
JarvisArt: Liberating Human Artistic Creativity via an Intelligent Photo Retouching Agent -Supplementary Material-

Yunlong Lin^{1*} Zixu Lin^{1*} Kunjie Lin^{1*} Jinbin Bai⁵ Panwang Pan⁴ Chenxin Li³
Haoyu Chen² Zhongdao Wang⁶ Xinghao Ding^{1†} Wenbo Li^{3♣} Shuicheng Yan^{5♠*}

¹ Key Laboratory of Multimedia Trusted Perception and Efficient Computing,
Ministry of Education of China, Xiamen University, Xiamen, Fujian, China

² The Hong Kong University of Science and Technology (Guangzhou)

³ The Chinese University of Hong Kong ⁴ Bytedance

⁵ National University of Singapore ⁶ Tsinghua University

Project Page: <https://jarvisart.vercel.app/>

This is the supplementary material for the paper: “JarvisArt: Liberating Human Artistic Creativity via an Intelligent Photo Retouching Agent.” We provide the following materials in this manuscript:

- Sec.A Details of the Agent-to-Lightroom Protocol.
- Sec.B More User Studies.
- Sec.C Implementation Details.
 - Baselines Details.
 - Human-region Masks for Localized Evaluation.
- Sec.D More Experimental Results.
 - Editing Results under Complex Prompt.
 - Comparison on MIT-FiveK.
- Sec.E Broad Impact.

A Details of the Agent-to-Lightroom Protocol

The Agent-to-Lightroom protocol is designed to streamline communication between JarvisArt and Adobe Lightroom, supporting the automation of image editing tasks. It defines the communication framework between the agent and Lightroom, allowing the agent to access multiple retouching tools within Lightroom. This enables flexible, non-destructive photo retouching. The protocol workflow includes five stages: 1) handshake, 2) file verification, 3) sandboxed execution, 4) asynchronous processing, and 5) result return. A detailed explanation of each stage follows:

Handshake and connection management. The client initiates a TCP connection to the Lightroom Lua plugin server. A "ping" message confirms the server's responsiveness, and the server replies with "pong." TCP keepalive options ensure connection stability, and the server manages connection lifecycle through callbacks, including handling reconnections.

File verification. Before processing, the agent verifies the existence of the target photo and retouching operation configuration files using standard OS functions. The Lua server performs additional verification using Lightroom's built-in file utilities to ensure valid files are processed.

Sandboxed execution. The Lua server operates within Lightroom's sandboxed environment, ensuring that all operations (e.g., catalog access, photo import, development settings, and export) are safely handled through official APIs, limiting access to Lightroom-specific tasks.

* Equal Contributions. ♣ Project Leader † Corresponding Author. ♠ Supervised the research.

Asynchronous processing. To maintain responsiveness, the client uses a dedicated thread to process server responses asynchronously. The Lua server delegates photo processing tasks to Lightroom’s background task system, allowing immediate feedback to the client without waiting for completion.

Result return and feedback. The server sends structured status messages (e.g., success, error, in progress) to the client, which outputs updates accordingly. The client’s HTTP server returns structured JSON responses (e.g., "status": "success") or error codes.

B More User Studies

To evaluate the effectiveness and usability of JarvisArt, we recruited 30 participants from diverse backgrounds, including postgraduate students, artists, and computer vision researchers. All participants had prior experience with image editing, covering a broad spectrum of skill levels to ensure a realistic representation of user proficiency. To mitigate learning effects, participants were randomly assigned to two groups: Group A used JarvisArt before Lightroom, while Group B used Lightroom first and then JarvisArt. Each participant completed a comprehensive evaluation consisting of 10 questions per system, covering four key categories: *Complexity and Efficiency*, *Consistency and Integration*, *Ease of Use*, and *Overall Satisfaction*. Detailed evaluation results are shown in Figure 2. The main findings are summarized as follows:

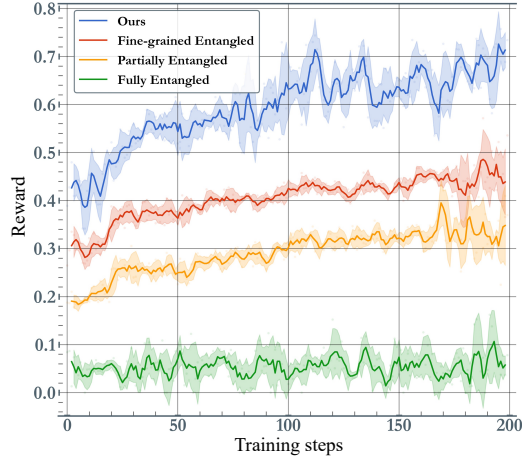


Figure 1: ROA reward trends across training steps for JarvisArt with different reward granularity.

- **Ease of use:** All participants rated JarvisArt as easy to use (Q1, score ≥ 3), with 66.7% awarding the highest score of 5. For independent operation and learning speed (Q2–Q4), over 90% of users rated JarvisArt 4 or 5, indicating that most users could quickly and independently learn to use the system with minimal need for technical support.
- **Complexity and efficiency:** Regarding system complexity (Q5), more than 96.67% of participants (score ≥ 3) considered JarvisArt appropriately complex, in contrast to Lightroom, which was frequently perceived as overly complicated. Additionally, 86.67% of users rated JarvisArt 4 or 5 for smoothness of use (Q6), suggesting that our design effectively reduced cognitive load and facilitated efficient task completion.
- **Consistency and integration:** For feature integration (Q7), 90% of users rated JarvisArt 4 or 5, and for system consistency (Q8), 93.3% did so, both significantly higher than Lightroom (16.67% and 40%, respectively). These results indicate a more cohesive and intuitive user experience with JarvisArt.
- **Overall satisfaction:** In terms of willingness to use in the future (Q9), 93.33% of users rated JarvisArt 4 or 5, and for confidence in use (Q10), 90% also gave high scores, both outperforming Lightroom (43.3% and 86.7%, respectively). This demonstrates strong user satisfaction and acceptance of JarvisArt.

C Implementation Details

C.1 Baselines Details

Our evaluated baselines are divided into two categories: specific photo retouching methods and instruction-based editing methods. The former category includes models that automatically transform source images into a target domain, such as 3DLUT [10] and RSFNet [6]. The latter category consists of models that perform editing based on user-provided instructions, including open-source methods like InstructPix2Pix [1], MagicBrush [11], OmniGen [9], VARGPT-v1.1 [12], and Step1X-Edit [5], as well as closed-source models such as GPT-4o [3] and Gemini-2-Flash [8]. For all baseline models,

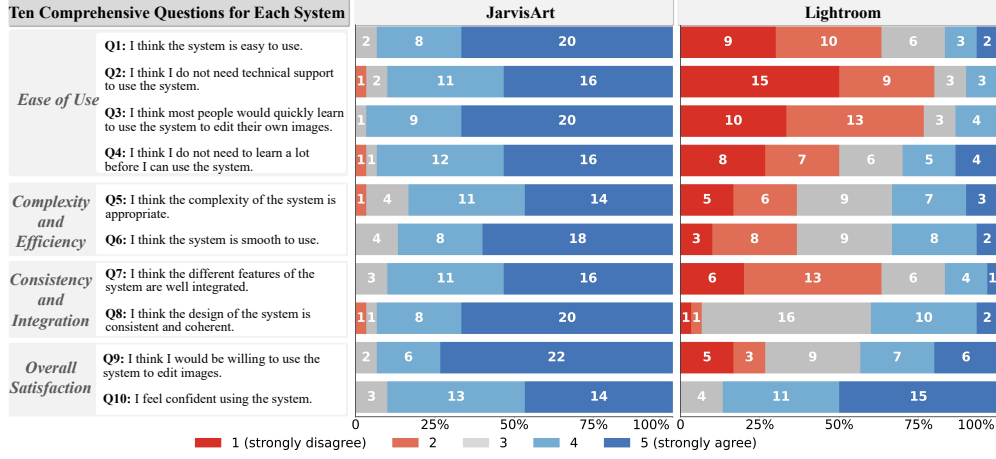


Figure 2: Questionnaire results and user ratings comparing JarvisArt with the commercial Adobe Lightroom system. Ratings are on a 5-point Likert scale (1=strongly disagree, 5=strongly agree).

Table 1: Configuration Details of Instruction-driven Baseline Models.

| Method | Configuration |
|---------------------|---|
| InstructPix2Pix [1] | num_inference_steps=10, image_guidance_scale=1 |
| MagicBrush [11] | seed=42, guidance_scale=7, num_inference_steps=20, image_guidance_scale=1.5 |
| OmniGen [9] | guidance_scale=2.5, img_guidance_scale=1.6, seed=0, num_inference_steps=50 |
| VARGPT-v1.1 [12] | cfg=3, tau=0.5, cfg_sc=3, top_k=900, top_p=0.97, gumbel=0, cfg_exp_k=0.0, softmax_merge_topk = -1, gt_leak=0, sampling_per_bits=1 |
| Step1X-Edit [5] | cfg_guidance=6.0, num_steps=28, num_samples=1, seed=42, max_length=640, image2image_strength=0.0, timesteps_truncate=1.0 |

we strictly followed the default hyperparameters specified in their official repositories or Huggingface implementations, as detailed in Table 1.

C.2 Human-region Masks for Localized Evaluation

In Figure 3, we present the high-resolution human-region masks provided by our MMArt-Bench dataset, which is derived from SMA2 [7] and PPR10K [4]. These masks are specifically designed for the calculation of region-level metrics, enabling precise evaluation of editing performance on specific regions. This facilitates a more fine-grained and targeted assessment, especially in scenarios where distinct editing effects are required for different regions within an image.

D Supplementary Experimental Results

D.1 Editing Results under Complex Prompt

Our system empowers users to perform complex retouching instructions (typically 40-word elaborative descriptions) to realize their creative intents. As illustrated in Figures 4 and 5, our approach exhibits strong instruction-following capabilities for complex editing tasks, even when confronted with ambiguous or intricately structured directives. This demonstrates the method’s effectiveness and generalizability.

Table 2: Quantitative evaluation on MIT-FiveK [2]. We highlight the best and second-best instruction-based results. SC, PQ, and O refer to the metrics evaluated by Gemini-2-Flash.

| Method | Instruction | $L1 \times 10^2 \downarrow$ | $L2 \times 10^3 \downarrow$ | SC \uparrow | PQ \uparrow | O \uparrow |
|---------------------|-------------|-----------------------------|-----------------------------|---------------|---------------|--------------|
| InstructPix2Pix [1] | ✓ | 16.23 | 49.54 | 6.36 | 8.34 | 7.15 |
| MagicBrush [11] | ✓ | 17.29 | 53.45 | 4.92 | 5.50 | 4.95 |
| OmniGen [9] | ✓ | 28.53 | 128.59 | 3.12 | 2.48 | 2.57 |
| VARGPT-v1.1 [12] | ✓ | 26.96 | 117.16 | 2.94 | 2.00 | 2.29 |
| Step1X-Edit [5] | ✓ | 22.08 | 91.72 | 7.20 | 8.48 | 7.69 |
| Gemini-2-Flash [8] | ✓ | 18.69 | 61.27 | 7.86 | 9.22 | 8.47 |
| GPT-4o [3] | ✓ | 21.49 | 78.11 | 8.72 | 9.76 | 9.22 |
| JarvisArt | ✓ | 12.98 | 30.05 | 7.36 | 9.82 | 8.48 |

D.2 Comparison on MIT-FiveK

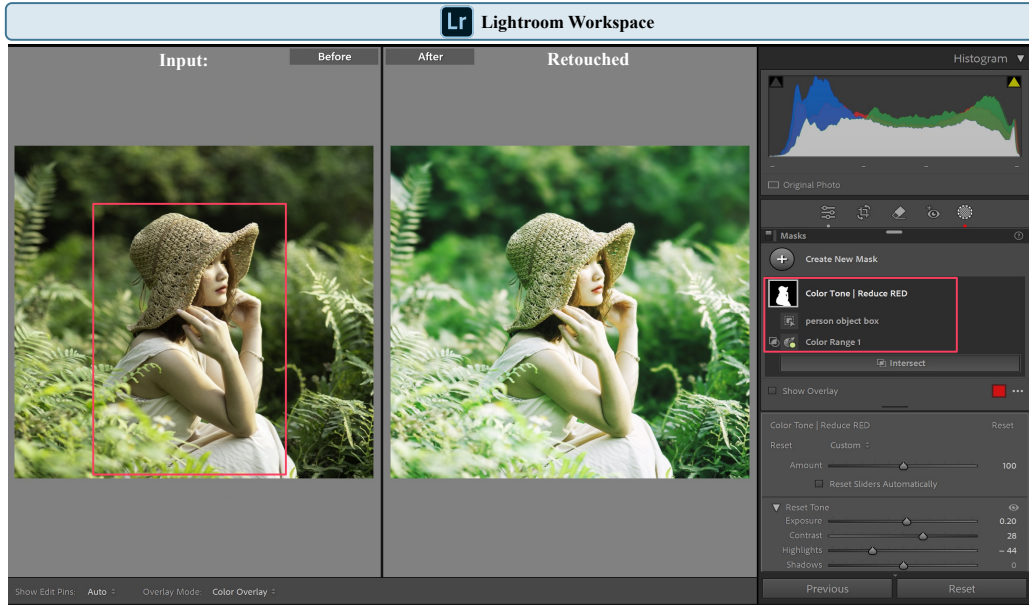
To assess the generalization ability of our system, we conduct comprehensive qualitative and visual comparisons on the MIT-FiveK [2] benchmark dataset. Specifically, we randomly select 50 data samples from MIT-FiveK [2] and generate corresponding user instructions based on the source images, expert C’s reference images, and the associated retouching parameter configurations (see Section 3.2 for implementation details). As shown in Table 2 and Figures 6 and 7, our system achieves state-of-the-art performance in both instruction-following and content preservation metrics. These results confirm the robustness and effectiveness of our method in faithfully executing user instructions while maintaining original image content. Furthermore, our approach consistently outperforms existing baselines on multiple real-world benchmarks, highlighting its strong generalization ability and practical applicability in interactive photo retouching.

E Broad Impact

JarvisArt revolutionizes the field of photo retouching by creating an intelligent artist agent that democratizes access to professional tools for non-expert users, fostering creativity in areas like education, marketing, and art. It streamlines workflows in industries reliant on visual media, while its transparent decision-making ensures trust and reliability. However, risks such as the potential misuse for hyper-realistic alterations, including deepfakes, and the creation of misleading content by distorting human images with offensive elements remain. To mitigate these issues, promoting responsible use and developing ethical guidelines will be key priorities in future model improvements.



Figure 3: Examples of (a) source photos, (b) human-region masks, and (c) their retouched references. The masks are used to calculate region-level metrics for evaluating local editing performance.



User Instruction:
 <Image> I want this image to have a soft, misty style. In the <box>[0.3, 0.25, 0.7, 0.6]</box> area, which is the face and upper body of the figure, I wanted the skin color to be smoother and more natural. The overall colors should blend softly, with a warm, green feeling that evokes nostalgia. The foliage in the background should be vibrant but not too strong, creating a serene atmosphere.

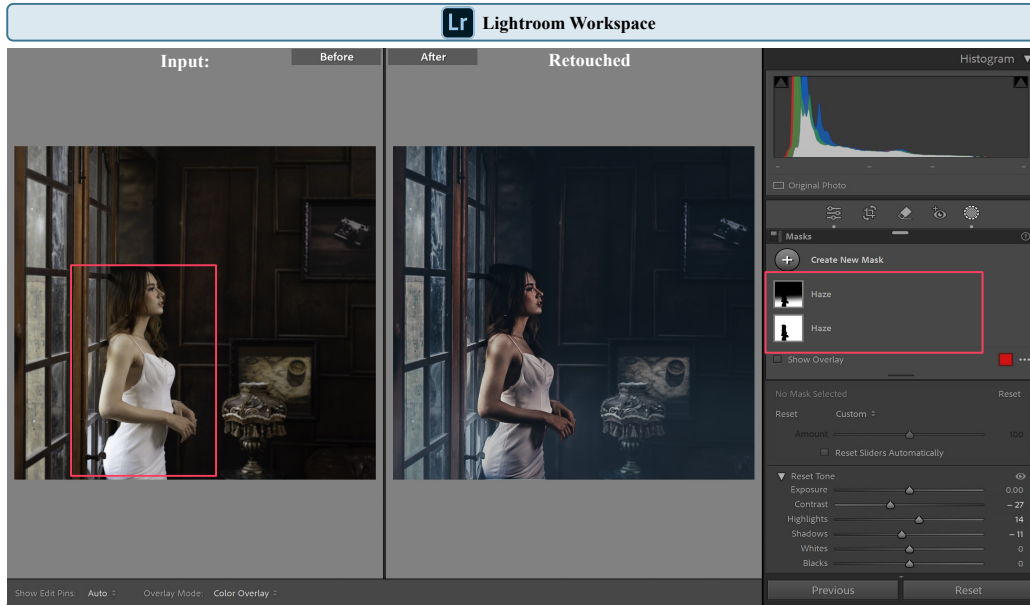
Answer with CoT annotation:

<think>Alright, I've got this image to edit for a soft, misty style. The user wants smoother skin tones and gentle color blending with a warm, vintage feel, focusing on the person's face and upper body in <box>0.3 0.25 0.7 0.6</box>. The background foliage should stay vibrant but not overpowering. First, I'll adjust the white balance for warmth. Then, apply a slight blur or glow effect for the misty style. For the skin tones, I'll use masks to reduce saturation and smooth textures in the specified area. Next, I'll tweak the tone curve to soften contrasts and warm up highlights and midtones. Background colors will be balanced to complement the foreground without being too intense. Adding subtle film grain can enhance the vintage look. Finally, I'll ensure all elements harmonize for a serene, nostalgic atmosphere. These steps align with the user's vision while preserving the original image's quality.</think>

<answer>PresetType = "Normal", Cluster = "", SupportsAmount2 = true, SupportsAmount = true, SupportsColor = true, SupportsMonochrome = true, SupportsHighDynamicRange = true, SupportsNormalDynamicRange = true, SupportsSceneReferred = true, SupportsOutputReferred = true, ProcessVersion = "15.4", WhiteBalance = "As Shot", ParametricShadowSplit = 25, ParametricMidtoneSplit = 50, ParametricHighlightSplit = 75, ColorGradeBlending = 50, DefringePurpleHueLo = 30, DefringePurpleHueHi = 70, DefringeGreenHueLo = 40, DefringeGreenHueHi = 60, PerspectiveScale = 100, ToneCurveName2012 = "Linear", HasSettings = true, Look = { Name = "Adobe Color", Amount = 1, Stubbed = true, Parameters = { CompatibleVersion = 251920384, ProcessVersion = "15.4", CurveRefineSaturation = 50, ConvertToGrayscale = false, RGBTable = "F4B5250D16A58B460C36CEC16C0627D2", ToneCurvePV2012 = { [1] = 0, [2] = 0, [3] = 16, [4] = 24, [5] = 64, [6] = 72, [7] = 128, [8] = 128, [9] = 192, [10] = 176, [11] = 244, [12] = 234, [13] = 255, [14] = 255 }, ToneCurvePV2012Red = { [1] = 0, [2] = 0, [3] = 255, [4] = 255 }, ToneCurvePV2012Green = { [1] = 0, [2] = 0, [3] = 255, [4] = 255 }, ToneCurvePV2012Blue = { [1] = 0, [2] = 0, [3] = 255, [4] = 255 } }, LensBlur = { Version = 1, Active = true, ImageOrientation = 0, FocalRange = "0 0 100 100", BlurAmount = 50, BokehShape = 0, BokehShapeDetail = 0, HighlightsThreshold = 50, HighlightsBoost = 50, CatEyeAmount = 0, CatEyeScale = 100, BokehAspect = 0, BokehRotation = 0, SphericalAberration = 0 }, MaskGroupBasedCorrections = { { What = "Correction", CorrectionAmount = 1, CorrectionActive = true, CorrectionName = "Color Tone | Reduce RED", LocalToningHue = 119, LocalToningSaturation = 0.1, LocalTint = -0.05, LocalCurveRefineSaturation = 100, CorrectionMasks = { { What = "Mask/RangeMask", MaskActive = true, MaskName = "Color Range 1", MaskBlendMode = 1, MaskInverted = true, MaskValue = 0, CorrectionRangeMask = { Version = 3, Type = 1, ColorAmount = 0.494949, Invert = true, SampleType = 0, PointModels = { "0.595958 0.896660 0.597516 0.480823 0.955119 0" } } }, { What = "Mask/Image", MaskActive = true, MaskName = "person object box", MaskBlendMode = 1, MaskInverted = true, MaskValue = 0, MaskVersion = 1, MaskSubType = 0, ReferencePoint = "0.500000 0.500000", ErrorReason = 0, Gesture = { { What = "Mask/Polygon", MaskActive = true, MaskBlendMode = 0, MaskInverted = false, MaskValue = 1, Points = { { X = 0.3, Y = 0.25 }, { X = 0.7, Y = 0.25 }, { X = 0.3, Y = 0.6 }, { X = 0.7, Y = 0.6 } } } } } } } }

<answer>

Figure 4: Editing results with JarvisArt are generated under complex prompts, with all retouching operations performed in a Lightroom environment, allowing for iterative adjustments.



User Instruction:

<Image> My goal was to create an atmospheric, nostalgic look. For the area <box>[0.3, 0.25, 0.7, 0.6]</box> around her face and shoulders, I wanted smoother textures and a subtle haze effect. The entire image should evoke a sense of timelessness with muted colors and enhanced contrast, making it feel like a classic portrait.

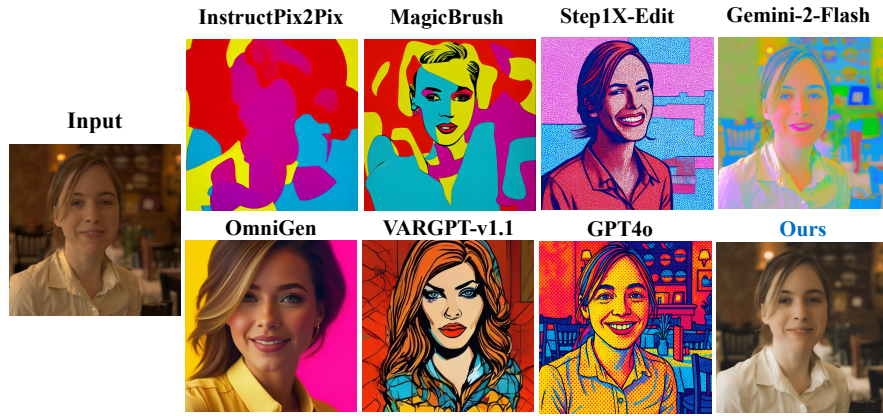
Answer with CoT annotation:

<think>Alright, let's dive into this image. The user wants a nostalgic feel with smooth textures and a subtle haze around the person's face and shoulders in <box>0.3 0.25 0.7 0.6</box>. We'll warm the white balance to 5185K for that vintage vibe. A slight overexposure of 0.23 brightens the subject, while contrast at 7 adds depth. Highlights reduced by -71 and shadows boosted by 50 preserve detail. Texture at 25 and clarity at 14 enhance details globally, but local dehaze softens the specified area. Vibrance up by 20 enriches colors subtly. Split toning with shadow hue at 229 and highlight hue at 48 adds a vintage touch. A dark vignette draws focus to the center. These tweaks balance global adjustments with targeted corrections, achieving the classic portrait look the user desires.</think>

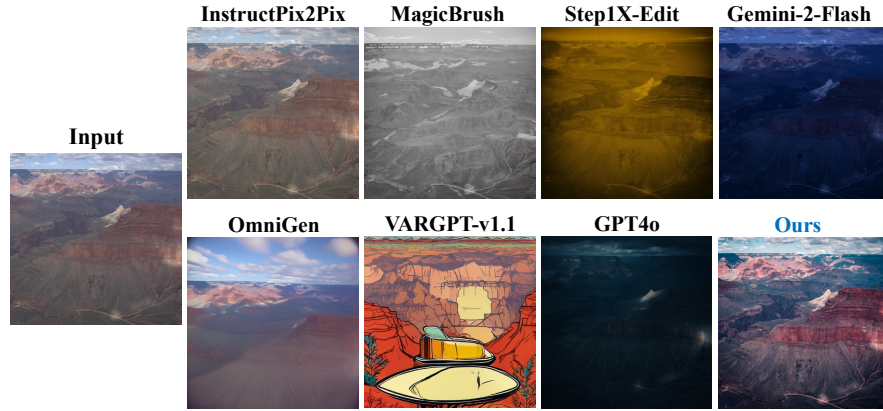
<answer>WhiteBalance = "Custom", Temperature = 5185, Tint = 10, Exposure2012 = 0.23, Contrast2012 = 7, Highlights2012 = -71, Shadows2012 = 50, Whites2012 = 12, Blacks2012 = -18, Texture = 25, Clarity2012 = 14, Dehaze = 11, Vibrance = 20, Saturation = 3, ParametricShadowSplit = 25, ParametricMidtoneSplit = 50, ParametricHighlightSplit = 75, Sharpness = 40, SharpenRadius = 1, SharpenDetail = 25, LuminanceSmoothing = 13, LuminanceNoiseReductionDetail = 50, ColorNoiseReduction = 25, ColorNoiseReductionDetail = 50, ColorNoiseReductionSmoothness = 50, HueAdjustmentOrange = -11, HueAdjustmentYellow = -3, SaturationAdjustmentOrange = -20, SaturationAdjustmentYellow = -50, SaturationAdjustmentBlue = -31, LuminanceAdjustmentOrange = -30, LuminanceAdjustmentYellow = 12, LuminanceAdjustmentAqua = -12, LuminanceAdjustmentBlue = -30, SplitToningShadowHue = 229, SplitToningShadowSaturation = 13, SplitToningHighlightHue = 48, SplitToningHighlightSaturation = 10, ColorGradeBlending = 100, AutoLateralCA = 1, DefringePurpleHueLo = 30, DefringePurpleHueHi = 70, DefringeGreenHueLo = 40, DefringeGreenHueHi = 60, PerspectiveScale = 100, PostCropVignetteAmount = -11, PostCropVignetteMidpoint = 50, PostCropVignetteFeather = 50, PostCropVignetteStyle = 1, RedHue = 30, RedSaturation = -10, GreenHue = 40, GreenSaturation = -10, BlueHue = -25, BlueSaturation = -10, OverrideLookVignette = true, ToneCurveName2012 = "Custom", HasSettings = true, ToneCurvePV2012 = { [1] = 0, [2] = 29, [3] = 48, [4] = 48, [5] = 127, [6] = 126, [7] = 190, [8] = 192, [9] = 255, [10] = 255 }, ToneCurvePV2012Red = { [1] = 0, [2] = 0, [3] = 47, [4] = 27, [5] = 115, [6] = 125, [7] = 159, [8] = 173, [9] = 255, [10] = 255 }, ToneCurvePV2012Green = { [1] = 0, [2] = 0, [3] = 43, [4] = 25, [5] = 113, [6] = 121, [7] = 160, [8] = 174, [9] = 255, [10] = 255 }, ToneCurvePV2012Blue = { [1] = 0, [2] = 0, [3] = 39, [4] = 25, [5] = 113, [6] = 124, [7] = 159, [8] = 174, [9] = 255, [10] = 255 }, MaskGroupBasedCorrections = { { What = "Correction", CorrectionAmount = 1, CorrectionActive = true, CorrectionName = "Haze", LocalToningHue = 240, LocalBlacks2012 = 0.0675, LocalClarity2012 = -0.198322, LocalDehaze = -0.208022, LocalTemperature = -0.1024, LocalTint = -0.09851, LocalTexture = -0.250785, CorrectionMasks = { { What = "Mask/Image", MaskActive = true, MaskName = "Subject 1", MaskBlendMode = 0, MaskInverted = true, MaskValue = 1, MaskVersion = 1, MaskSubType = 1, ReferencePoint = "0.306641 0.521994", WholeImageArea = "0/1,0/1,1707/1,2560/1", Origin = "0,531", ModelVersion = 234881976 } } }, { What = "Correction", CorrectionAmount = 1, CorrectionActive = true, CorrectionName = "Haze", LocalToningHue = 240, LocalBlacks2012 = 0.175759, LocalClarity2012 = -0.284554, LocalDehaze = -0.364241, LocalTemperature = -0.163366, LocalTint = -0.188755, LocalTexture = -0.288385, CorrectionMasks = { { What = "Mask/Gradient", MaskActive = true, MaskName = "Linear Gradient 1", MaskBlendMode = 0, MaskInverted = false, MaskValue = 1, ZeroX = 0.426926, ZeroY = 0.47269, FullX = 0.44185, FullY = 0.872828 }, { What = "Mask/Image", MaskActive = true, MaskName = "Subject 1", MaskBlendMode = 1, MaskInverted = false, MaskValue = 0, MaskVersion = 1, MaskSubType = 1, ReferencePoint = "0.306641 0.521994", WholeImageArea = "0/1,0/1,1707/1,2560/1", Origin = "0,531", ModelVersion = 234881976 } } } }

</answer>

Figure 5: Editing results with JarvisArt are generated under complex prompts, with all retouching operations performed in a Lightroom environment, allowing for iterative adjustments.



Make her look more vibrant and lively, like a pop art style.



I want a moody, atmospheric look with deeper shadows and cooler tones.



I aim for a melancholy blues style to evoke a sense of nostalgia and depth.



I want a dreamy, soft look with cooler tones for a nostalgic feel.

Figure 6: Visual comparisons of all instruction-based editing methods on MIT-FiveK [2].

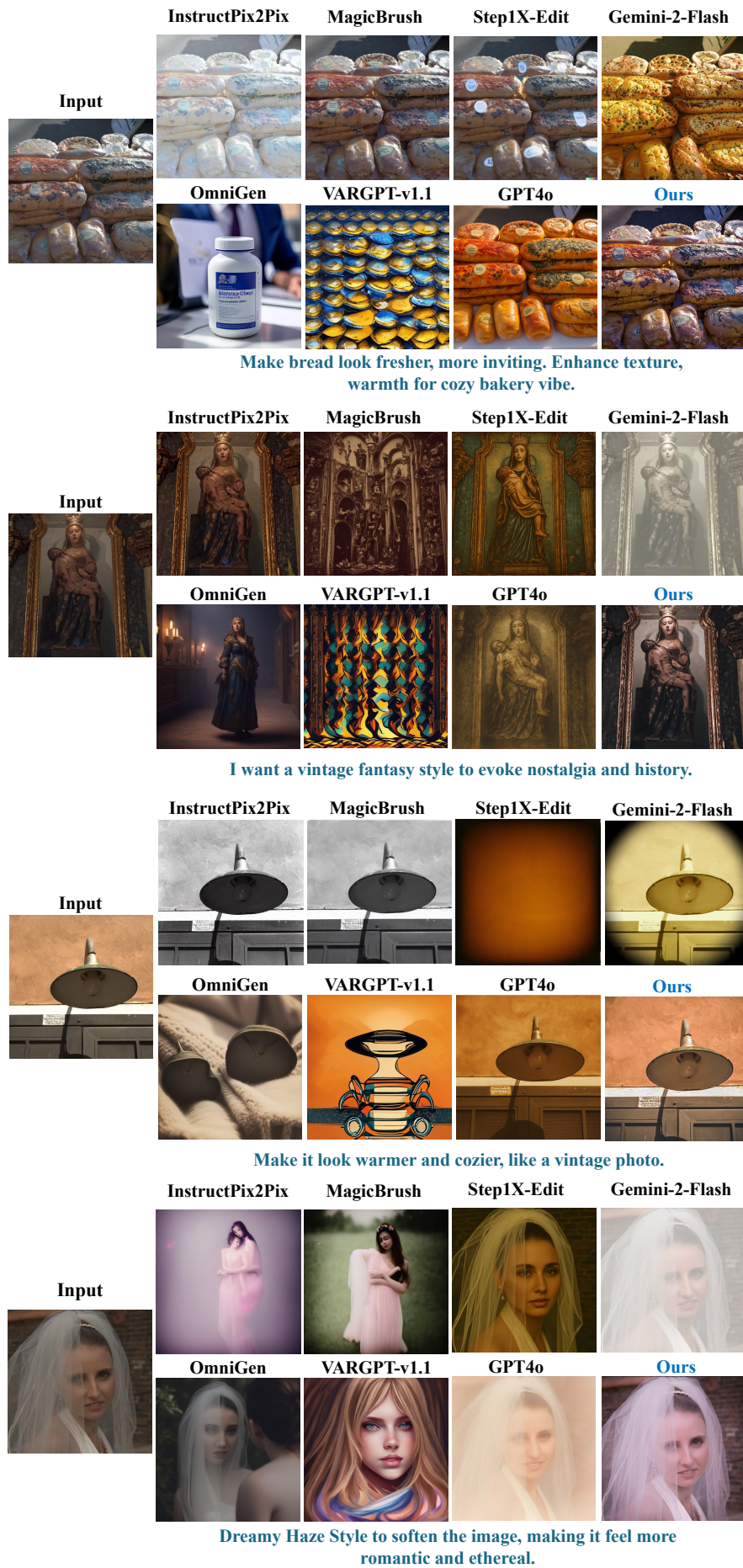


Figure 7: Visual comparisons of all instruction-based editing methods on MIT-FiveK [2].

References

- [1] T. Brooks, A. Holynski, and A. A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18392–18402, 2023.
- [2] V. Bychkovsky, S. Paris, E. Chan, and F. Durand. Learning photographic global tonal adjustment with a database of input / output image pairs. In *The Twenty-Fourth IEEE Conference on Computer Vision and Pattern Recognition*, 2011.
- [3] A. Hurst, A. Lerer, A. P. Goucher, A. Perelman, A. Ramesh, A. Clark, A. Ostrow, A. Welihinda, A. Hayes, A. Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.
- [4] J. Liang, H. Zeng, M. Cui, X. Xie, and L. Zhang. 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, 2021.
- [5] S. Liu, Y. Han, P. Xing, F. Yin, R. Wang, W. Cheng, J. Liao, Y. Wang, H. Fu, C. Han, et al. Step1x-edit: A practical framework for general image editing. *arXiv preprint arXiv:2504.17761*, 2025.
- [6] W. Ouyang, Y. Dong, X. Kang, P. Ren, X. Xu, and X. Xie. Rsfnet: A white-box image retouching approach using region-specific color filters. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12160–12169, 2023.
- [7] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, E. Mintun, J. Pan, K. V. Alwala, N. Carion, C.-Y. Wu, R. Girshick, P. Dollár, and C. Feichtenhofer. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024.
- [8] G. Team, R. Anil, S. Borgeaud, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, K. Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.
- [9] S. Xiao, Y. Wang, J. Zhou, H. Yuan, X. Xing, R. Yan, S. Wang, T. Huang, and Z. Liu. Omnigen: Unified image generation. *arXiv preprint arXiv:2409.11340*, 2024.
- [10] H. Zeng, J. Cai, L. Li, Z. Cao, and L. Zhang. 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, 2020.
- [11] K. Zhang, L. Mo, W. Chen, H. Sun, and Y. Su. Magicbrush: A manually annotated dataset for instruction-guided image editing. *Advances in Neural Information Processing Systems*, 36:31428–31449, 2023.
- [12] X. Zhuang, Y. Xie, Y. Deng, D. Yang, L. Liang, J. Ru, Y. Yin, and Y. Zou. Vargpt-v1. 1: Improve visual autoregressive large unified model via iterative instruction tuning and reinforcement learning. *arXiv preprint arXiv:2504.02949*, 2025.