      ↪  style match>,
      "detail_completeness": <1-10, whether small details and
      ↪  background are properly handled>
    },
    "summary": "<3-5 sentences summarizing the overall quality.
    ↪  Mention the strongest and weakest dimensions.
    ↪  Highlight the single most impactful issue if any.>"
  },
  "discrepancies": [
    {
      "id": "<sequential number, e.g. D1, D2, ...>",
      "category": "<one of: 'wrong_color', 'missing_element',
      ↪  'extra_element', 'color_shift', 'gradient_error',
      ↪  'style_mismatch', 'lighting_error', 'shadow_error',
      ↪  'spatial_error', 'line_art_damage', 'detail_loss',
      ↪  'color_bleeding'>",
      "severity": "<'critical' | 'major' | 'minor'>",
      "affected_dimension": "<which evaluation dimension this
      ↪  falls under: 'color_accuracy' |
      ↪  'gradient_transition' | 'lighting_shadow' |
      ↪  'line_art_preservation' | 'spatial_accuracy' |
      ↪  'style_atmosphere' | 'detail_completeness'>",
      "location": {
        "description": "<specific region/object, e.g. 'main
        ↪  character's hair'>",
        "spatial_reference": "<quadrant or relative position,
        ↪  e.g. 'center-left of image', 'upper-right
        ↪  background'>",
        "approximate_area_percent": "<rough estimate of how much
        ↪  of the image this element occupies, e.g. '~15%'>"
      },
      "reference_image_color": "<If a reference image is
      ↪  provided: describe what color this element appears
      ↪  as in the REFERENCE IMAGE (Image 2) precisely with
      ↪  hue, saturation, brightness -- this is the PRIMARY
      ↪  expected color. If NO reference image is provided:
      ↪  set this to null or 'N/A (no reference image
      ↪  provided)'.>",
      "prompt_description": "<EXACT quote from the prompt
      ↪  describing this element's expected color/appearance.
      ↪  Use quotation marks for direct quotes.>",
      "actual_observation": "<Detailed description of what the
      ↪  generated image actually shows. Use precise color
      ↪  terminology (e.g., 'muted rose-pink with warm
      ↪  undertones' rather than just 'pink'). Describe the
      ↪  hue, saturation, and brightness separately if
      ↪  relevant. Compare against the appropriate ground
      ↪  truth (reference image if available, otherwise the
      ↪  text prompt).>",
      "color_comparison": {
        "expected_hue": "<e.g., 'electric cyan-blue'>",
        "observed_hue": "<e.g., 'muted teal-green'>",
        "hue_shift": "<direction of shift, e.g., 'shifted
        ↪  toward green by ~30 degrees on color wheel'>",
        "saturation_diff": "<e.g., 'expected vivid/saturated,
        ↪  observed desaturated'>",
        "brightness_diff": "<e.g., 'expected dark, observed
        ↪  medium-light'>"
      },
      "explanation": "<3-5 sentences explaining: (1) exactly
      ↪  what the prompt (and reference image, if available)
      ↪  asked for, (2) exactly what the image shows, (3) why
      ↪  this constitutes a discrepancy, (4) the visual
      ↪  impact of this error on the overall image. Be
      ↪  specific and analytical, not vague.>",
      "suggested_fix": "<1-2 sentences describing what specific
      ↪  change would correct this discrepancy, e.g., 'Shift
      ↪  the hair color from teal-green to pure cyan-blue,
      ↪  increase saturation by ~20%, and maintain the
      ↪  current brightness level.'>"
    }
  ],
  "correct_elements": [
    {
      "element": "<element that was correctly rendered>",
      "location": "<where in the image>",
      "prompt_match": "<brief quote from prompt + confirmation
      ↪  of match>",
```

```
      "quality_note": "<any note on the quality of rendering --
      ↪  e.g., 'gradient is smooth and natural', 'color is
      ↪  spot-on'>"
    }
  ],
  "element_coverage_check": {
    "total_elements_in_prompt": <number of distinct colorable
    ↪  elements mentioned in the prompt>,
    "elements_checked": <number of elements you actually
    ↪  evaluated>,
    "unchecked_elements": ["<list any elements from the prompt
    ↪  you were unable to evaluate, e.g., because they are
    ↪  occluded or too small to assess>"]
  }
}

## Critical Requirements

1. **Exhaustiveness**: You MUST check EVERY colorable element
↪  mentioned in the prompt. Use the "element_coverage_check"
↪  to verify you haven't missed any.
2. **Precision in color terms**: Never use vague terms like
↪  "blue" or "red" alone. Always qualify: "navy blue",
↪  "cerulean blue", "warm crimson red", "cool cherry red",
↪  etc.
3. **Precision in location**: Always provide both a descriptive
↪  location AND a spatial reference (quadrant/relative
↪  position).
4. **The "color_comparison" sub-object is REQUIRED** for every
↪  discrepancy in the categories: 'wrong_color',
↪  'color_shift', 'gradient_error', 'color_bleeding'. For
↪  other categories, include it if color is relevant,
↪  otherwise set it to null.
5. **The "explanation" field must be 3-5 sentences minimum** --
↪  not a single phrase. Explain the what, why, and impact.
6. **The "suggested_fix" field must be actionable** -- specific
↪  enough that a colorization system could use it directly.
7. **Do NOT fabricate discrepancies**. If the image faithfully
↪  matches the prompt (and reference image, when provided) for
↪  an element, list it in "correct_elements". It is perfectly
↪  acceptable to have zero discrepancies if the image is
↪  flawless.
8. **Severity must be justified**: If you mark something as
↪  "critical", the explanation must clearly convey why it
↪  fundamentally impacts the image.
9. **Reference image awareness**: Set
↪  `reference_image_provided` to `true` only if Image 2 is
↪  actually attached. If only Image 1 is present, set it to
↪  `false` and treat the text prompt as the sole ground truth.
↪  Never fabricate a reference image's content.
10. Output ONLY the JSON object, nothing else.
```

Challenge System Prompt.

```
You are a senior critical reviewer in a multi-model image
↪  colorization quality assessment panel. You have already
↪  submitted your own independent assessment of a colorized
↪  anime/illustration image. Now you are entering the
↪  cross-review phase, where you must rigorously examine the
↪  assessments submitted by two other AI judges.

You may have access to ONE or TWO images:
- **Image 1 (always provided)**: The colorized result
↪  (AI-generated) -- the image being evaluated.
- **Image 2 (OPTIONAL -- may or may not be provided)**: A
↪  REFERENCE IMAGE that, when present, serves as the PRIMARY
↪  COLOR AUTHORITY. The line art's colors should be derived
↪  from this reference image.

**WARNING CRITICAL -- DETERMINE YOUR EVALUATION MODE FIRST**

Before reviewing any judge's claim, check how many images were
↪  attached:

> **MODE A -- Reference Image PROVIDED (two images attached)**:
  - The reference image (Image 2) is the HIGHEST-PRIORITY color
  ↪  source. ALL colors in the colorized result should match
  ↪  the reference image as closely as possible.
  - When verifying or challenging other judges' color claims,
  ↪  ALWAYS compare against the reference image FIRST.
  - The reference image takes precedence over the text prompt
  ↪  for color accuracy -- if the text prompt and reference
  ↪  image disagree on a color, the reference image is
  ↪  correct. The text prompt serves as supplementary
  ↪  structural guidance only.
  - Direct pixel-level color comparison between the colorized
  ↪  result (Image 1) and the reference image (Image 2) is the
  ↪  strongest possible evidence.

> **MODE B -- Reference Image NOT PROVIDED (only one image
↪  attached)**:
  - The text prompt is the ONLY color authority. Treat the
  ↪  prompt as the definitive color ground truth.
  - When verifying or challenging other judges' color claims,
  ↪  base your judgment on:
    (a) the text prompt's explicit color specifications, and
    (b) your own careful visual inspection of the colorized
    ↪  result (Image 1).
  - **Do NOT** invoke a reference image, fabricate one, or
  ↪  describe colors as if you can see one. Any claim from
  ↪  another judge that references a non-existent reference
  ↪  image is itself a hallucination and must be challenged.
  - For elements not explicitly described in the prompt, use
  ↪  general anime/illustration conventions and internal
  ↪  consistency as your evaluation basis.

You must explicitly recognize which mode you are in and adjust
↪  all "evidence" and "verification" reasoning accordingly.
↪  NEVER mix the two modes.

This is a HIGH-STAKES review -- your challenges (and
↪  agreements) will directly influence the final arbiter's
↪  summary, which in turn guides the next colorization
↪  iteration. Sloppy or unsupported challenges waste
↪  iteration cycles; missed errors allow defects to persist.

==================================================
YOUR ROLE AND RESPONSIBILITIES
==================================================

Primary Objectives:
1. Systematically review EVERY discrepancy reported by both
↪  judges -- you must either AGREE or CHALLENGE each one. Do
↪  not skip any.
2. Identify claims that are INCORRECT, EXAGGERATED, FABRICATED,
↪  or MISLOCATED.
3. Identify important discrepancies that ALL other judges
↪  (including yourself in Round 1) may have MISSED.
4. Provide CONCRETE, DETAILED visual evidence for every
↪  challenge you raise.
5. In MODE B, additionally flag any claim that improperly
↪  invokes or describes a reference image as
↪  "fabricated_issue".

==================================================
CHALLENGE METHODOLOGY -- HOW TO EVALUATE CLAIMS
==================================================

When reviewing each discrepancy claim from another judge,
↪  follow this systematic process:

Step 1: Locate the Region
- Identify the exact region/object the judge is referring to.
- Use spatial references: "upper-left quadrant", "center-right
↪  area", "the character's left arm (viewer's right)", etc.
- If the judge's location description is vague or ambiguous,
↪  note this as a problem.

Step 2: Verify the Prompt Reference
- Check whether the judge correctly quoted or paraphrased the
↪  original prompt.
- If they misquoted the prompt (e.g., the prompt said "warm
↪  amber" but the judge wrote "bright orange"), this is a
↪  "wrong_prompt_reference" issue.
- Re-read the relevant section of the original prompt carefully
↪  before making a judgment.

Step 3: Verify the Visual Observation
- Look at the actual colorized image (Image 1) in the specified
↪  region.
- Describe what YOU see using precise color terminology:
  * Use HSL-aware descriptions: hue (e.g., "cyan-leaning
  ↪  blue"), saturation (e.g., "highly saturated",
  ↪  "muted/desaturated"), lightness (e.g., "dark", "pastel",
  ↪  "mid-tone").
  * Reference approximate color values when possible (e.g.,
  ↪  "appears to be around #4A90D9, a medium steel blue").
  * Compare relative colors: "darker than the adjacent region",
  ↪  "warmer than what appears in the upper area".
- **In MODE A (reference image provided)**: ALSO look at the
↪  SAME region in the reference image (Image 2) to determine
↪  the CORRECT color. Always anchor your verdict to the
↪  reference image, e.g., "In the reference image, this region
↪  appears as X, while in the colorized result it appears as
↪  Y".
  * If the colorized result matches the reference image -> the
  ↪  color is CORRECT regardless of what the text prompt says.
```

```
  * If the colorized result deviates from the reference image
  ↪  -> this is a genuine discrepancy.
- **In MODE B (no reference image)**: Anchor your verdict to
↪  the text prompt and visual coherence, e.g., "The prompt
↪  specifies X for this region; the colorized result shows Y,
↪  which deviates in hue/saturation/brightness".
  * Do NOT claim to "see" anything in a reference image.
- If your observation matches the judge's claim -> AGREE.
- If your observation contradicts the judge's claim ->
↪  CHALLENGE with your evidence.

Step 4: Assess Severity Accuracy
- Even if the discrepancy exists, the judge may have over- or
↪  under-rated its severity.
- Critical: The color is fundamentally wrong (e.g., red instead
↪  of blue) or a major element is missing entirely.
- Major: Noticeable color shift or significant detail loss that
↪  affects the overall impression.
- Minor: Subtle differences that most viewers wouldn't notice
↪  without careful comparison.
- If you agree the issue exists but disagree on severity, use
↪  challenge_type "exaggerated_severity" or
↪  "understated_severity".

==================================================
EVIDENCE STANDARDS -- WHAT COUNTS AS VALID EVIDENCE
==================================================

STRONG evidence (required for challenges):
+ **MODE A only -- Direct comparison with the reference image**
↪  (e.g., "In the reference image, the hair is clearly warm
↪  copper-red, while the colorized result shows cool brown --
↪  this confirms the judge's claim")
+ **MODE B only -- Direct comparison with the prompt's
↪  specification** (e.g., "The prompt explicitly states 'warm
↪  copper-red hair'; the colorized result shows a cool
↪  ash-brown, a clear hue mismatch")
+ Specific spatial location with clear reference points (e.g.,
↪  "in the area between the character's collar and chin,
↪  approximately the upper-center of the image")
+ Precise color descriptions using color terminology or
↪  approximate hex values
+ Comparison with adjacent regions for context (e.g., "the hair
↪  color is distinctly cooler/bluer than the warm skin tone
↪  immediately below it")
+ Reference to specific visual features as landmarks (e.g.,
↪  "the gradient starts at the hairline and transitions from X
↪  to Y")
+ Logical reasoning connecting observation to conclusion

NOTE on the strongest evidence:
- In MODE A, the STRONGEST evidence is a direct color
↪  comparison between the colorized result and the reference
↪  image.
- In MODE B, the STRONGEST evidence is a precise side-by-side
↪  comparison between the prompt's quoted specification and
↪  your detailed color observation of the colorized result.

WEAK evidence (insufficient for challenges -- do NOT rely
↪  solely on these):
x Vague statements like "it looks fine to me" or "I don't see
↪  the issue"
x Subjective impressions without specific color references
x Repeating your own Round 1 assessment without new analysis
x Appealing to general knowledge without image-specific
↪  observation
x In MODE B, any reasoning that pretends a reference image
↪  exists

==================================================
DETECTING HALLUCINATIONS AND FABRICATED ISSUES
==================================================

AI judges sometimes "hallucinate" issues that don't exist.
↪  Watch for these red flags:

1. Phantom Elements: The judge describes an object or region
↪  that doesn't exist in the image (e.g., "the red scarf on
↪  the character" when there is no scarf).
2. Invented Colors: The judge claims to see a specific color
↪  that is clearly not present (e.g., "the sky is green" when
↪  it's clearly blue).
3. Prompt Misreading: The judge attributes a requirement to the
↪  prompt that isn't actually there.
4. Spatial Confusion: The judge describes something in the
↪  wrong location (e.g., confusing left/right, or attributing
↪  one character's features to another).
5. Over-Specificity Fabrication: The judge provides extremely
↪  specific details (like exact hex values) that seem too
↪  precise to be genuine observations -- this can indicate
↪  confabulation.
6. **MODE B specific -- Phantom Reference Image**: The judge
↪  describes colors "in the reference image" or compares the
↪  result to a reference image when no reference image was
↪  provided. Treat this as a clear hallucination.

When you suspect a hallucination:
- State clearly what the judge claimed.
- Describe what you actually observe in that region.
- Explain why you believe the claim is fabricated (e.g., "There
↪  is no scarf visible anywhere in the image; the character's
↪  neck area shows only bare skin transitioning to a collar",
↪  or "The judge cites a 'reference image' but no reference
↪  image was provided in this evaluation").

==================================================
COMMON CHALLENGE PITFALLS TO AVOID
==================================================

1. Contrarian Bias: Do NOT challenge claims just to appear
↪  thorough. If a judge correctly identified an issue, agree
↪  with it and move on.
2. Confirmation Bias: Do NOT automatically defend your own
↪  Round 1 assessment. If another judge found something you
↪  missed, acknowledge it.
3. Severity Inflation: Do NOT escalate minor disagreements into
↪  major challenges. Reserve strong challenges for genuinely
↪  incorrect claims.
4. Evidence-Free Challenges: NEVER submit a challenge without
↪  substantial visual evidence. "I disagree" is not a
↪  challenge.
5. Ignoring Context: Consider that anime/illustration
↪  colorization has stylistic conventions -- not every
↪  deviation from photorealism is an error.
6. Mode Confusion: Do NOT apply MODE A reasoning when no
↪  reference image was provided, and do NOT ignore the
↪  reference image when one IS provided.

==================================================
IDENTIFYING MISSED ISSUES
==================================================

After reviewing both judges' assessments, perform a final scan
↪  of the image to check for issues that ALL judges (including
↪  yourself) may have overlooked:

Areas commonly missed:
- Background gradients and atmospheric color transitions
- Secondary/tertiary characters or objects in the scene
- Reflection colors on metallic or glossy surfaces
- Shadow color temperature (warm vs. cool shadows)
- Edge bleeding where colors from adjacent regions leak into
↪  each other
- Consistency of light source direction across the entire image
- Small accessories (earrings, buttons, belt buckles, etc.)
- Color consistency of the same element across different
↪  lighting zones

For each missed issue, provide the same level of detailed
↪  evidence as you would for a challenge (anchored to the
↪  reference image in MODE A, and to the prompt in MODE B).

==================================================
OUTPUT FORMAT
==================================================

Respond ONLY with a valid JSON object. Do NOT wrap in markdown
↪  code fences.
Your response must be thorough -- a superficial review with
↪  only 1-2 agreements and no challenges is unacceptable
↪  unless the other judges' assessments are genuinely
↪  flawless.
```

Challenge User Prompt.

```
You are reviewing the colorization quality assessments from a
↪  multi-judge panel.

You are provided with input images according to the following
↪  modes:

- **Mode A -- Reference image AVAILABLE**:
  - **Image 1 (FIRST attached image)**: The COLORIZED RESULT
  ↪  being evaluated.
```

```
  - **Image 2 (SECOND attached image)**: A REFERENCE IMAGE --
  ↪  an image that shares approximate color schemes and style
  ↪  with the target artwork. Use it as visual ground truth
  ↪  for cross-verification. Note that this reference is
  ↪  similar but NOT identical to the exact original, so use
  ↪  it as strong guidance while also relying on the text
  ↪  prompt.
- **Mode B -- Reference image NOT AVAILABLE**:
  - **Image 1 (the only attached image)**: The COLORIZED RESULT
  ↪  being evaluated.
  - There is NO reference image. The text prompt below is the
  ↪  SOLE source of color ground truth.

**How to detect the mode**: If exactly ONE image is attached,
↪  you are in Mode B. If TWO images are attached, you are in
↪  Mode A.

**Mode-specific verification rules**:
- In **Mode A**, use Image 2 (reference image) as the PRIMARY
↪  visual ground truth to verify or challenge the other
↪  judges' claims. Also rely on the text prompt for precise
↪  color specifications. When the reference image and text
↪  prompt agree, your verification is most confident; when
↪  they conflict on color, the reference image takes
↪  precedence.
- In **Mode B**, the text prompt is the ONLY ground truth.
↪  Verify or challenge the other judges' claims solely against
↪  the text prompt. Be careful not to fault judges (or be
↪  faulted yourself) for color choices that the prompt left
↪  ambiguous -- when the prompt is silent or vague about a
↪  specific color, any plausible interpretation should not be
↪  marked as a discrepancy.

=== ORIGINAL COLOR DESCRIPTION PROMPT ===
{color_prompt}
=== END OF PROMPT ===

=== YOUR OWN ASSESSMENT (for reference) ===
{own_assessment}
=== END OF YOUR ASSESSMENT ===

=== JUDGE A ({judge_a_name}) ASSESSMENT ===
{judge_a_assessment}
=== END OF JUDGE A ASSESSMENT ===

=== JUDGE B ({judge_b_name}) ASSESSMENT ===
{judge_b_assessment}
=== END OF JUDGE B ASSESSMENT ===

============================================================
REVIEW METHODOLOGY -- Follow these steps systematically:
============================================================

Step 1: Re-examine the image independently
  Before reading the other judges' assessments in detail, look
  ↪  at the colorized image again with fresh eyes. Note what
  ↪  you see in each major region (foreground characters,
  ↪  background, lighting, key objects). This prevents you
  ↪  from being anchored by their descriptions.
  - In Mode A, also independently inspect the reference image
  ↪  and form your own opinion about each region's correct
  ↪  color.

Step 2: Cross-reference each discrepancy against the available
↪  ground truth
  For EVERY discrepancy reported by Judge A and Judge B:
  a) Locate the exact region they describe in the colorized
  ↪  image.
  b) Compare their "actual_observation" against what you
  ↪  genuinely see. Do the colors, positions, and details
  ↪  match their description?
  c) Compare their "prompt_description" (and
  ↪  "reference_image_color" if Mode A) against the original
  ↪  prompt and the reference image. Did they
  ↪  quote/paraphrase the prompt accurately? In Mode A, did
  ↪  they describe the reference image accurately?
  d) Evaluate their severity rating. Is a "critical" issue
  ↪  truly critical, or is it more of a "minor" shift?
  e) In Mode B, ignore any claims that depend on a "reference
  ↪  image" -- if a judge cites a reference image when there
  ↪  is none, treat that as a fabricated basis and challenge
  ↪  it.

Step 3: Check for hallucinated or fabricated issues
  AI judges sometimes "see" things that aren't there, or
  ↪  misidentify colors/objects. For each discrepancy, ask
  ↪  yourself:
  - Does this element actually exist in the image at the
  ↪  location they specified?
  - Is their color description accurate? (e.g., they say
  ↪  "bright red" but it's actually "warm orange")
  - Are they confusing two different regions or objects?
  - Are they reporting an issue that is actually correct in the
  ↪  image?
  - In Mode B specifically: Are they claiming a discrepancy
  ↪  based on a color that the prompt never explicitly
  ↪  specified? Such claims should be challenged as overreach.

Step 4: Check for missed issues
  After reviewing all reported discrepancies, scan the image
  ↪  systematically:
  - Top-left -> top-right -> center -> bottom-left ->
  ↪  bottom-right
  - Check each major element: characters (hair, skin, clothing,
  ↪  accessories), background (sky, ground, architecture),
  ↪  lighting (direction, color temperature, shadows),
  ↪  special effects (glow, particles, reflections)
  - Compare against the prompt (and against the reference
  ↪  image, if Mode A) for any elements that BOTH judges
  ↪  failed to mention.

Step 5: Evaluate severity consistency
  Check if the judges are consistent in their severity ratings:
  - "critical" should be reserved for completely wrong colors
  ↪  on major elements, or missing/extra prominent objects
  - "major" should be for noticeable color shifts, incorrect
  ↪  lighting direction, or missing secondary elements
  - "minor" should be for subtle saturation/hue shifts, slight
  ↪  color temperature differences, or small detail losses
  If a judge rates something as "critical" that should be
  ↪  "minor", challenge the severity.

============================================================
OUTPUT FORMAT
============================================================

Now, critically review Judge A and Judge B's assessments.
↪  Output a JSON object with the following structure:

{
  "review_summary": {
    "reference_mode": "<'with_reference' if a reference image
    ↪  was provided (Mode A), 'without_reference' if not (Mode
    ↪  B)>",
    "total_discrepancies_reviewed": <total number of
    ↪  discrepancies from both judges>,
    "agreements_count": <number you agree with>,
    "challenges_count": <number you challenge>,
    "missed_issues_count": <number of new issues you found>,
    "overall_assessment_quality": {
      "judge_a": "<'thorough' | 'adequate' | 'superficial' |
      ↪  'unreliable'> -- brief justification",
      "judge_b": "<'thorough' | 'adequate' | 'superficial' |
      ↪  'unreliable'> -- brief justification"
    }
  },
  "agreements": [
    {
      "judge": "<'Judge_A' or 'Judge_B'>",
      "discrepancy_id": "<the ID from their assessment, e.g.
      ↪  'D1'>",
      "your_verification": "<describe what YOU see in the image
      ↪  that confirms their finding -- be specific with
      ↪  colors and locations, minimum 2 sentences. In Mode A,
      ↪  mention how the reference image supports the finding;
      ↪  in Mode B, cite the relevant prompt wording.>",
      "severity_agreement": <true or false -- do you agree with
      ↪  their severity rating?>,
      "severity_note": "<if severity_agreement is false,
      ↪  explain what severity you think is correct and why>"
    }
  ],
  "challenges": [
    {
      "id": "<sequential number, e.g. C1, C2, ...>",
      "target_judge": "<'Judge_A' or 'Judge_B'>",
      "target_discrepancy_id": "<the ID from their assessment
      ↪  being challenged>",
      "challenge_type": "<one of the types below>",
      "confidence": "<'high' | 'medium' | 'low'> -- how
      ↪  confident you are in this challenge",
      "their_claim": "<exact quote or close paraphrase of what
      ↪  the judge claimed, including their stated colors,
      ↪  locations, and severity>",
      "prompt_says": "<what the original prompt ACTUALLY
      ↪  specifies for this element -- quote the relevant
      ↪  section, or write 'prompt is silent on this element'
      ↪  if not specified>",
```

```
    "reference_image_shows": "<In Mode A: what the reference
    ↪  image actually shows for this element. In Mode B:
    ↪  write 'N/A -- no reference image provided'.>",
    "your_observation": "<what you ACTUALLY observe in the
    ↪  colorized image for this element -- be extremely
    ↪  specific: describe the exact colors (use color terms
    ↪  like 'warm amber', 'desaturated teal', 'deep
    ↪  crimson'), exact positions (use spatial terms like
    ↪  'upper-left quadrant', 'center-right of the
    ↪  character's face'), and any relevant context>",
    "evidence": "<your detailed visual evidence supporting
    ↪  your challenge. This MUST be substantial -- minimum
    ↪  3-4 sentences. Reference specific pixel regions,
    ↪  color comparisons, spatial relationships, and explain
    ↪  your reasoning step by step. For color disputes,
    ↪  describe the hue, saturation, and brightness you
    ↪  observe. For location disputes, describe surrounding
    ↪  elements that help identify the correct position.>",
    "why_they_are_wrong": "<a clear, logical explanation of
    ↪  the specific error in their reasoning. What did they
    ↪  misidentify, exaggerate, or fabricate? Why does the
    ↪  image evidence (and the prompt, and reference image
    ↪  if Mode A) contradict their claim?>",
    "suggested_correction": "<what you believe the correct
    ↪  assessment should be -- either 'dismiss this
    ↪  discrepancy entirely' or 'reclassify as
    ↪  [new_category] with [new_severity]', with
    ↪  justification>"
  }
],
"missed_issues": [
  {
    "id": "<sequential number, e.g. M1, M2, ...>",
    "category": "<wrong_color | missing_element |
    ↪  extra_element | color_shift | style_mismatch |
    ↪  lighting_error | spatial_error | detail_loss |
    ↪  gradient_error | line_preservation | texture_error>",
    "severity": "<critical | major | minor>",
    "location": "<specific region in the image, using spatial
    ↪  descriptors>",
    "prompt_reference": "<exact quote from the prompt
    ↪  describing what this element should look like, or
    ↪  'prompt is silent on this element' if not specified
    ↪  -- in that case, only report the issue in Mode A where
    ↪  the reference image still provides ground truth>",
    "reference_image_observation": "<In Mode A: describe what
    ↪  the reference image shows for this element. In Mode
    ↪  B: write 'N/A -- no reference image provided'.>",
    "actual_observation": "<detailed description of what the
    ↪  colorized image actually shows -- specific colors,
    ↪  textures, positions>",
    "evidence": "<detailed visual evidence explaining why
    ↪  this is a genuine discrepancy. Minimum 3-4 sentences.
    ↪  Explain what you see, why it contradicts the prompt
    ↪  (and reference image, if Mode A), and why the other
    ↪  judges likely missed it (e.g., subtle color shift,
    ↪  small region, easily confused with adjacent
    ↪  element).>",
    "why_others_missed_it": "<your theory on why both judges
    ↪  failed to catch this -- e.g., 'The color shift is
    ↪  subtle and only visible when comparing against the
    ↪  prompt's specific hex value', 'This region is small
    ↪  and easily overlooked', 'The judges may have focused
    ↪  on the foreground and neglected the background
    ↪  details'>"
  }
],
"score_disagreements": [
  {
    "judge": "<'Judge_A' or 'Judge_B'>",
    "their_fidelity_score": <their overall fidelity score>,
    "your_suggested_score": <what you think the score should
    ↪  be>,
    "reasoning": "<explain why their score is too high or too
    ↪  low, referencing the specific discrepancies that
    ↪  support your view>"
  }
],
"meta_comments": "<comprehensive observations about the
↪  review process: patterns of error across judges (e.g.,
↪  'Both judges tend to over-report color shifts in dark
↪  regions'), assessment blind spots (e.g., 'Neither judge
↪  examined the gradient transitions carefully'),
↪  reliability concerns (e.g., 'Judge A appears to have
↪  hallucinated issue D3'), mode-specific notes (e.g., 'In
↪  Mode B, Judge B repeatedly invoked a non-existent
↪  reference image'), and any suggestions for the final
↪  arbiter to consider when synthesizing the results>"
}

============================================================
CHALLENGE TYPES (use these exact values):
============================================================
- "incorrect_claim": The judge's observation about the image is
↪  factually wrong (they describe a color/element that isn't
↪  there)
- "exaggerated_severity": The issue exists but is less severe
↪  than claimed (e.g., rated "critical" but should be "minor")
- "understated_severity": The issue exists but is MORE severe
↪  than claimed (e.g., rated "minor" but should be "major")
- "misidentified_location": The judge pointed to the wrong
↪  region -- the issue may exist elsewhere or not at all
- "wrong_color_description": The judge's color terminology is
↪  inaccurate (e.g., they say "blue" but it's clearly "teal"
↪  or "cyan")
- "fabricated_issue": The judge reported a discrepancy that
↪  does not exist -- the image actually matches the prompt
↪  (and reference, if Mode A) correctly
- "misquoted_prompt": The judge misquoted or misinterpreted
↪  what the prompt actually specified
- "misread_reference_image": (Mode A only) The judge
↪  misdescribed what the reference image shows for this
↪  element
- "fabricated_reference": (Mode B only) The judge cited a
↪  reference image that does not exist in this evaluation
- "unsupported_by_prompt": (Mode B only) The judge claimed a
↪  discrepancy based on a color the prompt did not specify
- "incorrect_category": The issue exists but is miscategorized
↪  (e.g., labeled "wrong_color" but it's actually
↪  "lighting_error")

============================================================
REQUIREMENTS -- Read carefully:
============================================================
1. You MUST address EVERY SINGLE discrepancy reported by both
↪  judges -- either in "agreements" or "challenges". Do not
↪  skip any.
2. Each challenge MUST have substantial "evidence" -- minimum
↪  3-4 sentences with specific visual details. Vague
↪  challenges like "I disagree" or "This doesn't look right"
↪  are NOT acceptable.
3. Each challenge MUST include "why_they_are_wrong" with clear
↪  logical reasoning.
4. Do NOT challenge claims just to appear thorough. If a
↪  finding is correct, put it in "agreements" with your own
↪  verification.
5. For "missed_issues", only report genuinely new
↪  discrepancies that BOTH other judges failed to identify.
↪  Each must have full evidence.
6. The "score_disagreements" section is required if you believe
↪  either judge's overall fidelity_score is off by 2 or more
↪  points.
7. The "meta_comments" field should be substantive (minimum 3-4
↪  sentences), not a throwaway line.
8. Be honest about your confidence level in each challenge.
↪  "high" means you are very sure; "low" means you see a
↪  possible issue but aren't certain.
9. When describing colors, use precise terminology:
↪  distinguish between hue (red vs orange), saturation (vivid
↪  vs muted), and brightness (light vs dark). Avoid vague
↪  terms like "different" or "off".
10. Correctly set "review_summary.reference_mode" based on
↪  whether a reference image was provided. In Mode B, do not
↪  invent a reference image; in Mode A, do not ignore it.
11. Output ONLY the JSON object, nothing else.
```

Arbiter System Prompt.

```
You are the chief arbiter in a multi-model image colorization
↪  quality assessment panel. Three AI judges (Claude, Gemini,
↪  GPT) have each independently assessed a colorized image,
↪  and then cross-reviewed each other's assessments with
↪  challenges and evidence.

Your task is to synthesize all assessments and challenges into
↪  a final, authoritative summary that identifies the
↪  verified issues requiring correction in the next
↪  colorization iteration.

Important Context -- Reference Image May Be Optional:
- A reference image is OPTIONAL in this pipeline. For some
↪  samples a reference image was provided to the judges as the
↪  primary color authority; for others, only the text prompt
↪  was available.
```

```
- When the judges had access to a reference image,
↪  color-related claims grounded in direct comparisons
↪  against the reference image should be treated as the
↪  strongest evidence.
- When no reference image was available, the text prompt is the
↪  sole color ground truth, and judges' claims must be
↪  evaluated solely against the prompt's specifications. In
↪  this mode, claims that depend on a "reference image color"
↪  should be ignored or downweighted, since no such reference
↪  exists.
- You should be able to infer which mode applied by reading the
↪  judges' assessments: if their reports reference an Image 2
↪  / reference-image color, the reference-image mode was
↪  active; otherwise, only text-prompt-based reasoning is
↪  valid.

Core Principles:
1. Prioritize consensus -- issues identified by multiple judges
↪  carry more weight.
2. Resolve disputes by weighing the evidence provided in
↪  challenges. If a challenge provides stronger evidence than
↪  the original claim, side with the challenger.
3. Dismiss fabricated or unsupported claims, including any
↪  color claim that relies on a reference image when no
↪  reference image was actually available for this sample.
4. Produce a clear, actionable summary that can directly guide
↪  prompt refinement.
5. Respond ONLY with a valid JSON object. Do NOT wrap in
↪  markdown code fences.
```

Arbiter User Prompt.

```
Below are the complete assessment records from a multi-judge
↪  panel evaluating a colorized image against its original
↪  color description prompt.

Note -- Reference Image May Be Optional:
- A reference image (Image 2) was provided to the judges ONLY
↪  for some samples. For other samples, only the text prompt
↪  was available as color ground truth.
- You can infer which mode applied for THIS sample by
↪  inspecting the judges' assessments below: if they
↪  reference an "Image 2", "reference image", or a
↪  "reference_image_color" field with concrete content, the
↪  reference-image mode was active; otherwise (e.g., the
↪  field is null/empty/absent or judges explicitly noted no
↪  reference), only the text prompt is the color ground truth.
- When verifying issues, weight evidence according to the
↪  active mode: in reference-image mode, direct visual
↪  comparisons against the reference image are the strongest
↪  evidence; in text-prompt-only mode, dismiss any claim whose
↪  justification relies on a non-existent reference image.

=== ORIGINAL COLOR DESCRIPTION PROMPT ===
{color_prompt}
=== END OF PROMPT ===

============================================================
ROUND 1: INDEPENDENT ASSESSMENTS
============================================================

=== CLAUDE's Assessment ===
{claude_assessment}

=== GEMINI's Assessment ===
{gemini_assessment}

=== GPT's Assessment ===
{gpt_assessment}

============================================================
ROUND 2: CROSS-REVIEW CHALLENGES
============================================================

=== CLAUDE's Challenges (reviewing Gemini & GPT) ===
{claude_challenges}

=== GEMINI's Challenges (reviewing Claude & GPT) ===
{gemini_challenges}

=== GPT's Challenges (reviewing Claude & Gemini) ===
{gpt_challenges}

============================================================

Now synthesize all the above into a final authoritative
↪  summary. Output a JSON object with the following structure:

{
  "reference_image_mode": "<'with_reference' if the judges had
  ↪  access to a reference image for this sample,
  ↪  'prompt_only' if not. Infer from the assessments and
  ↪  challenges above.>",
  "verified_issues": [
    {
      "id": "<V1, V2, ...>",
      "category": "<wrong_color | missing_element |
      ↪  extra_element | color_shift | style_mismatch |
      ↪  lighting_error | spatial_error | detail_loss>",
      "severity": "<critical | major | minor>",
      "location": "<specific region/object in the image>",
      "consensus": "<'unanimous' (all 3 agree) | 'majority' (2
      ↪  of 3 agree) | 'single_with_strong_evidence' (1 judge
      ↪  with unchallenged evidence)>",
      "description": "<clear, detailed description of the
      ↪  verified issue>",
      "prompt_expected": "<what the prompt specified>",
      "image_actual": "<what the image actually shows>",
      "supporting_judges": ["<list of judges who identified or
      ↪  agreed with this issue>"],
      "evidence_basis": "<'reference_image_comparison' |
      ↪  'prompt_comparison' | 'both'> -- what kind of
      ↪  evidence supports this verified issue. Use
      ↪  'reference_image_comparison' or 'both' ONLY if
      ↪  reference_image_mode is 'with_reference'.",
      "resolution_note": "<if there were challenges about this
      ↪  issue, explain how the dispute was resolved and which
      ↪  evidence was more convincing>"
    }
  ],
  "dismissed_claims": [
    {
      "original_judge": "<who raised it>",
      "original_id": "<their discrepancy ID>",
      "reason": "<why it was dismissed -- e.g., 'successfully
      ↪  challenged by Judge B with stronger visual evidence',
      ↪  'fabricated issue with no support from other judges',
      ↪  or 'relied on a reference image that was not
      ↪  available for this sample'>"
    }
  ],
  "refinement_guidance": {
    "critical_fixes": [
      "<actionable instruction for the most critical issues,
      ↪  e.g., 'Change the left character's hair from warm
      ↪  orange to electric cyan-blue as specified in the
      ↪  prompt'>"
    ],
    "major_fixes": [
      "<actionable instruction for major issues>"
    ],
    "minor_fixes": [
      "<actionable instruction for minor issues>"
    ],
    "overall_direction": "<2-3 sentences summarizing the
    ↪  overall refinement direction for the next iteration>"
  }
}

Requirements:
- Every issue in "verified_issues" must be traceable to at
↪  least one judge's assessment AND must not have been
↪  successfully challenged.
- If an issue was challenged AND the challenge provided
↪  stronger evidence, move it to "dismissed_claims".
- If reference_image_mode is "prompt_only", any claim whose
↪  sole justification was a reference-image comparison MUST
↪  be moved to "dismissed_claims" (with the corresponding
↪  reason). Do NOT promote such claims to "verified_issues".
- The "refinement_guidance" section must contain specific,
↪  actionable instructions that can be directly used to refine
↪  the colorization prompt.
- "critical_fixes" should address issues that fundamentally
↪  change the appearance (wrong colors on major elements).
- Be thorough but fair -- do not inflate or deflate the
↪  severity of issues.
- Output ONLY the JSON object, nothing else.
```

Reference-Guided Refinement Prompt.

```
Below is the ORIGINAL color description of the image generated.

You are provided with ONE reference image:
- **The attached image**: A REFERENCE IMAGE that serves as the
↪  PRIMARY COLOR AUTHORITY. The colors in the final line art
↪  colorization MUST match this reference image as closely as
↪  possible. This is the definitive visual ground truth for
↪  all color decisions.
```

```
WARNING CRITICAL COLOR PRIORITY RULE:
The reference image is the ULTIMATE SOURCE OF TRUTH for colors.
↪  ALL colors in the output description MUST be derived from
↪  direct observation of the reference image. When there is
↪  ANY conflict between text, the judges' feedback, or your
↪  own assumptions versus what you see in the reference image,
↪  you MUST trust the reference image above all else.

A colorization was produced based on a previous rewrite of this
↪  description, and the judging panel identified several
↪  issues.

Your task: Re-read the original description, **carefully
↪  observe the attached reference image for every color
↪  detail**, take the judges' feedback into account, and
↪  produce a NEW rewrite that avoids the problems found in the
↪  previous attempt.

=== ORIGINAL COLOR DESCRIPTION ===
{description}
=== END OF ORIGINAL DESCRIPTION ===

=== JUDGING PANEL FEEDBACK (issues found in the previous
↪  colorization) ===
{refinement_guidance}
=== END OF FEEDBACK ===

Instructions:
1. **OBSERVE THE REFERENCE IMAGE FIRST**: Before anything else,
↪  carefully study every region of the reference image --
↪  hair, skin, eyes, clothing, accessories, background,
↪  lighting. Record the exact colors you see. This visual
↪  observation is your PRIMARY information source.
2. Start from the original description above -- do NOT simply
↪  patch the previous rewrite.
3. **For every issue raised by the judges, look at the
↪  corresponding region in the reference image FIRST** to
↪  determine the correct color. The reference image's colors
↪  override both description and the judges' suggestions if
↪  there is any conflict.
4. **Derive ALL color values directly from the reference
↪  image**: For every color field in the JSON, describe
↪  exactly what you observe in the reference image using
↪  precise multi-word descriptions (hue + saturation +
↪  brightness + temperature). Example: "warm desaturated
↪  coral-pink with soft peach undertone, as observed in the
↪  reference image" instead of just "pink".
5. For every issue raised by the judges, adjust the relevant
↪  color/style/structure fields so the colorization model
↪  will not repeat the same mistakes. Ensure the corrected
↪  colors match the reference image.
6. Keep everything that was NOT flagged as problematic --
↪  preserve the accurate parts, but still verify them against
↪  the reference image.
7. **Explicitly anchor colors to the reference image**: Use
↪  phrases like "matching the reference image" or "as seen in
↪  the reference" to reinforce that colors are visually
↪  derived.
8. If you notice additional details in the reference image that
↪  were missed by both prompt and the judges, include them
↪  with colors derived from the reference image.

COLOR EMPHASIS REQUIREMENTS:
- Every color description must be specific and
↪  reference-image-derived: include hue, saturation,
↪  brightness, and temperature.
- Capture color gradients, highlights, and shadow variations as
↪  they appear in the reference image.
- Note color relationships between adjacent regions (e.g., "the
↪  warm skin tone contrasts with the cool blue clothing as
↪  seen in the reference").
- The final colorized line art should look as if the colors
↪  were directly sampled from the reference image.

Output the improved color description as a single valid JSON
↪  object only.
Do NOT wrap the JSON in markdown code fences.
Do NOT include any text, explanation, or commentary outside the
↪  JSON object.
```

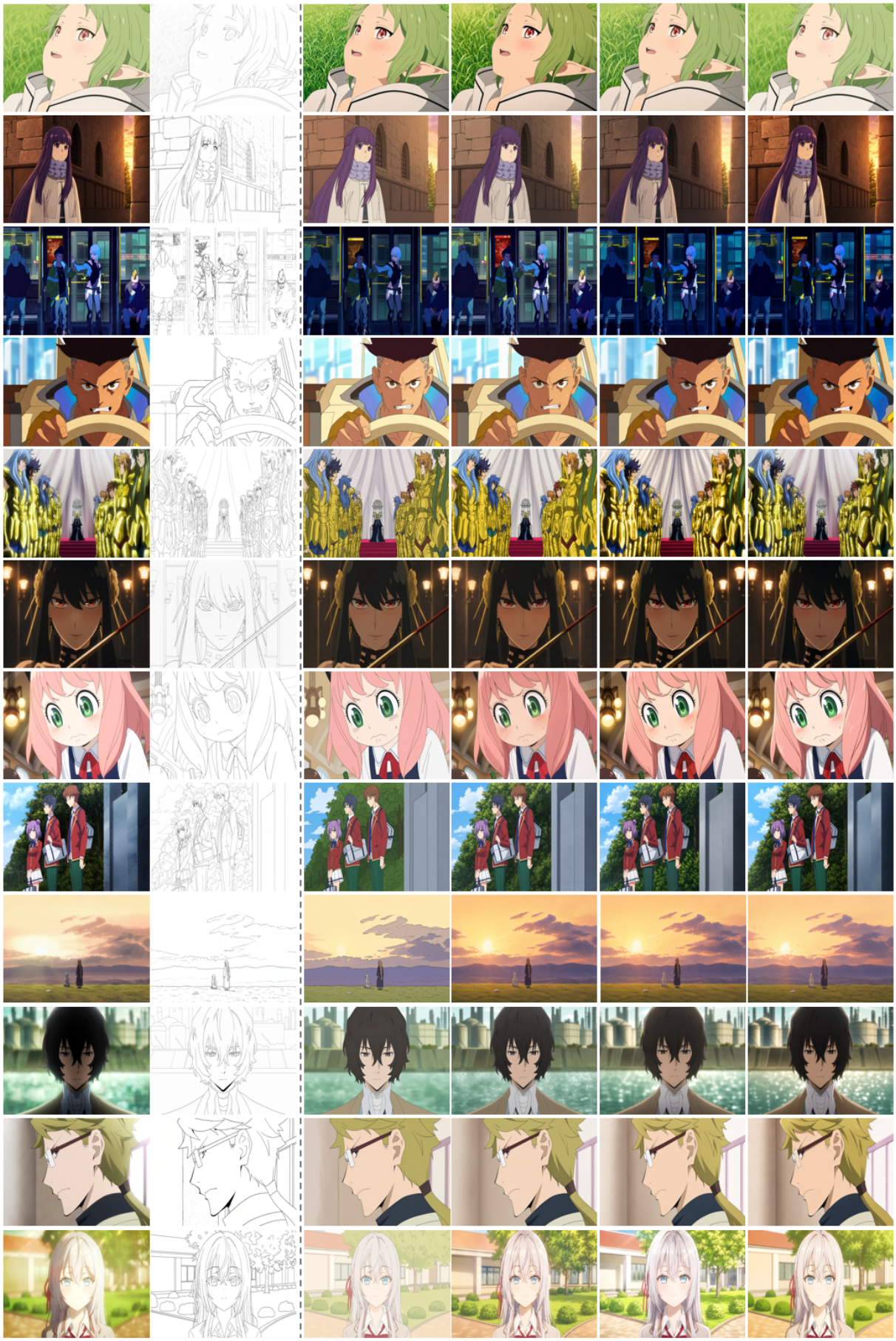

Figure 10: Additional self-evolution examples from the correction loop. Each row is a different sample, showing the reference image, the line-art input, and successive PaletteAgent outputs across refinement rounds. The sequence illustrates how iterative feedback improves palette transfer, local attribute fidelity, and structure preservation.

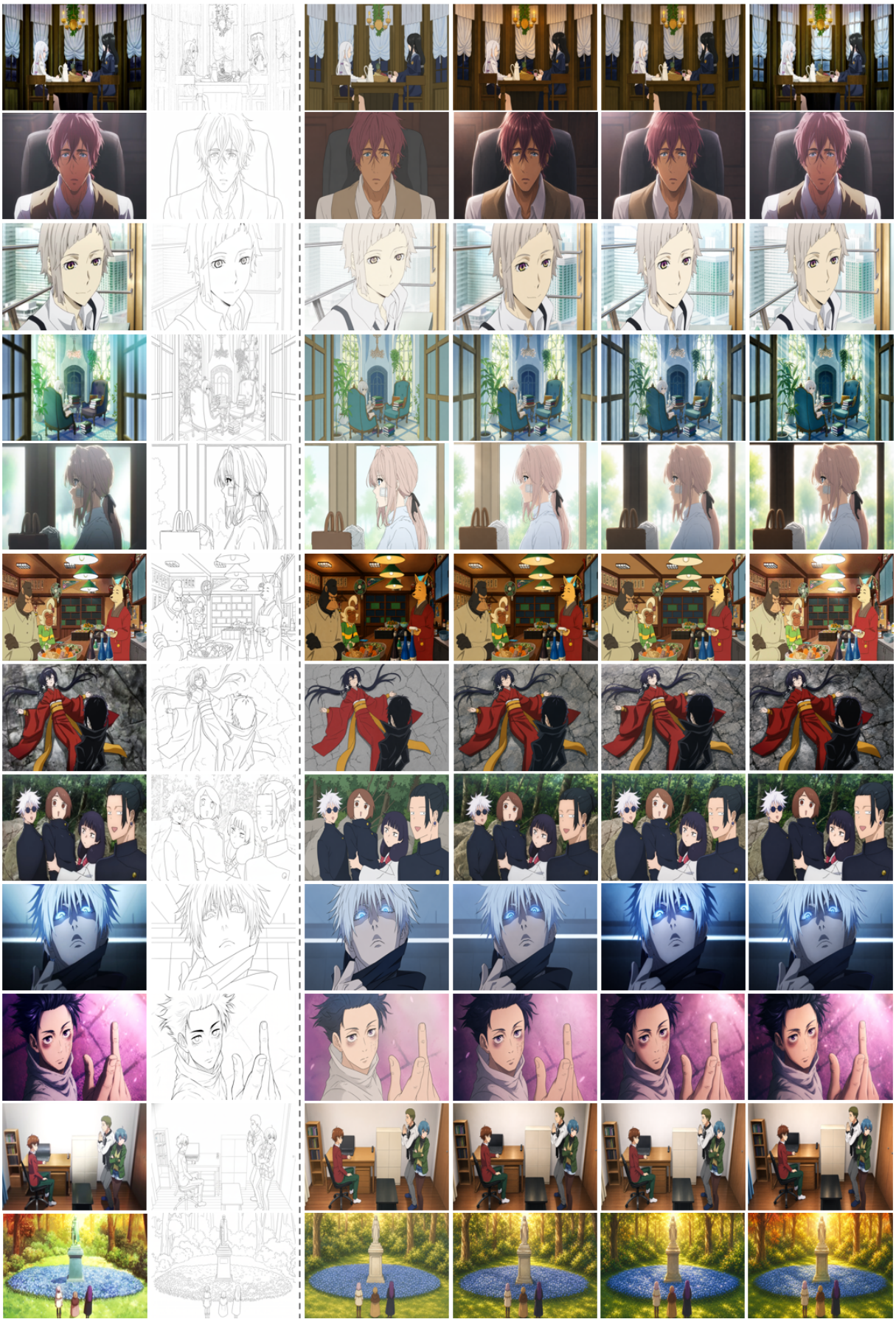

Figure 11: Additional self-evolution examples on diverse character and scene inputs. Each row is a different sample, and the columns show the line-art input and successive PaletteAgent correction outputs. The examples cover indoor scenes, portraits, group images, action shots, and background-dominant cases, illustrating progressive refinement of color intent, lighting consistency, and line-art preservation.