

AdaRetouch: An Interleaved Vision-Language Agent for Adaptive Multi-Turn Image Retouching

Ziang Liu^{1,2,*} Yongheng Zhang^{1,*} Qi Wen^{2,*} Kai Zhao^{2,†}
Jianwei Zhang² Yuezhou Dong² Yiren Hao² Bo Li²
Peishan Dai^{1,‡} Hao Zhang^{2,‡}

¹School of Computer Science and Engineering, Central South University, China

²vivo BlueImage Lab, vivo Mobile Communication Co., Ltd., China

*Equal contribution. †Project lead. ‡Corresponding authors.

Abstract

Image retouching requires sophisticated aesthetic understanding and precise parameter control, yet existing agents typically rely on fixed, open-loop execution or software-specific supervision. We introduce *AdaRetouch*, an interleaved vision-language agent for adaptive, multi-turn image retouching. Given an instruction and an image, *AdaRetouch* dynamically diagnoses the visual state, invokes non-destructive tools, and iteratively observes the results until the intent is satisfied. To train this closed-loop behavior, we propose a software-agnostic pipeline that constructs trajectories featuring curriculum-driven tool sequences and step-wise reasoning. *AdaRetouch* is initialized via supervised fine-tuning and further optimized using process-level reinforcement learning for quality and efficiency. Experiments on *AdaRetouch-Bench* show that *AdaRetouch* outperforms general MLLMs, existing agents, and diffusion editors in intent alignment, perceptual quality, and calibrated tool planning.

1 Introduction

Image retouching is essential for improving the perceptual quality and communicative value of visual content. By adjusting exposure, contrast, color temperature, tonal balance, and local hues, it can transform imperfect photographs into more compelling images (Hu et al., 2018; Chen et al., 2018; Moran et al., 2020; Wu et al., 2024). As a result, it is widely used in professional photography, social media content creation, e-commerce presentation, and mobile imaging (Gharbi et al., 2017; Lv et al., 2021; Delbracio et al., 2021). Meanwhile, although modern editing software such as *Photoshop* and *Lightroom* provides powerful restoration and enhancement tools through fine-grained sliders and local adjustments, producing high-quality retouched results **remains challenging** for users (Gokbudak and Oztireli, 2023; Chen et al., 2025). As effective retouching demands both aesthetic expertise and

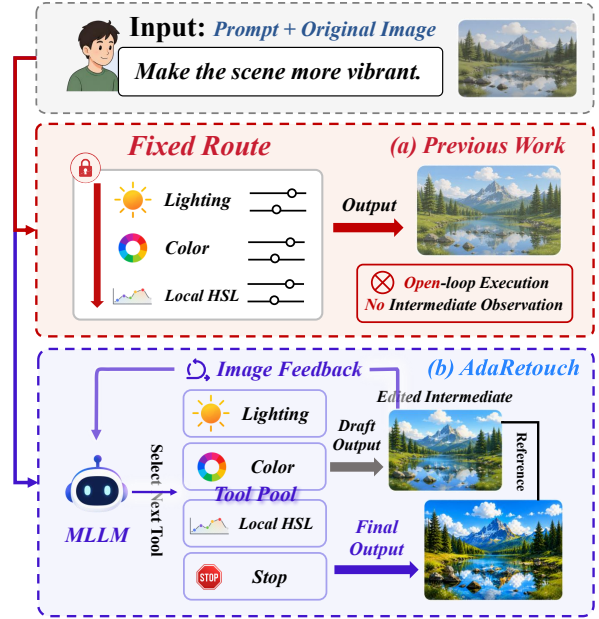


Figure 1: (a) Previous Work vs. (b) *AdaRetouch*. Previous retouching methods follow **fixed open-loop** tool routes without **intermediate feedback**, while *AdaRetouch* **adaptively selects** tools based on intermediate image observations for **multi-turn retouching**.

fine-grained manual control under varied user intents, automatic systems capable of understanding user intent, diagnosing aesthetic flaws, and executing precise edits are of substantial research and practical value (Kong et al., 2016; Talebi and Milanfar, 2018; Ke et al., 2021).

The emergence of multimodal large language models (MLLMs) has rapidly advanced automatic image retouching, making practical systems that can understand user intent, reason about visual aesthetics, and perform executable editing operations increasingly feasible (Zhang et al., 2025, 2026b,c). Specifically, recent studies have begun to explore MLLM-based retouching agents that translate natural-language instructions and visual feedback into controllable, non-destructive editing actions. *MonetGPT* (Dutt et al., 2025) teaches MLLMs to understand procedural editing

operations through visual puzzles, while *Jarvis-Art* (Lin et al., 2026) coordinates Lightroom tools according to user instructions. Building on this direction, *JarvisEvo* (Lin et al., 2025) introduces a self-evolving editor-evaluator framework to improve agent performance, and *RetouchIQ* (Wu et al., 2026) applies reinforcement learning with generalist reward models for optimization. Collectively, these works demonstrate the promise of MLLM agents for interpretable, executable, and controllable image retouching (Zhang et al., 2026a). Despite their progress, as shown in Figure 1 (a), existing methods still face key challenges:

- **Inflexible Execution and Limited Adaptive Reasoning.** Existing methods predict parameters in one pass or follow tool sequences (Dutt et al., 2025; Wu et al., 2026). Yet retouching is state-dependent, requiring observation and adjustment. Such workflows struggle with diverse images and intents, causing redundant edits, missed refinements, or over-editing.
- **Tool-Coupled Supervision and Limited Generalization.** Current datasets (Bychkovsky et al., 2011; Liang et al., 2021) and agents (Lin et al., 2025, 2026) are often tied to specific software, limiting transferability. Meanwhile, the scarcity of multi-step trajectories forces reliance on final-output supervision, leaving agents without sufficient intermediate feedback for robust, tool-agnostic editing.

To address the challenges, we introduce the *AdaRetouch*, an interleaved vision-language agent for adaptive multi-turn image retouching. As illustrated in Figure 1 (b), unlike previous fixed-route and open-loop retouching pipelines, *AdaRetouch* performs closed-loop editing by observing intermediate results and adaptively selecting the next tool step by step. Given a user instruction and an input image, the agent first infers the desired aesthetic direction, invokes an editing tool, observes the rendered intermediate image, and then adaptively determines the next action according to the updated visual state until the result satisfies the user intent. We equip *AdaRetouch* with three complementary types of tools: global lighting adjustment, global color grading, and local HSL adjustment. These tools cover common retouching scenarios and enable fine-grained editing in a coarse-to-fine manner from global corrections to local refinements.

To enable this, we construct a software-agnostic data pipeline for interleaved trajectories of instructions, tool calls, and visual feedback, and train *AdaRetouch* with SFT and RL. This provides denser supervision for closed-loop tool planning in practice. The RL stage leverages process-level rewards to improve editing efficiency, suppress redundant operations, and enhance final quality. Experiments on our benchmark show that *AdaRetouch* outperforms general MLLMs, retouching agents, and diffusion models in instruction alignment, perceptual quality, and tool planning.

Our contributions are summarized as follows:

- ① We construct a data generation pipeline for image retouching, producing interleaved image-text trajectories with multi-step reasoning and tool calls. We introduce a benchmark for evaluating vision-language retouching agents.
- ② We propose *AdaRetouch*, a vision-language retouching agent for multi-step tool invocation. Trained with SFT and RL, it leverages process rewards from intermediate image feedback to encourage step-wise improvement.
- ③ Experiments show that *AdaRetouch* outperforms general MLLMs, existing retouching agents, and diffusion-based models, achieving stronger user-intent understanding and more reliable, controllable editing behavior.

All data and source code associated with this work are open-sourced at <https://github.com/VesperInk/AdaRetouch>.

2 *AdaRetouch*

This section introduces *AdaRetouch*. We formulate the multi-turn retouching task and agent architecture in Sec. 2.2.1, describe the interleaved vision-language data construction pipeline in Sec. 2.1, and present the two-stage training paradigm with cold-start SFT and fine-grained RL in Sec. 2.2, highlighting how feedback enables adaptive, reliable, and controllable editing decisions.

2.1 Interleaved Trajectory Data Construction

Existing retouching datasets lack intermediate states, and random noise synthesis induces unnatural shifts. To overcome this, as shown in Figure 2, we construct trajectories through four stages: contextual degradation, trajectory optimization, instruction synthesis, and rationale annotation.

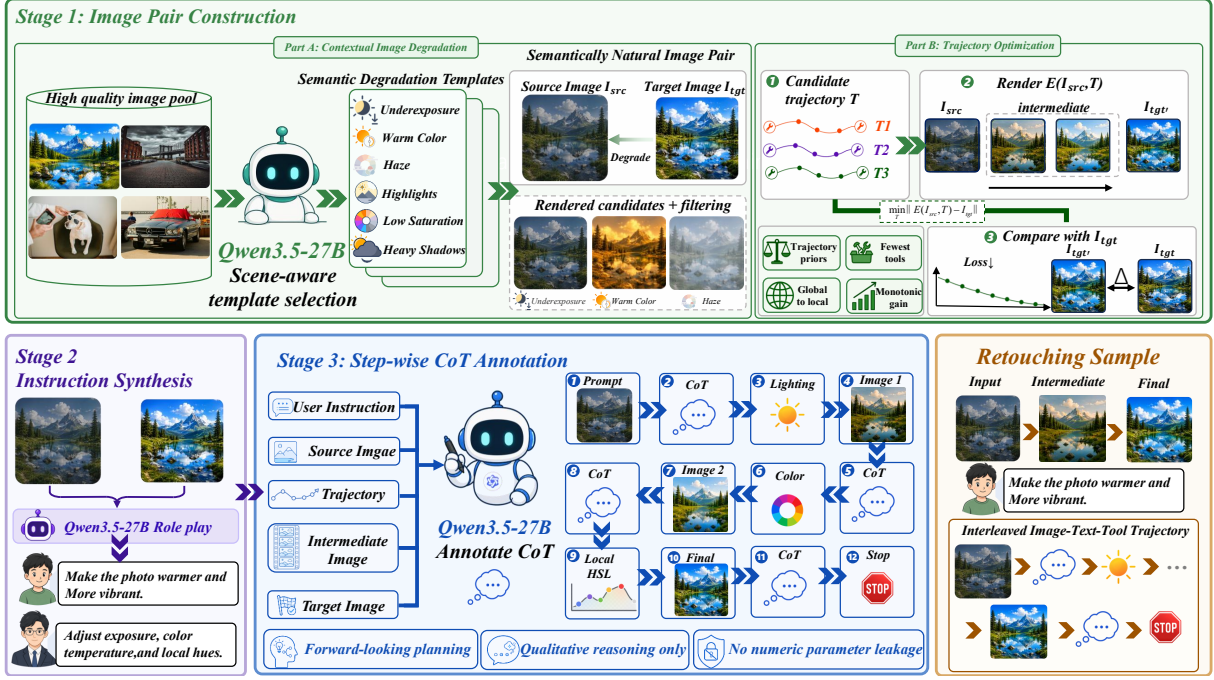


Figure 2: Overview of the interleaved vision-language trajectory data construction pipeline, including (1) *Contextual Image Degradation*, (2) *Trajectory Optimization*, (3) *Instruction Synthesis*, and (4) *Step-Wise CoT Annotation*, followed by a retouching sample illustrating the resulting interleaved image-text-tool trajectory.

① **Contextual Image Degradation.** We source target images I_{tgt} from PPR10K (Liang et al., 2021), Pexels, and licensed open-source resources. To synthesize unretouched source images I_{src} , we curate 66 semantic degradation templates defining rendering parameters and directions. Instead of arbitrary application, we utilize Qwen3.5-27B (Qwen Team, 2026) to analyze the scene semantics and visual style of each I_{tgt} and assign the three most contextually appropriate templates. Following rendering and filtering severely corrupted instances, we obtain semantically natural image pairs $\langle I_{src}, I_{tgt} \rangle$ with balanced distribution of real-world flaws.

② **Curriculum-Driven Trajectory Optimization.** Since photographic rendering is non-invertible, extracting editing actions requires optimizing a sequence of tool calls from I_{src} to I_{tgt} . We employ Gradient Descent to fit this sequence, guided by three professional principles: (1) *Curriculum Complexity*: We select the trajectory with the fewest necessary tool calls satisfying a reconstruction-quality criterion. (2) *Global-to-Local Ordering*: The sequence must prioritize global adjustments (e.g., `adjust_lighting`) before local refinements (e.g., `adjust_hsl`). (3) *Monotonicity*: Each tool must demonstrate a tangible visual gain. This optimization yields the discrete tool configuration trajectory denoted by the set $T = \{T_1, \dots, T_N\}$.

③ **Diverse Instruction Synthesis.** To simulate varied real-world requests, we prompt Qwen3.5-27B (Qwen Team, 2026) to translate target editing intents into user instructions U via role-playing. To enhance policy generalization, we generate instructions from two distinct perspectives: *laypersons* (expressing intuitive, jargon-free visual needs) and *experts* (providing technical, parameter-driven directives) to ensure comprehensive data coverage.

④ **Step-wise CoT Annotation.** We construct an interleaved vision-language CoT in three phases under realistic multi-turn editing constraints: (1) *Initial Planning*: the model analyzes the source image, instruction U , and trajectory T to form a global strategy. (2) *Intermediate Reasoning*: given past actions, CoTs, and intermediate images, it identifies remaining flaws and justifies the next operation. (3) *Termination*: it checks whether the result matches user intents before finally outputting the `<STOP_EDIT>`. See Appendix B for prompts.

The AdaRetouch dataset provides diverse interleaved trajectories covering broad real-world retouching scenarios. As shown in Figure 3, the collected data spans multiple domains (e.g., natural motifs, portraits, built environments). This diversity encourages the agent to learn robust editing policies across varied subjects, styles, and complex user intents. See Appendix C for dataset statistics.

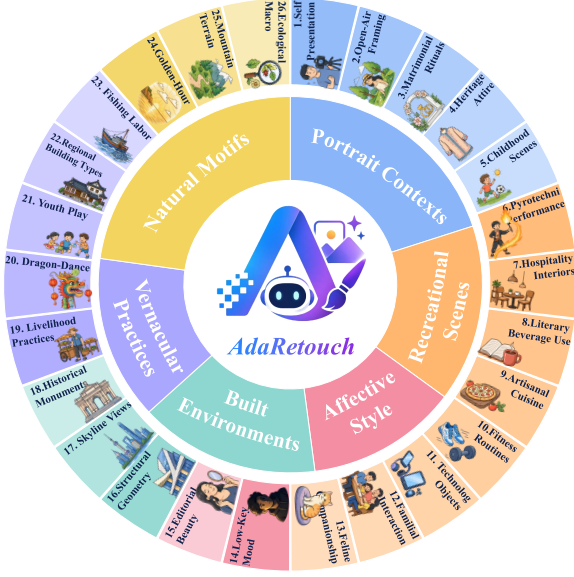


Figure 3: Distribution of the AdaRetouch dataset.

2.2 Adaptive Policy Learning via SFT and RL

2.2.1 Overview

At each step t , the agent policy π_θ predicts an action conditioned on the user instruction, trajectory history, and current image:

$$(c_t, \tau_t) \sim \pi_\theta(\cdot | U, H_{t-1}, I_t), \quad (1)$$

$$\tau_t \in \mathcal{A} \cup \{\text{<STOP_EDIT>}\}.$$

AdaRetouch formulates interactive image retouching as a Markov Decision Process (MDP). Given a user instruction U and an initial image I_0 , the state $s_t = \{U, H_{t-1}, I_t\}$ consists of the instruction, trajectory history, and current image at each step. Each action $a_t = (c_t, \tau_t)$ contains a textual Chain-of-Thought (CoT) c_t and a tool call τ_t .

After executing τ_t from action space $\mathcal{A}$, the rendering environment $E(\cdot)$ updates the visual state:

$$I_{t+1} = E(I_t, \tau_t). \quad (2)$$

This interleaved vision-language loop iterates adaptively until $\tau_t = \text{<STOP_EDIT>}$, yielding the final retouched image I_{edit} with improved quality. See Appendix A for tool and rendering details.

2.2.2 Cold-Start Supervised Fine-Tuning.

As shown in Figure 4, we first perform supervised fine-tuning (SFT) on the interleaved trajectory dataset to initialize a reliable editing policy before reinforcement learning. Each training instance contains the user instruction, source image, previous reasoning traces, executed tool calls, and intermediate rendered images. The model is optimized

to imitate the annotated next-step behavior, including both the textual CoT c_t and the executable tool call τ_t , so that it learns to couple visual diagnosis with grounded parameter prediction.

This stage provides step-wise supervision for three capabilities: *Context Adaptation*, which conditions decisions on interleaved histories and current visual states; *Intent Alignment*, which maps layperson and expert instructions to consistent editing goals; and *Action Grounding*, which learns tool selection, parameter prediction, global-to-local ordering, and proper termination with `<STOP_EDIT>`. Thus, SFT yields a stable multi-turn editing policy for subsequent fine-grained RL.

2.2.3 Fine-Grained RL Supervision.

Building on the SFT baseline, we apply Group Relative Policy Optimization (GRPO) (Shao et al., 2024) to refine planning and final editing quality. To reduce reward sparsity in the mixed continuous-discrete action space, we use a multi-dimensional reward covering format compliance, final quality, process monotonicity, and efficiency:

$$R_{\text{total}} = \begin{cases} \tilde{\Delta}_{\text{PSNR}} - P, & \text{if valid,} \\ R_{\text{gate}}, & \text{otherwise,} \end{cases} \quad (3)$$

where P penalizes process regression and wasteful operations, $\tilde{\Delta}_{\text{PSNR}}$ denotes clipped PSNR improvement, and R_{gate} is assigned to invalid outputs.

Format Penalty: A valid trajectory must correctly use `<think>...</think>` and `<tool_call>...</tool_call>` tags and restrict tool calls to the action space $\mathcal{A}$. Invalid trajectories receive $R_{\text{gate}} = -15$ to prevent malformed-output reward hacking.

Final Quality Reward: We measure terminal improvement by PSNR gain (Δ_{PSNR}), cap PSNR at $\tau = 30$, and clip the reward to $[c_{\min}, c_{\max}]$ to limit outliers and stabilize policy optimization dynamics.

Process Monotonicity Penalty: To provide process-level supervision, we penalize PSNR drops between adjacent intermediate images. Let p_t be the PSNR of the rendered image I_t against the target. The penalty is

$$P_{\text{mono}} = \alpha \sum_{t=1}^N [p_{t-1} - p_t]_+, \quad (4)$$

where $[x]_+ = \max(0, x)$ and α controls the penalty strength. This encourages monotonic improvement and reduces oscillatory edits.

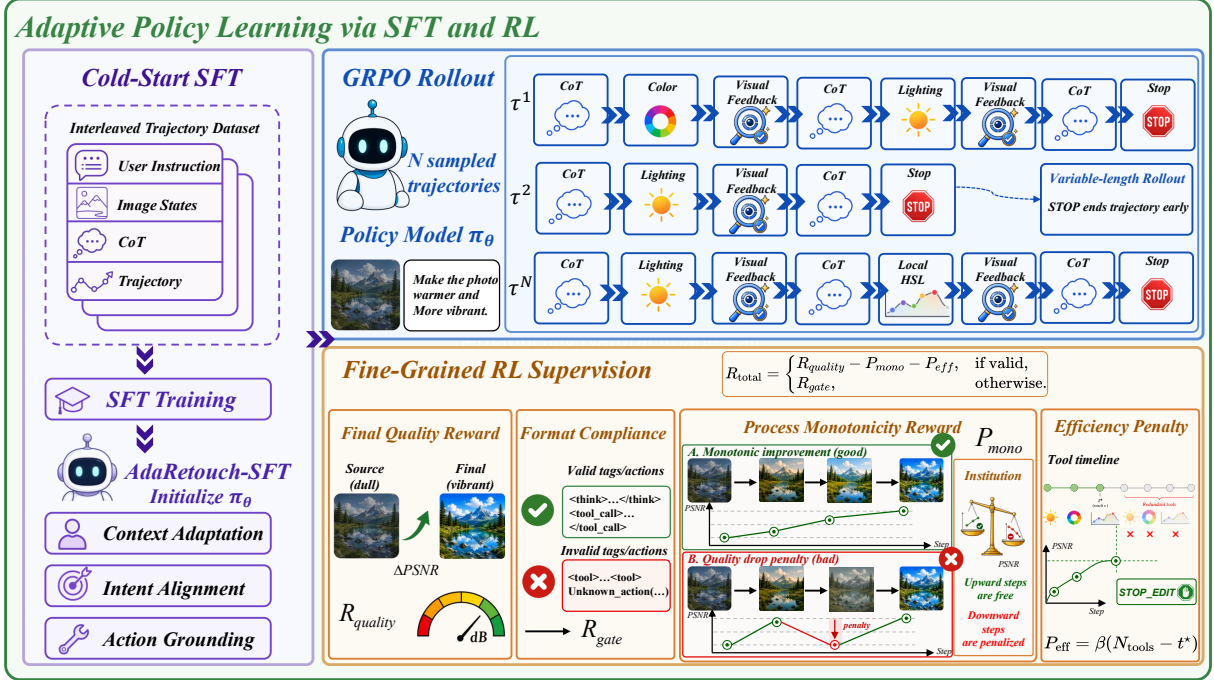


Figure 4: Overview of adaptive policy learning in AdaRetouch, including (1) Cold-start SFT initializes the policy from interleaved editing trajectories. (2) GRPO rollouts sample adaptive tool-use paths with visual feedback. (3) Fine-grained RL rewards optimize image quality, format compliance, process monotonicity, and efficiency.

Efficiency Penalty. We further discourage redundant tool calls after sufficient quality is reached. Let t^* be the first step satisfying $p_{t^*} \geq \tau$:

$$P_{\text{eff}} = \begin{cases} \beta(N_{\text{tools}} - t^*), & \text{if such } t^* \text{ exists,} \\ 0, & \text{otherwise,} \end{cases} \quad (5)$$

where β controls the efficiency penalty, favoring shorter trajectories when quality is comparable.

3 Experiments

3.1 Experimental Settings

Implementation Details. We initialize *AdaRetouch* from *Qwen2.5-VL-7B-Instruct* (Bai et al., 2025) and train it in two stages: supervised fine-tuning on 100K annotated samples followed by reinforcement learning on 40K samples. We use LLaMA-Factory for SFT and verl for RL (Zheng et al., 2024; Sheng et al., 2025). More specifics regarding the detailed training settings are provided in Appendix D.

Baselines. We compare *AdaRetouch* with three categories of baselines: (1) *MLLM-based retouching agents*, including *JarvisArt* (Lin et al., 2026), *JarvisEvo* (Lin et al., 2025), and *MonetGPT* (Dutt et al., 2025); (2) *zero-shot retouching agent loops*, which we build with general MLLMs, including *Gemini-3.1-Pro* (Google DeepMind, 2026b) and *GPT-5.4* (OpenAI, 2026), under a unified prompting

framework capped at 8 interaction rounds; and (3) *diffusion-based models*, including *BAGEL* (Deng et al., 2025), *Qwen-Image-Edit-2511* (Wu et al., 2025), *FLUX.2-Dev* (Labs, 2025), *FireRed-Image-Edit* (Team et al., 2026), *GPT-Image-1* (Hurst et al., 2024), as well as the newly proposed *Nano-Banana-2* (Google DeepMind, 2026a).

Benchmark and Evaluation Metrics. To evaluate the multi-step retouching ability of agents, we construct *AdaRetouch-Bench*, containing 1,200 test samples. The benchmark is evenly split into single-tool, two-tool, and three-tool retouching tasks, enabling us to assess whether a model can adaptively select an appropriate number of tool calls under different levels of editing complexity. This balanced distribution ensures a comprehensive evaluation across diverse real-world retouching scenarios.

We evaluate pixel-level fidelity using L1 and L2 distances between the retouched and reference images. The perceptual-distance metrics DISTS, GMSD, and TD additionally assess texture preservation and retouching consistency. For high-level perceptual evaluation, we employ Gemini-2.5-Flash (Comanici et al., 2025) to score instruction alignment as Semantic Consistency (SC) and visual coherence and artifacts as Perceptual Quality (PQ) on a 0–10 scale. The overall perceptual score O is their geometric mean: $O = \sqrt{SC \times PQ}$.

Method	Paradigm	$L_1 \times 10^2 \downarrow$	$L_2 \times 10^3 \downarrow$	DISTS $\downarrow$	GMSD $\downarrow$	TD $\downarrow$	SC $\uparrow$	PQ $\uparrow$	O $\uparrow$
JarvisArt (Lin et al., 2026)	Retouching agents	9.51	17.62	0.105	0.041	0.754	8.39	9.15	8.74
JarvisEvo (Lin et al., 2025)		8.86	15.73	0.081	0.056	0.648	8.21	9.03	8.43
MonetGPT (Dutt et al., 2025)		12.07	20.86	0.096	0.067	0.781	7.83	7.79	7.81
GPT-5.4 (OpenAI, 2026)	General MLLMs	8.31	12.40	0.059	0.034	0.402	8.64	6.56	7.62
Gemini-3.1-Pro (Google DeepMind, 2026b)		10.04	18.57	0.078	0.055	0.603	8.79	8.93	8.88
BAGEL (Deng et al., 2025)	Open-source Diffusion models	13.77	33.92	0.205	0.154	1.438	7.02	5.93	6.44
FLUX.2-Dev (Labs, 2025)		16.70	50.81	0.178	0.183	1.301	8.71	8.47	8.55
FireRed-Image-Edit (Team et al., 2026)		10.91	23.67	0.128	0.092	0.730	8.80	9.09	8.90
Qwen-Image-Edit-2511 (Wu et al., 2025)		17.08	54.65	0.169	0.135	0.875	8.81	8.02	8.21
GPT-Image-1 (Hurst et al., 2024)	Closed-source	27.20	123.89	0.193	0.193	1.542	8.68	8.04	8.31
Nano-Banana-2 (Google DeepMind, 2026a)	Diffusion models	12.21	29.69	0.085	0.065	0.752	8.85	9.10	8.94
AdaRetouch	Ours	3.96	3.55	0.029	0.015	0.380	9.01	9.35	9.22

Table 1: Quantitative comparison on AdaRetouch-Bench. indicates the best performance, and indicates the second-best performance.

3.2 Main Results

Table 1 presents the quantitative evaluation results on AdaRetouch-Bench. Overall, AdaRetouch consistently achieves state-of-the-art performance across all metrics, outperforming existing baselines. We summarize our key findings as follows:

① **Superior Intent Execution.** Compared to specialized retouching agents and general MLLMs, AdaRetouch achieves significantly lower L_1 and L_2 errors. This demonstrates that our cold-start SFT and fine-grained RL pipeline bridges the gap between natural language instructions and continuous parameter execution, enabling more precise intent alignment than fixed-route models.

② **Non-Destructive Editing.** While diffusion-based models can synthesize plausible visual results, their stochastic pixel-generation nature alters the original image layout, leading to severe pixel-level distortions and excessively high L_1 and L_2 errors. In contrast, AdaRetouch strictly employs non-destructive tool operations, preserving structural fidelity while securing higher Semantic Consistency (SC) and Perceptual Quality (PQ).

③ **Effective Adaptive Planning.** AdaRetouch attains the highest overall score (O). Guided by process-level rewards, our interleaved closed-loop framework can dynamically inspect intermediate visual feedback. This allows the agent to adaptively determine the optimal sequence and termination point, preventing the error accumulation and over-editing typical in open-loop pipelines.

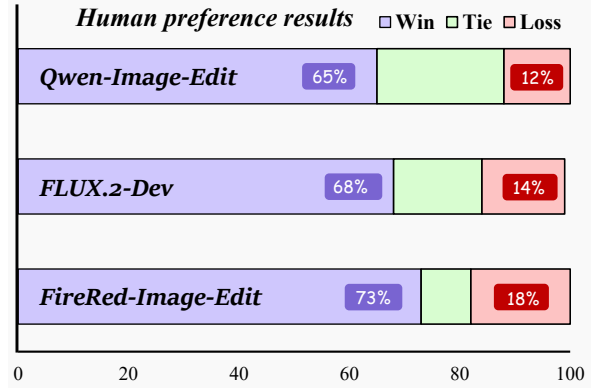


Figure 5: Human preference comparison between AdaRetouch and competing methods.

3.3 Human preference

Following JarvisEvo (Lin et al., 2025), we conduct a human preference study on 250 diverse test samples. We recruit 20 participants to compare AdaRetouch with Qwen-Image-Edit-2511, FLUX.2-Dev, and FireRed-Image-Edit. In each trial, participants are shown an input image, a retouching instruction, and two edited results in random order, and are asked to choose the better result in terms of instruction alignment and aesthetic quality, or mark a tie when both results are equally good.

As shown in Figure 5, AdaRetouch is consistently preferred over all baselines. It achieves win rates of 73.0%, 68.0%, and 65.0% against FireRed-Image-Edit, FLUX.2-Dev, and Qwen-Image-Edit-2511, respectively. These results indicate that AdaRetouch produces more instruction-faithful and visually appealing retouching results.

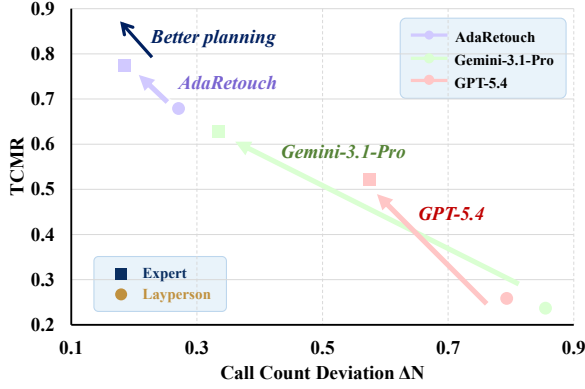


Figure 6: Comparison of call count deviation and tool call match rate under expert and layperson instructions.

3.4 Adaptive Tool Invocation Analysis

Beyond final image quality, a practical retouching agent should infer an appropriate editing trajectory, including when to stop and which tools to invoke. We conduct a process-level analysis on **AdaRetouch-Bench** at the tool-category level, ignoring continuous parameters. We report two metrics: call count deviation (ΔN), which measures step-count calibration, and tool call match rate (TCMR), which measures exact matching of the ordered tool sequence.

- **Trajectory analysis.** AdaRetouch achieves a low call count deviation of $\Delta N = 0.228$ and a high tool call match rate of $\text{TCMR} = 0.727$. These results indicate that AdaRetouch can estimate the required number of editing steps while recovering the reference tool order, suggesting a calibrated tool invocation policy.
- **Instruction robustness analysis.** We assess instruction robustness by measuring the straight-line distance between expert and layperson results in the $(\Delta N, \text{TCMR})$ space, where both instructions share the same ground-truth trajectory and shorter distances indicate stronger robustness. AdaRetouch achieves a distance of 0.129, indicating that it maps both technical and non-technical descriptions to similar practical editing plans.

Overall, these findings show that AdaRetouch learns reliable sequence planning and maintains stable tool invocation behavior across different user intents across a variety of practical scenarios.

3.5 Ablation Study

We ablate the two-stage training paradigm and RL reward design on AdaRetouch-Bench. Table 2(a)

(a) Training paradigm

Variant	L_1 $\times 10^2 \downarrow$	L_2 $\times 10^3 \downarrow$	SC $\uparrow$	PQ $\uparrow$	O $\uparrow$
SFT	4.61	4.80	8.86	9.15	9.04
SFT + RL	3.96	3.55	9.01	9.35	9.22

(b) RL reward component

Variant	Non-mono. traj. (%) $\downarrow$	Over-edit @30 (%) $\downarrow$
Full objective	14.42	10.70
w/o P_m	16.59	10.78
w/o P_e	15.38	11.71
w/o P_m, P_e	18.25	12.89

Table 2: Ablation study of (a) Training paradigms and (b) RL reward components.

shows consistent quantitative gains from RL fine-tuning over SFT. Since high-end perceptual refinements are disproportionately harder than coarse corrections, these gains indicate that RL exploration can thus effectively break the SFT ceiling.

For RL reward analysis (Table 2b), we evaluate two trajectory behaviors: **trajectory non-monotonicity** and **over-editing**. Removing P_m or P_e worsens the corresponding behavior, while the outcome-only variant degrades on both metrics. This confirms that process-oriented rewards improve step-wise stability and reduce redundant operations. For the sake of completeness, detailed metric definitions are provided in Appendix E.

3.6 Case Study

As shown in Fig. 7, MLLMs such as Gemini-3.1-Pro tend to over-edit the image, as seen in the second row. Diffusion-based methods show limited instruction following under complex instruction and may fail to preserve the original structure, causing identity or environment distortions, e.g., FireRed in the third row. Existing retouching agents often preserve content better, but their semantic understanding of user intent remains insufficient. In contrast, our method provides more professional and fine-grained retouching, better aligning with subtle user requirements, yielding superior visual outcomes.

4 Related Work

Instruction-Guided Image Retouching. This task adjusts tone, color, and illumination according to user intent while preserving photographic realism (Zhang et al., 2023; Fu et al., 2024; Sheynin et al., 2024). Early enhancement and

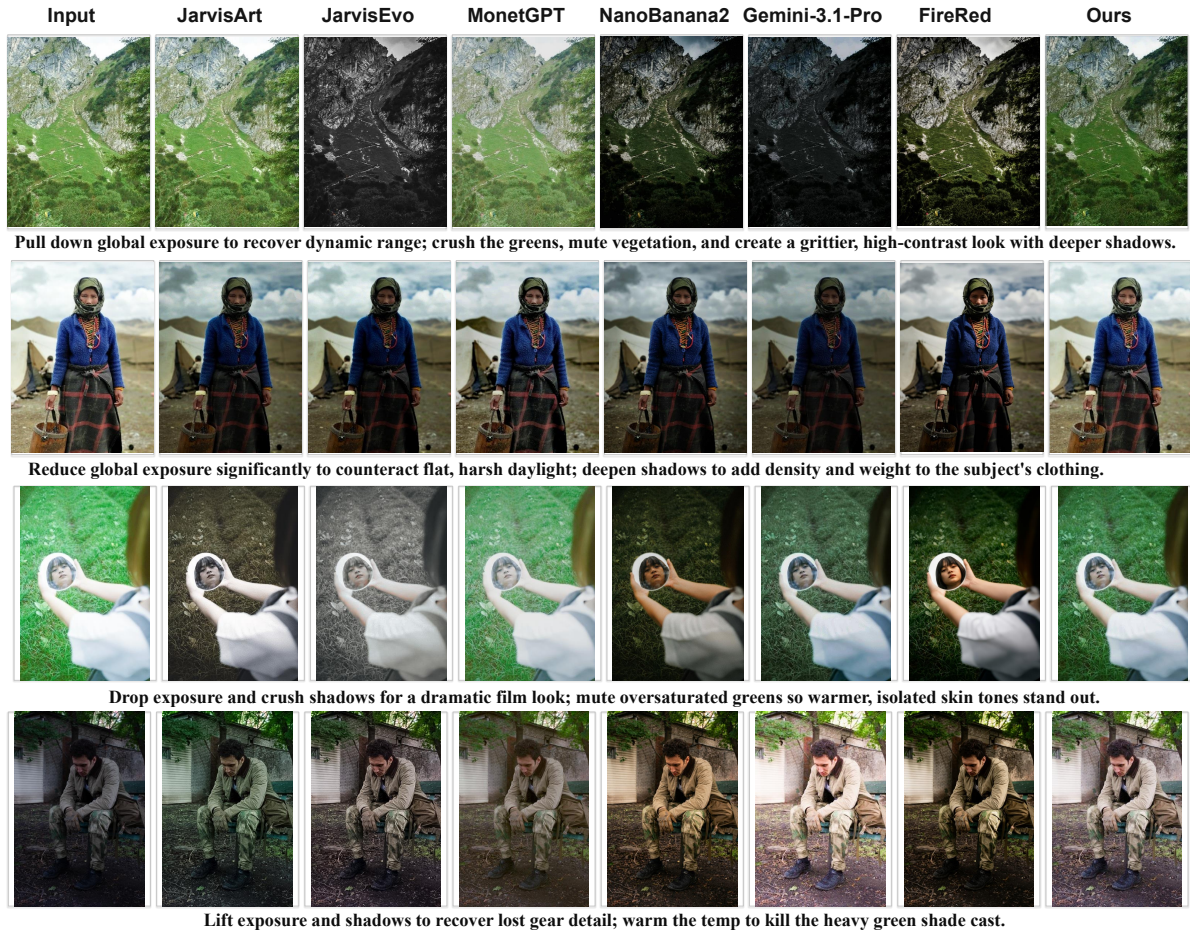


Figure 7: Case Study of visual comparisons across diverse instruction-guided retouching scenarios, showing that our method better follows user intent while preserving natural image content.

parametric methods learn from expert-retouched data or improve controllability with exposure adjustment, local filters, and image-adaptive LUTs (Yan et al., 2016; Chen et al., 2018; Wang et al., 2019; Moran et al., 2020; Zeng et al., 2020; Zhang et al., 2022). Diffusion-based editors offer flexible language control (Sheynin et al., 2024; Duan et al., 2025), but their outcome-supervised pixel generation can alter content and obscure editability. AdaRetouch instead uses non-destructive tool trajectories with real-time intermediate visual feedback loops.

MLLM-based Retouching Agents. Such agents build on multimodal tool-use systems with visual feedback (Yang et al., 2023). Early visual agents demonstrate how language models can coordinate external vision tools for perception, generation, and editing tasks (Wu et al., 2023; Yang et al., 2023). Recent agents connect MLLMs with procedural tools or professional software (Dutt et al., 2025; Chen et al., 2025; Lin et al., 2026). Reward-driven and iterative editors further improve alignment and

perceptual quality through self-evolution, learned rewards, or closed-loop refinement (Ye-Bin et al., 2025; Lin et al., 2025; Wu et al., 2026). However, existing agents are often software-coupled, open-loop, or weakly supervised at intermediate steps. AdaRetouch learns software-agnostic interleaved trajectories with process-level RL.

5 Conclusion

In this work, to overcome the limitations of rigid, open-loop editing frameworks, we present [AdaRetouch](#), an interleaved vision-language agent for adaptive multi-turn image retouching. It uses intermediate visual feedback to select tools and decide when to stop, enabling closed-loop control beyond fixed open-loop pipelines. With software-agnostic trajectories and SFT plus process-level RL, AdaRetouch achieves significantly stronger instruction alignment, perceptual quality, and tool planning than existing agents, general MLLMs, and diffusion editors on AdaRetouch-Bench.

Limitations

AdaRetouch currently operates with three types of retouching tools, covering common global lighting, global color, and local HSL adjustments. While effective for many scenarios, this tool set does not yet cover the full range of professional editing operations, such as masking, sharpening, denoising, or geometry-aware local refinement. In addition, the current policy always continues editing from the latest intermediate image. A more flexible agent could dynamically select any previous intermediate state as the starting point for further refinement, enabling better recovery from suboptimal steps and more exploratory editing trajectories. We leave broader tool integration and adaptive state selection for future work.

References

- Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibao Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, and 8 others. 2025. [Qwen2.5-vl technical report](#). *ArXiv*, abs/2502.13923.
- Vladimir Bychkovsky, Sylvain Paris, Eric Chan, and Frédo Durand. 2011. Learning photographic global tonal adjustment with a database of input/output image pairs. In *CVPR 2011*, pages 97–104. IEEE.
- Haoyu Chen, Keda Tao, Yizao Wang, Xinlei Wang, Lei Zhu, and Jinjin Gu. 2025. Photoartagent: Intelligent photo retouching with language model-based artist agents. *arXiv preprint arXiv:2505.23130*.
- Yu-Sheng Chen, Yu-Ching Wang, Man-Hsin Kao, and Yung-Yu Chuang. 2018. Deep photo enhancer: Unpaired learning for image enhancement from photographs with gans. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6306–6314.
- Gheorghe Comanici, Eric Bieber, Mike Schaeckermann, Ice Pasupat, Naveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, and 1 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.
- Tri Dao. 2024. Flashattention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations*, volume 2024, pages 35549–35562.
- Mauricio Delbracio, Damien Kelly, Michael S Brown, and Peyman Milanfar. 2021. Mobile computational photography: A tour. *Annual review of vision science*, 7(1):571–604.
- Chaorui Deng, Deyao Zhu, Kunchang Li, Chenhui Gou, Feng Li, Zeyu Wang, Shu Zhong, Weihao Yu, Xiaonan Nie, Ziang Song, and 1 others. 2025. Emerging properties in unified multimodal pretraining. *arXiv preprint arXiv:2505.14683*.
- Zheng-Peng Duan, Jiawei Zhang, Zheng Lin, Xin Jin, XunDong Wang, Dongqing Zou, Chun-Le Guo, and Chongyi Li. 2025. Diffretouch: Using diffusion to retouch on the shoulder of experts. In *Proceedings of the AAAI Conference on Artificial Intelligence*.
- Niladri Shekhar Dutt, Duygu Ceylan, and Niloy J Mitra. 2025. Monetgpt: Solving puzzles enhances mllms’ image retouching skills. *ACM Transactions on Graphics (TOG)*, 44(4):1–12.
- Tsu-Jui Fu, Wenze Hu, Xianzhi Du, William Wang, Yinfei Yang, and Zhe Gan. 2024. Guiding instruction-based image editing via multimodal large language models. In *International Conference on Learning Representations*, volume 2024, pages 54820–54833.
- Michaël Gharbi, Jiawen Chen, Jonathan T Barron, Samuel W Hasinoff, and Frédo Durand. 2017. Deep bilateral learning for real-time image enhancement. *ACM Transactions on Graphics (TOG)*, 36(4):1–12.
- Fazilet Gokbudak and A Cengiz Oztireli. 2023. One-shot detail retouching with patch space neural transformation blending. In *Proceedings of the 20th ACM SIGGRAPH European Conference on Visual Media Production*, pages 1–10.
- Google DeepMind. 2026a. [Gemini 3.1 Flash Image Preview](#).
- Google DeepMind. 2026b. [Gemini 3.1 Pro Model Card](#).
- Yuanming Hu, Hao He, Chenxi Xu, Baoyuan Wang, and Stephen Lin. 2018. Exposure: A white-box photo post-processing framework. *ACM Transactions on Graphics (TOG)*, 37(2):1–17.
- Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.
- Junjie Ke, Qifei Wang, Yilin Wang, Peyman Milanfar, and Feng Yang. 2021. Musiq: Multi-scale image quality transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5148–5157.
- Shu Kong, Xiaohui Shen, Zhe Lin, Radomir Mech, and Charless Fowlkes. 2016. Photo aesthetics ranking network with attributes and content adaptation. In *European conference on computer vision*, pages 662–679. Springer.
- Black Forest Labs. 2025. [FLUX.2: Frontier Visual Intelligence](#).

- Jie Liang, Hui Zeng, Miaomiao Cui, Xuansong Xie, and Lei Zhang. 2021. Ppr10k: A large-scale portrait photo retouching dataset with human-region mask and group-level consistency. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 653–661.
- Yunlong Lin, Zixu Lin, Kunjie Lin, Jinbin Bai, Panwang Pan, Chenxin Li, Haoyu Chen, Zhongdao Wang, Xinghao Ding, Wenbo Li, and 1 others. 2026. Jarvis-art: Liberating human artistic creativity via an intelligent photo retouching agent. *Advances in Neural Information Processing Systems*, 38:52088–52130.
- Yunlong Lin, Linqing Wang, Kunjie Lin, Zixu Lin, Kaixiong Gong, Wenbo Li, Bin Lin, Zhenxi Li, Shiyi Zhang, Yuyang Peng, and 1 others. 2025. Jarvisveto: Towards a self-evolving photo editing agent with synergistic editor-evaluator optimization. *arXiv preprint arXiv:2511.23002*.
- Pei Lv, Jianqi Fan, Xixi Nie, Weiming Dong, Xiaoheng Jiang, Bing Zhou, Mingliang Xu, and Changsheng Xu. 2021. User-guided personalized image aesthetic assessment based on deep reinforcement learning. *IEEE Transactions on Multimedia*, 25:736–749.
- Sean Moran, Pierre Marza, Steven McDonagh, Sarah Parisot, and Gregory Slabaugh. 2020. Deeplpf: Deep local parametric filters for image enhancement. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12826–12835.
- OpenAI. 2026. [GPT-5.4 Thinking System Card](#).
- Qwen Team. 2026. [Qwen3.5: Towards native multi-modal agents](#).
- Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 3505–3506.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.
- Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2025. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pages 1279–1297.
- Shelly Sheynin, Adam Polyak, Uriel Singer, Yuval Kirstain, Amit Zohar, Oron Ashual, Devi Parikh, and Yaniv Taigman. 2024. Emu edit: Precise image editing via recognition and generation tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8871–8879.
- Hossein Talebi and Peyman Milanfar. 2018. Nima: Neural image assessment. *IEEE transactions on image processing*, 27(8):3998–4011.
- Super Intelligence Team, Changhao Qiao, Chao Hui, Chen Li, Cunzheng Wang, Dejie Song, Jiale Zhang, Jing Li, Qiang Xiang, Runqi Wang, and 1 others. 2026. Firered-image-edit-1.0 technical report. *arXiv preprint arXiv:2602.13344*.
- Ruixing Wang, Qing Zhang, Chi-Wing Fu, Xiaoyong Shen, Wei-Shi Zheng, and Jiaya Jia. 2019. Underexposed photo enhancement using deep illumination estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6849–6857.
- Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, and 1 others. 2025. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*.
- Chenfei Wu, Shengming Yin, Weizhen Qi, Xiaodong Wang, Zecheng Tang, and Nan Duan. 2023. Visual chatgpt: Talking, drawing and editing with visual foundation models. *arXiv preprint arXiv:2303.04671*.
- Jiarui Wu, Yujin Wang, Lingen Li, Fan Zhang, and Tianfan Xue. 2024. Goal conditioned reinforcement learning for photo finishing tuning. *Advances in Neural Information Processing Systems*, 37:46294–46318.
- Qiucheng Wu, Jing Shi, Simon Jenni, Kushal Kafle, Tianyu Wang, Shiyu Chang, and Handong Zhao. 2026. Retouchiq: Mllm agents for instruction-based image retouching with generalist reward. *arXiv preprint arXiv:2602.17558*.
- Zhicheng Yan, Hao Zhang, Baoyuan Wang, Sylvain Paris, and Yizhou Yu. 2016. Automatic photo adjustment using deep neural networks. *ACM Transactions on Graphics (TOG)*, 35(2):1–15.
- Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Ehsan Azarnasab, Faisal Ahmed, Zicheng Liu, Ce Liu, Michael Zeng, and Lijuan Wang. 2023. Mm-react: Prompting chatgpt for multimodal reasoning and action. *arXiv preprint arXiv:2303.11381*.
- Moon Ye-Bin, Roy Miles, Tae-Hyun Oh, Ismail Elezi, and Jiankang Deng. 2025. Retouchllm: Training-free code-based image retouching with vision language models. *arXiv preprint arXiv:2510.08054*.
- Hui Zeng, Jianrui Cai, Lida Li, Zisheng Cao, and Lei Zhang. 2020. Learning image-adaptive 3d lookup tables for high performance photo enhancement in real-time. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(4):2058–2073.
- Fengyi Zhang, Hui Zeng, Tianjun Zhang, and Lin Zhang. 2022. Clut-net: Learning adaptively compressed representations of 3dluts for lightweight image enhancement. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 6493–6501.

Kai Zhang, Lingbo Mo, Wenhui Chen, Huan Sun, and Yu Su. 2023. Magicbrush: A manually annotated dataset for instruction-guided image editing. *Advances in Neural Information Processing Systems*, 36:31428–31449.

Yongheng Zhang, Xu Liu, Ruihan Tao, Qiguang Chen, Hao Fei, Wanxiang Che, and Libo Qin. 2025. ViT-CoT: Video-Text Interleaved Chain-of-Thought for Boosting Video Understanding in Large Language Models. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pages 5267–5276.

Yongheng Zhang, Ziang Liu, Jiaxuan Zhu, Shuai Wang, Xiangqi Chen, Haojing Huang, Jiayi Kuang, Siyu Chen, Ao Shen, Hao Wu, Qiufeng Wang, Qian-Wen Zhang, Junnan Dong, Wenhao Jiang, Ying Shen, Hai-Tao Zheng, Yinghui Li, Di Yin, Xing Sun, and Philip S. Yu. 2026a. *From Chatbot to Digital Colleague: The Paradigm Shift Toward Persistent Autonomous AI*. Preprint, arXiv:2606.14502.

Yongheng Zhang, Zhipeng Xu, Hao Wu, Yinghui Li, Di Yin, Xing Sun, and Philip S. Yu. 2026b. *Latent Visual Cache for Video Reasoning*. Preprint, arXiv:2607.02607.

Yongheng Zhang, Guang Yang, Ruihan Hou, Qiguang Chen, Ziang Liu, Xiaolong Liu, Manman Zhang, Yan-chao Hao, Zheng Wei, Hao Wu, Libo Qin, Peishan Dai, Yinghui Li, Di Yin, and Xing Sun. 2026c. *Thinking in Video: Can Video Generators Really Reason About the Real World?* Preprint, arXiv:2607.17523.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyang Luo. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 3: system demonstrations)*, pages 400–410.

Appendix

A Tool Space and Rendering Environment

Action-space overview. AdaRetouch exposes a compact, software-agnostic retouching action space with three callable tools: global lighting adjustment, global color grading, and selective HSL adjustment. All exposed arguments are integer-valued controls in the range $[-100, 100]$. During both data construction and agent evaluation, a tool call is represented as a JSON-style function call. The parameter descriptions are listed in Table 3, and the tool space is summarized in Table 4.

Rendering protocol. The rendering environment applies tool calls sequentially and cumulatively along a trajectory: the image produced after step t is used as the input state for step $t + 1$, so later

parameters operate on the already edited intermediate image rather than being independently applied to the original image.

B Instruction Synthesis and Step-wise CoT Annotation

This section provides the prompt templates used to construct instruction-following retouching trajectories. The annotation pipeline contains three components: (i) a shared agent system prompt that defines the executable tool-use protocol, (ii) a user-instruction synthesis prompt that converts paired images and recovered parameters into natural requests, and (iii) step-wise CoT construction prompts that annotate the reasoning before each tool invocation and the final stopping decision.

B.1 Agent System Prompt

The following system prompt defines the retouching agent interface. It is used not only for constructing AdaRetouch training records but also for evaluating general MLLM baselines, including Gemini-3.1-Pro and GPT-5.4, under the same zero-shot agent setting. This ensures that all models share the same interaction protocol: observe the current image, reason about the remaining editing need, output a valid tool call, receive the rendered intermediate image, and stop once the user intent is satisfied. The full prompt is shown in Figure 8.

B.2 User Instruction Synthesis

For each recovered source-target pair, we synthesize two types of user requests: layperson instructions and expert instructions. The synthesis model is given the source image, target image, and a structured parameter summary derived from the recovered tool trajectory. The parameter summary is used only as privileged annotation context, while the generated user instructions are required to describe the intended visual change in natural language without exposing exact parameter values. The full instruction synthesis prompt is shown in Figure 9 for complete reproducibility.

B.3 Step-wise CoT Construction

We annotate step-wise reasoning in three stages based on the synthesized user instruction and the corresponding optimized trajectory: (1) an initial prompt to diagnose the image and justify the first action, (2) an intermediate prompt to address remaining flaws, and (3) a final prompt for stopping decisions. Although the teacher model sees the

Tool	Parameter	Description
adjust_lighting	exposure	Adjusts the overall luminance of the image. Positive values brighten the entire scene, while negative values darken it.
	contrast	Increases or decreases the difference between light and dark areas. Positive values make darks darker and lights lighter, while negative values flatten the image.
	highlights	Targets brighter regions such as skies or bright light sources. Positive values brighten highlights, while negative values darken them to recover overexposed details.
	shadows	Targets darker regions. Positive values lift shadows to reveal hidden details, while negative values darken shadows for more depth.
	whites	Adjusts the absolute brightest pixels. Positive values push bright areas closer to pure white, while negative values reduce glare and intense brightness.
	blacks	Adjusts the absolute darkest pixels. Positive values gently lift blacks for a faded look, while negative values push dark pixels toward solid black.
adjust_color_grading	temperature	Adjusts warm/cool balance. Positive values add warmth, while negative values cool the image.
	tint	Adjusts green/magenta balance. Positive values shift toward magenta, while negative values shift toward green.
	vibrance	Intelligently adjusts muted colors while protecting skin tones and already saturated colors.
	saturation	Uniformly increases or decreases the vividness of all colors. Positive values make colors more intense, while negative values drain color globally.
adjust_hsl	hue	Rotates the hue of pixels in a selected color channel.
	saturation	Increases or decreases the vividness of a selected color channel.
	lightness	Brightens or darkens a selected color channel.

Table 3: Parameter descriptions for the AdaRetouch tool space. All parameters take integer values in $[-100, 100]$

oracle data, the generated first-person CoT avoids oracle information, target images, and exact parameters. These prompts are further detailed in Figures 10–12 for complete reference.

C Dataset Statistics

Table 5 reports the source image distribution used for dataset construction. The split contains a total of 70,764 images across 6 broad Level-1 categories and 26 fine-grained Level-2 categories. Percentages are computed over the full source pool. Fine-grained categories span portraits, domestic scenes, affective styles, built environments, cultural practices, and natural landscapes.

D Training Details

Supervised fine-tuning. We perform SFT on 100K interleaved trajectory samples using LLaMA-Factory (Zheng et al., 2024). From Qwen2.5-VL-7B-Instruct (Bai et al., 2025), we tune the language model while freezing the vision tower and multimodal projector, so optimization focuses on retouching-specific reasoning, tool invocation, parameter prediction, and stopping behavior.

Reinforcement learning. Starting from the SFT checkpoint, we use verl (Sheng et al., 2025) to run multi-turn GRPO (Shao et al., 2024) on 40K samples. Algorithm 1 summarizes the details. The

Tool	Scope	Description
adjust_lighting	Global	Adjusts the global lighting, contrast, and specific tonal ranges (shadows, highlights, whites, blacks) of the image.
adjust_color_grading	Global	Adjusts the global color temperature, tint, vibrance, and overall saturation of the image.
adjust_hsl	Selective	Performs selective color adjustments by modifying the Hue, Saturation, and Lightness of 8 individual color channels: Red, Orange, Yellow, Green, Cyan, Blue, Purple, and Magenta. This allows precise, targeted color editing, for example, shifting only the sky’s blue hue, desaturating only greens, or darkening only skin-tone oranges, without affecting other colors in the image.

Table 4: Tool space used by AdaRetouch.

Level-1	Level-2	Percentage
Situated Portrait Studies	Self-Presentation	5.7063%
	Open-Air Subject Framing	5.1368%
	Matrimonial Ritual	4.5927%
	Heritage Attire	3.0581%
	Childhood Scenes	1.4668%
Domestic Recreation Contexts	Pyrotechnic Performance	2.9167%
	Hospitality Interior	2.3600%
	Literary Beverage Use	2.1296%
	Artisanal Cuisine	2.0674%
	Kinetic Fitness Routines	1.7961%
	Technology Objects	1.5559%
	Familial Interaction	1.4160%
	Feline Companionship	0.9595%
Affective Style Imagery	Low-Key Mood	6.8552%
	Editorial Beauty	5.9324%
Built Environment Forms	Structural Geometry	6.3394%
	Skyline Views	5.5381%
	Monumental Historical	3.3463%
Vernacular Culture Practices	Livelihood practices	4.0105%
	Dragon-Dance	3.0990%
	Youth Play	2.6624%
	Regional Building Types	2.2116%
	Fishing Labor	2.0773%
Natural Landscape Motifs	Golden-Hour	8.6188%
	Mountain Terrain	7.1420%
	Macro Ecological	7.0050%

Table 5: Level-1 and Level-2 source-category distribution. Percentages are computed over the full image pool.

policy samples tool-use trajectories with rendered visual feedback and is optimized by rewards for format validity, final quality, process monotonicity, and efficiency. The core hyperparameters are summarized in Table 6.

Algorithm 1 Fine-Grained GRPO Supervision

Input: Policies $\pi_\theta, \pi_{\text{ref}}$; data $\mathcal{D}$; group size G

Output: Optimized policy π_θ

```

1: for each sample  $(I_{\text{src}}, I_{\text{tgt}}, U) \in \mathcal{D}$  do
2:   Sample  $G$  trajectories  $\{\tau_i\}_{i=1}^G$  from  $\pi_\theta$ 
3:   for each trajectory  $\tau_i$  do
4:     if  $\tau_i$  is invalid then
5:       Set  $R_i = R_{\text{gate}}$ 
6:     else
7:       Render images & compute PSNR sequence
8:        $\tilde{\Delta}_{\text{PSNR}} = \text{clip}(\Delta_{\text{sat}}, c_{\text{min}}, c_{\text{max}})$ 
9:        $R_i = \tilde{\Delta}_{\text{PSNR}} - P_{\text{mono}} - P_{\text{eff}}$ 
10:    end if
11:  end for
12:  Normalize rewards to get relative advantages
     $\{\hat{A}_i\}_{i=1}^G$ 
13:  Update  $\pi_\theta$  via GRPO loss regularized by  $\pi_{\text{ref}}$ 
14: end for
```

E Evaluation Protocols and Additional Analysis

E.1 Tool Invocation Additional Statistics

We evaluate tool invocation at the tool-category level, ignoring continuous parameters, to assess if the agent recovers the reference step count and ordered tool plan. For sample i , let N_i^{pred} and N_i^{gt} denote the predicted and ground-truth numbers of tool calls, and let τ_i^{pred} and τ_i^{gt} denote the two corresponding ordered tool-category sequences. We use the call count deviation:

$$\Delta N = \frac{1}{M} \sum_i |N_i^{\text{pred}} - N_i^{\text{gt}}|, \quad (6)$$

where lower values indicate better step-count calibration.

We define TCMR as an ordered tool-sequence exact match rate:

$$\text{TCMR} = \frac{1}{M} \sum_i \mathbf{1}[\tau_i^{\text{pred}} = \tau_i^{\text{gt}}]. \quad (7)$$

Here τ_i contains only the tool categories and their order, e.g., [lighting, color, hsl]. Thus, TCMR

Stage	Hyperparameter	Value / Setting
SFT	Training samples	100K
	Base model	Qwen2.5-VL-7B-Instruct
	learning_rate	1.0×10^{-5}
	warmup_ratio	0.1
	num_train_epochs	3.0
	Hardware	8 × A100 80GB
	Training stack	DeepSpeed ZeRO-3 (Rasley et al., 2020), FlashAttention2 (Dao, 2024)
RL	Training samples	40K
	learning_rate	1.0×10^{-6}
	train_batch_size	128
	total_epochs	2
	max_assistant_turns	5
	n	8
	max_prompt_length	4096
	max_response_length	20480
	max_tool	
	response_length	512
	Rollout backend	asynchronous vLLM with tool-agent loop
	Use KL Loss	False
	Hardware	16 × A100 80GB

Table 6: Core SFT/RL training settings.

does not compare continuous parameter values, but requires the complete ordered tool-category sequence to match exactly.

To quantify robustness across user-intent styles, we additionally compute the Euclidean distance between the expert and layperson results in the two-dimensional $(\Delta N, \text{TCMR})$ space:

$$d_{\text{intent}} = ((\Delta N_{\text{lay}} - \Delta N_{\text{exp}})^2 + (\text{TCMR}_{\text{lay}} - \text{TCMR}_{\text{exp}})^2)^{1/2}. \quad (8)$$

A smaller d_{intent} means that technical and non-technical prompts lead to more similar tool-planning behavior. Table 7 reports the detailed statistics corresponding to Figure 6.

E.2 Ablation Metric Definitions

For the reward ablation in Table 2(b), we evaluate process-level trajectory behavior using the rendered intermediate images. For a given sample i , let $p_{i,t}$ denote the Peak Signal-to-Noise Ratio (PSNR) between the rendered intermediate image at step t

and the reference image, with $p_{i,0}$ corresponding to the unedited source image.

Trajectory non-monotonicity. This metric measures the percentage of predicted trajectories that exhibit at least one quality regression between adjacent editing steps. Formally, a trajectory i is counted as non-monotonic if there exists any step t such that $p_{i,t} < p_{i,t-1}$, reflecting instability.

Over-edit@30. This metric computes the percentage of trajectories that continue editing after achieving a satisfactory quality threshold. Specifically, a trajectory is flagged if the model performs any editing step $t > t^*$, where t^* is the first step satisfying $p_{i,t^*} \geq 30$, indicating redundant modifications.

F Detailed Trajectory Case Studies

Figures 13 and 14 provide detailed trajectory case studies, visualizing the user intent, intermediate observations, selected tool sequence, and step-wise visual changes for qualitative inspection, demonstrating how the agent dynamically adjusts its strategy based on visual feedback during editing.

Model	Exp. $\Delta N \downarrow$	Exp. TCMR $\uparrow$	Lay. $\Delta N \downarrow$	Lay. TCMR $\uparrow$	Dist. $\downarrow$
AdaRetouch	0.1850	0.7750	0.2709	0.6789	0.1289
Gemini-3.1-Pro	0.3350	0.6283	0.8550	0.2367	0.6510
GPT-5.4	0.5750	0.5217	0.7933	0.2583	0.3421

Table 7: Detailed tool invocation statistics under expert and layperson instructions. Intent distance is computed in the $(\Delta N, \text{TCMR})$ space between the two instruction styles.

Agent System Prompt

You are a professional AI photo retouching assistant. Your goal is to help users improve their photos by calling image-editing tools in a thoughtful, step-by-step manner.

Workflow

Given an input photo and a user's retouching request, you will:

1. Analyze the image and the user's intent.
2. Decide which tool to call and with what parameters.
3. After each tool is applied, you will receive the updated photo as visual feedback.
4. Repeat until the image fully satisfies the user's intent, then stop.

Output Format

Before each tool call, output your reasoning inside a `<think>` block, then immediately output the tool call:

`<think>`

Your internal reasoning about the current image state, what problems remain, and why you are choosing this specific tool and parameter directions.

`</think>`

`<tool_call>`

`{"name": "<tool_name>", "arguments": {<parameters>}}`

`</tool_call>`

After the final tool has been applied and the image is satisfactory, output your final self-review inside a `<think>` block, then output `<STOP_EDIT>` on a new line to signal that editing is complete:

`<think>`

Your reflection on the specific visual improvements achieved across all steps, and why no further edits are needed without risking over-editing.

`</think>`

`<STOP_EDIT>`

Rules

- Only call tools from the provided tool list.
- Stop editing as soon as the user's intent is satisfied; do not over-edit.

Figure 8: Agent system prompt used for AdaRetouch training and zero-shot general MLLM baseline evaluation.

User Instruction Synthesis Prompt

You are a professional image editing intent analysis expert. Your core task is to analyze the provided source image, target image, and editing parameters to infer and simulate user photo editing intentions.

Reference Information

To help you perceive subtle differences, the exact parameter adjustments applied to Image 1 to get Image 2 are provided below.

Two types of adjustments may be present:

- Global adjustments (adjust_lighting, adjust_color_grading): affect the entire image uniformly, including overall brightness, contrast, tone mapping, color temperature, and saturation.
- Local/selective adjustments (adjust_hsl): target specific color ranges only, such as skin tones, sky, foliage, or water.

Applied Adjustments:

{{PARAM_SUMMARY}}

Analysis and Generation Workflow

Step 1. Visual Change Analysis:

Analyze the visual changes made from Image 1 to Image 2. Use the applied adjustments as clues to locate specific changes in the image content.

Step 2. Intent Inference:

Infer the user's photo editing instruction or overall intent for this transformation.

Step 3. Generate User Intents:

Generate two final user prompts.

Layperson Intents:

- Generate one casual sentence from the perspective of an ordinary user.
- Focus on mood, aesthetics, or the subject's appearance.
- Do not mention technical parameters.
- Keep it under 25 words.

Expert Intents:

- Generate one sentence from the perspective of an art director or lead photographer.
- Reveal the professional rationale, such as tonal separation, color grading, highlight recovery, or skin protection.
- Do not mention exact numerical values.
- Keep it conversational and concise.

Strict Output Format

Output a valid JSON object:

```
{
  "visual_change_analysis": "...",
  "intent_inference": "...",
  "layperson_intents": ["..."],
  "expert_intents": ["..."]
}
```

Figure 9: Prompt for synthesizing layperson and expert user instructions from paired images and recovered tool parameters.

Initial CoT Prompt

You are a Master Photo Retoucher acting as a Teacher to generate training data for an AI Agent. Your task is to write the Agent's inner monologue before the first tool action.

Oracle information for annotation only:

1. The original image before editing.
2. The user's retouching intent.
3. The full tool sequence and the exact parameters for Step 1.
4. The image rendered after Step 1.
5. For multi-step trajectories, the best result after all steps.

Writing rules:

1. Write a natural first-person inner monologue as if the agent is planning right now.
2. Briefly internalize the user's intent and identify the main visible issues in the original image.
3. Explain why the first tool is the right priority and how it sets up later steps.
4. Translate the parameter direction into qualitative language, such as "soften the highlights", "add gentle warmth", or "reduce the overly vivid blues".
5. Do not use exact numerical values.

Negative constraints:

- Do not write retrospectively.
- Do not mention the target image, final image, ground truth, oracle, or dataset.
- Do not mechanically repeat parameter values.

Figure 10: Prompt for annotating the initial reasoning before the first tool invocation.

Intermediate CoT Prompt

You are a Master Photo Retoucher acting as a Teacher to generate training data for an AI Agent. Your task is to write the Agent's inner monologue before the current intermediate step.

Oracle information for annotation only:

1. The full trajectory history so far, including previous CoTs, previous tool calls, and rendered images.
2. The current step number, current tool, and current parameters.
3. The image rendered after the current step.
4. The best result after all steps.

Writing rules:

1. Act as if the agent is looking at the most recent rendered image and deciding what to do next.
2. Acknowledge what the previous step improved on specific visual elements.
3. Identify the remaining flaws that still prevent the image from matching the user's intent.
4. Explain why the current tool is now appropriate.
5. For `adjust_hsl`, explain which specific color regions need selective refinement and why another global adjustment is insufficient.
6. Use qualitative parameter descriptions only.

Negative constraints:

- Do not copy previous CoT steps.
- Do not mention the target image, final image, ground truth, oracle, or dataset.
- Do not use exact parameter values.

Figure 11: Prompt for annotating intermediate reasoning conditioned on previous actions and rendered visual feedback.

Final Self-Review Prompt

You are an AI Photo Retouching Agent performing a final self-review.
You have finished executing all tools and are looking at the final edited result.

Writing rules:

1. In first person, describe how specific visual elements improved from the original input to align with the user's intent.
2. If selective color adjustments were used, describe how specific color zones benefited from targeted per-channel treatment.
3. Confirm that the multi-step strategy is coherent.
4. Judge whether meaningful visual issues remain.
5. Explain why further editing would bring little benefit or risk over-editing.
6. Do not mention exact parameter values, datasets, ground truth, or oracle information.

Output exactly:

One short first-person critique paragraph.

<STOP_EDIT>

Figure 12: Prompt for annotating the final self-review and stopping decision.

Instruction

The photo looks a bit washed out; can you darken the background trees and make the straw look like rich gold instead of pale yellow?



Output

ROUND 1

<think>The photo currently suffers from a lack of depth and vibrancy, appearing somewhat flat and washed out under the bright light. To address the user's request for darker background trees and richer straw, I need to first establish a stronger tonal foundation. My priority is to correct the global lighting to remove that hazy quality before targeting specific colors. I will start by reducing the overall exposure to bring down the brightness levels, which should immediately add weight to the image... **</think>**

<tool_call>{"name":"adjust_lighting","arguments":{"exposure":-16,"contrast":32,"highlights":-6,"shadows":-6,"whites":-3,"blacks":-7}} **</tool_call>**



ROUND 2

<think>The global lighting adjustments have successfully removed the washed-out haze and added necessary weight to the shadows, particularly in the background foliage. However, the straw still lacks the specific richness requested; it appears somewhat pale and beige rather than a deep, vibrant gold. Additionally, while the trees are darker, their color could be more distinctively forest-like to maximize contrast against the subject. Since the overall exposure and contrast are now balanced... **</think>**

<tool_call>{"name":"adjust_hsl","arguments":{"red":{"hue":-6,"saturation":7,"lightness":-6},"orange":{"hue":-6,"saturation":27,"lightness":-20},"yellow":{"hue":-4,"saturation":24,"lightness":-29},"green":{"hue":-3,"saturation":17,"lightness":-28},"cyan":{"hue":-6,"saturation":10,"lightness":-20},"blue":{"hue":-6,"saturation":13,"lightness":-20},"purple":{"hue":-6,"saturation":6,"lightness":-6},"magenta":{"hue":-6,"saturation":6,"lightness":-6}}}} **</tool_call>**



FINAL

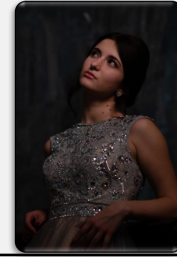
<think>I am satisfied with how the final image has transformed from its original washed-out state. By first establishing a stronger tonal foundation, I was able to remove the haze and give the scene necessary weight. The subsequent selective color adjustments were crucial here; targeting the yellow and orange channels allowed me to deepen the straw into a rich, harvest-gold hue without affecting the rest of the palette, while simultaneously darkening the greens made the background trees recede effectively to create depth. This two-step approach worked logically, moving from global lighting correction to precise color grading. At this stage, the balance between the warm subject and the cool, dark background feels natural, and pushing the saturation or contrast further would likely introduce an artificial look or lose texture in the shadows, so I believe the edit is complete. **</think>**

<STOP_EDIT></STOP_EDIT>

Figure 13: Detailed trajectory case study showing the complete multi-turn retouching process for a representative example with the tool sequence *adjust_lighting* -> *adjust_hsl*.

Instruction

Let's drastically lift the exposure and crush the contrast to flatten the dynamic range, then push the white balance significantly warmer to create a soft, dreamy aesthetic while ensuring the specular highlights on the beads don't clip.

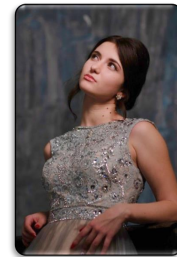


Output

ROUND 1

<think>The current image is quite dark with deep shadows obscuring the texture of the dress and the background, which contradicts the desired soft, dreamy aesthetic. To achieve this, I need to fundamentally alter the lighting structure first. My priority is to drastically lift the overall exposure to bring out the details in the subject's face and the intricate beadwork on her dress. Simultaneously, I must crush the contrast significantly to flatten the dynamic range, removing the harsh separation between light and dark... </think>

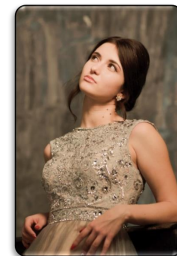
<tool_call>{"name":"adjust_lighting","arguments":{"exposure":30,"contrast":-41,"highlights":-6,"shadows":30,"whites":-4,"blacks":36}}</tool_call>



ROUND 2

<think>The previous adjustments have successfully lifted the exposure and flattened the contrast, bringing out the intricate beadwork and softening the shadows as intended. However, the image currently retains a somewhat neutral, slightly cool tone that feels too clinical for the desired dreamy aesthetic. To complete the transformation, I need to shift the color palette significantly. It is now the right moment to apply a strong warm shift to the white balance, infusing the scene with a golden, romantic glow that complements the soft lighting...</think>

<tool_call>{"name":"adjust_color_grading","arguments":{"temperature":20,"tint":10,"vibrance":11,"saturation":-4}}</tool_call>



FINAL

<think>I am satisfied with how the drastic exposure lift has revealed the intricate textures of the beaded bodice, which were previously lost in shadow, while the crushed contrast has successfully flattened the dynamic range to create that requested soft, ethereal quality. The significant shift toward warmer tones has transformed the mood from clinical to romantic, giving the skin a gentle, golden glow that harmonizes perfectly with the now-subdued background. My strategy of addressing the lighting structure before applying the color grade proved effective, ensuring the warmth didn't exacerbate any remaining harshness. At this stage, the image fully embodies the intended dreamy aesthetic; pushing the brightness or warmth further would likely introduce an unnatural orange cast or wash out the delicate details of the dress, so I consider this edit complete.</think>

<STOP_EDIT></STOP_EDIT>

Figure 14: Detailed trajectory case study showing the complete multi-turn retouching process for a representative example with the tool sequence *adjust_lighting* -> *adjust_color_grading*.