Supplementary Material of
Restore, Assess, Repeat: A Unified Framework for Iterative Image Restoration

I-Hsiang Chen[1,2,*] Isma Hadji[1] Enrique Sanchez[1] Adrian Bulat[1,3] Sy-Yen Kuo[2,4]
Radu Timofte[5] Georgios Tzimiropoulos[1,6] Brais Martinez[1]

[1]Samsung AI Center Cambridge [2]National Taiwan University [3]Technical University of Iasi
[4] Chang Gung University [5]University of Wurzburg [6]Queen Mary University of London

1. Latent Quality Assessment

1.1. Finetuning and Usage

To enable rich degradation-awareness of the LQA module, we follow the DepictQA [10] pipeline and further finetune the original model on three complementary tasks: (1) distortion identification, (2) assessment reasoning, and (3) quality comparison, as illustrated in Figure 1. To better adapt LQA to our restoration setting, we further extend the distortion-identification corpus using all training pairs from the restoration datasets. During inference, we primarily use distortion identification to generate conditioning signals, and quality comparison to determine the RAR stopping criterion, all using the same VLM backbone.

1.2. Architecture Details

To unify the IQA and restoration modules, we introduced two components: latent space alignment and conditioning alignment, enabling the assessment model to operate directly in the shared latent domain. We describe each component below.

Latent Space Alignment Latent space alignment enables the IQA module to operate directly within the shared latent space of the restoration model. Instead of using the original IQA encoder $\mathcal{E}_{IQA}$, we replace it with the restoration encoder $\mathcal{E}_{restore}$ and introduce an image adapter A_I to bridge their feature spaces. Training proceeds in two stages: first, we align the adapter outputs to the IQA encoder representations using simple MSE loss; then, we perform end-to-end finetuning, following the original IQA losses. An overview is shown in Figure 2.

Conditioning Alignment Stable Diffusion 3.5 [2] relies on three separate text encoders (CLIP-L [7], CLIP-G [7], and T5-XXL [9]) to produce conditioning embeddings, making text-based conditioning computationally expensive and inefficient for iterative restoration. Moreover, the IQA mod-

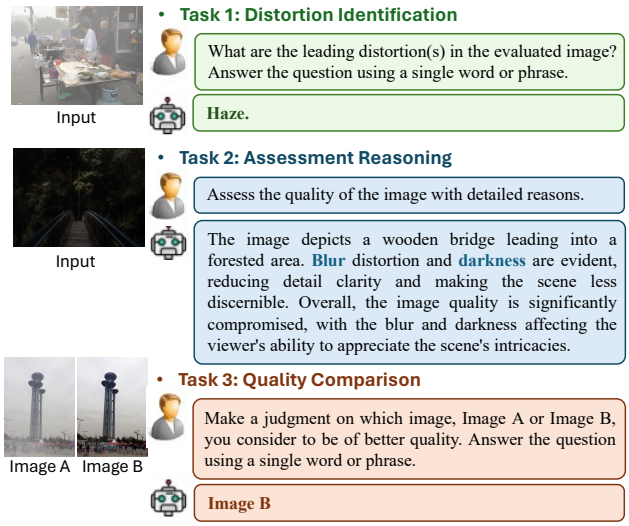

Figure 1. **Illustrations of the three prompt-driven IQA tasks used in LQA *training***: distortion identification, assessment reasoning, and quality comparison. These tasks collectively preserve rich degradation-awareness within the unified VLM backbone.

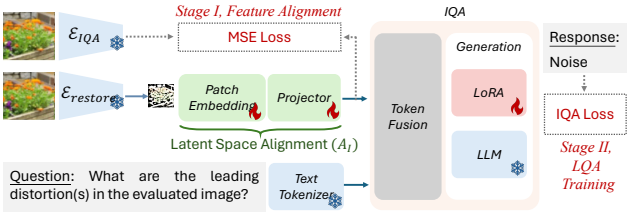

Figure 2. **Overview of the latent space alignment module**, which allows us to use a single encoder for both IQA and restoration modules.

Project: https://restore-assess-repeat.github.io/

* Work conducted during I-Hsiang's internship at Samsung AI Center, Cambridge, UK.

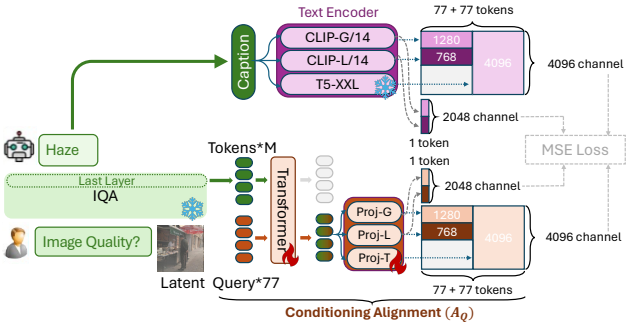

Figure 3. **Overview of the conditioning alignment module**, which maps LQA outputs into the UIR conditioning space.

Table 1. **Distortion identification on composite degradations.** Evaluated on KADIS700K [6] dataset.

Method	Precision ↑	Recall ↑	F1-score ↑
DepictQA (Pixel-based IQA)	0.7713	0.7564	0.7612
LQA	**0.8750**	**0.8748**	**0.8749**

Table 2. **Distortion identification on single degradations.** F1-scores on each restoration dataset.

Method	Haze	Blur	Rain	LOL	Raindrop	Noise	Low-Res	Average
CLIP	0.674	0.721	0.889	0.882	0.757	0.805	0.723	0.778
SA-CLIP	1.000	0.944	0.995	1.000	1.000	1.000	0.998	0.991
LQA	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**0.975**	**0.996**

Table 3. **Sensitivity Analysis of User Prompts.**

Degradation	Prompts	PSNR ↑	SSIM ↑	LPIPS ↓	CLIPIQA ↑	MUSIQ ↑	MINIQA ↑
Group C	Fixed	19.33	0.6579	0.1489	0.6554	56.56	0.4653
	Random	19.31	0.6572	0.1493	0.6566	56.72	0.4675

ule naturally outputs free-form text, which would require decoding and subsequent re-encoding through these large text encoders in our target setup, a process that introduces unnecessary information loss.

Instead, we project the IQA output latent $\tilde{Q}_{\text{deg}}$ into the UIR conditioning space through an adapter A_Q, which is trained to match the embeddings produced by the restoration model's text-conditioning branch. The alignment is optimized in two stages: first finetuning A_Q, and then jointly training it with the UIR module, as shown in Figure 3.

This design eliminates the need for IQA decoding and text re-encoding, thereby avoiding both information loss and computational inefficiency, while enabling fully end-to-end conditioning.

1.3. Evaluation of LQA

To validate the accuracy of our LQA module, we evaluate distortion identification performance on both composite and single-degradation settings. On the publicly available KADIS700K [6] composite degradation benchmark, LQA significantly outperforms the original DepictQA model. We further compare against CLIP [2] and SA-CLIP (as adopted in AutoDIR [5]) on the restoration datasets, where LQA achieves near-perfect F1-scores across all degradation types. These results demonstrate that LQA preserves strong degradation-awareness while operating entirely in the shared latent space.

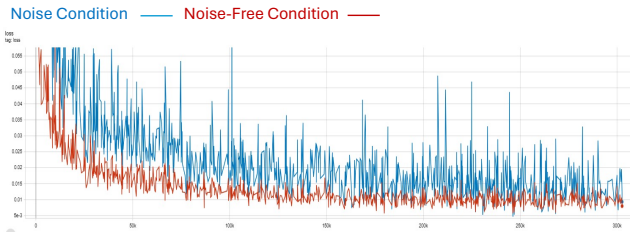

Figure 4. Training curves comparing noise-free flow matching with the standard noise-conditioned variant, showing significantly faster and stable convergence.

1.4. Prompt Definition

We follow the DepictQA [10] procedure and use a fixed prompt template in our LQA. The exact prompt formulation is provided in Figure 1. To evaluate sensitivity to prompt wording, we additionally construct a predefined pool of 24 semantically equivalent prompts and randomly sample one at test time for each decision step. Results on composite distortion-Group C are summarized in Table 3. As shown, the performance remains highly consistent between the fixed-prompt and random-prompt settings across all full-reference and no-reference metrics. These results indicate that the stopping decision is not sensitive to minor variations in prompt wording, suggesting that the LQA-based termination mechanism provides a stable decision signal.

2. More Experimental Results

2.1. Noise-free Flow Matching Analysis

In standard flow matching formulation, the model is trained to learn a mapping from noise to target distributions, while using additional conditioning as an extra input. Instead, we proposed a noise-free flow matching where we directly learn mapping from degraded images distribution to target distribution. This definition, offers several practical advantages in the context of iterative restoration. First, by removing the stochastic noise conditioning, the model avoids the accumulation of noise-induced errors across multiple refinement steps, making the RAR procedure substantially more stable, as shown in Figure 7 of the main paper. Second, the clean latent trajectory enables the LQA module to reliably assess intermediate latents at any iteration without being corrupted by injected noise, ensuring that the quality

Table 4. **Ablation** of the integration process between the IQA and the UIR modules.

Backbone	Configuration				Unknwon (UDC)			Composite (Group-C)		
	IQA	Text Space	Image Space	Iterative Training	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
AutoDIR	CLIP	Text	Pixel	✗	22.47	0.7267	0.2199	18.61	0.5443	0.5019
SD1.5	CLIP	Text	Pixel	✗	20.67	0.6217	0.2211	16.70	0.4848	0.2997
SD1.5	DepictQA	Text	Pixel	✗	22.34	0.6596	0.2016	17.67	0.5260	0.2267
SD1.5	DepictQA	Text	Latent	✗	23.66	0.7055	0.1498	18.06	0.5475	0.1792
SD1.5	DepictQA	Embeddings	Latent	✗	25.57	0.7473	0.1319	18.37	0.5546	0.1751
SD1.5	DepictQA	Embeddings	Latent	✓	23.05	0.6970	0.1544	18.17	0.5530	0.1774
SD3.5	DepictQA	Text	Pixel	✗	24.70	0.7766	0.1589	17.89	0.5776	0.2289
SD3.5	DepictQA	Text	Latent	✗	24.90	0.7952	0.1131	18.72	0.6278	0.1576
SD3.5	DepictQA	Embedding	Latent	✗	28.49	0.8494	0.1007	18.76	0.6254	0.1730
SD3.5	DepictQA	Embedding	Latent	✓	**28.60**	**0.8559**	**0.0826**	**19.16**	**0.6494**	**0.1495**

assessment remains consistent throughout the process. Finally, as illustrated in Figure 4, the noise-free variant converges significantly faster, showing smoother optimization dynamics and reduced variance during training. This property makes iterative assessment during training feasible, allowing the model to perform targeted refinement with stable and efficient updates within the RAR loop.

2.2. Effectiveness of Proposed Modules

To assess the effectiveness of the proposed modules, we provide extended ablations on the integration between the IQA and UIR components across both SD1.5 and SD3.5. As shown in Table 4, replacing CLIP [8] with DepictQA [10] consistently improves generalization under unknown and composite degradations. Notably, combining DepictQA [10] with latent-space restoration already yields performance comparable to AutoDIR [5], which operates in pixel space and relies on an additional NAFNet [1] refinement stage. Further replacing text-based conditioning with aligned embeddings leads to clear improvements across most metrics, highlighting the advantage of a tighter and more direct coupling between IQA and UIR.

2.3. Investigation of Latent-space RAR

To further analyze the impact of RAR, we compare our latent-space formulation with a vanilla baseline, i.e., "SD3.5 with text conditioning operating in pixel space", which involves three separate stages: (1) generating text-based conditioning through the IQA encoder, and (2) encoding the image again from pixel space using UIR encoder and preforming restoration and (3) repeated VAE encode/decode operations to feedback into the IQA. As shown in Table 3, performing RAR entirely in latent space yields three key advantages. (1) *Reduced information loss*, as illustrated in Figure 5, repeated pixel-space encode/decode operations inevitably wash out fine details (e.g., the eye region), whereas latent refinement preserves structural fidelity across iterations. (2) *Higher efficiency*, pixel-space pipelines incur substantial overhead from text decoding/encoding (81.19 ms) and VAE (1.22 s), while our latent-space approach requires only a lightweight adapter (14.93 ms) for a 256×256 image on an A100 GPU. (3) *End-to-end optimization*, the shared latent formulation enables joint optimization of LQA and UIR, resulting in substantial improvements. For example, +13.63% PSNR and +48.01% LPIPS on unknown degradations with SD3.5.

(a) Input (b) Pixel + Text (c) Latent + Embedding (d) GT

Figure 5. **Comparison of RAR when iterative operating in pixel space versus latent space.** Please zoom in on the eye region for detailed comparison.

2.4. Quantitative Evaluation for Single Distortion

In the quantitative evaluation, we further assess the performance of the proposed method across seven single-degradation tasks. We note that in single-degradation benchmarks, annotators typically remove only the **main** distortion (e.g., rain), while other degradations may remain in the image. As our iterative framework removes all iden-

Table 5. Quantitative comparison on **Single Degradation** tasks.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	CLIP-IQA ↑	MUSIQ ↑	MANIQA ↑
Denoise (2 sets)						
LD	22.58	0.6250	0.1860	0.4586	44.66	0.3481
AirNet	29.10	0.8030	0.2130	0.4246	47.29	0.2997
PromptIR	29.89	0.8240	0.1810	0.4124	46.22	0.2905
AutoDIR	29.68	0.8320	0.1670	0.4654	43.46	0.2996
AgenticIR	27.89	0.8020	0.2680	0.4653	59.30	0.3917
RAR	30.21	0.8673	0.0419	0.7144	56.73	0.4287
Derain						
LD	23.21	0.6510	0.1470	0.5505	62.31	0.3399
AirNet	30.99	0.9290	0.0550	0.7338	70.50	0.4899
PromptIR	33.97	0.9380	0.0490	0.7426	70.89	0.5068
AutoDIR	35.09	0.9650	0.0470	0.5992	63.87	0.3490
AgenticIR	33.72	0.9485	0.0713	0.5809	66.38	0.3481
RAR	28.36	0.8342	0.0565	0.6953	58.40	0.4119
Dehaze						
LD	23.49	0.7630	0.0910	0.3633	60.88	0.3264
AirNet	26.52	0.9440	0.0310	0.4960	68.41	0.4309
PromptIR	29.13	0.9710	0.0220	0.5139	67.88	0.4239
AutoDIR	29.34	0.9730	0.0200	0.4004	62.54	0.3188
AgenticIR	23.87	0.8972	0.0834	0.3786	63.26	0.3000
RAR	27.18	0.9037	0.0332	0.4578	50.46	0.3438
Deraindrop						
LD	24.84	0.7380	0.1040	0.4977	67.83	0.4758
AirNet	27.13	0.8920	0.0940	0.4254	65.88	0.4684
PromptIR	27.41	0.9000	0.0810	0.4288	66.12	0.4740
AutoDIR	30.10	0.9240	0.0630	0.4089	69.33	0.4110
AgenticIR	21.76	0.8320	0.2163	0.3352	61.51	0.3149
RAR	25.33	0.8061	0.0710	0.4929	55.30	0.4155
Deblur						
LD	22.53	0.6950	0.2410	0.2170	49.64	0.2298
AirNet	26.50	0.8000	0.2010	0.1921	25.14	0.1148
PromptIR	26.82	0.8190	0.1980	0.2128	25.69	0.1191
AutoDIR	27.07	0.8280	0.1570	0.2494	47.30	0.2063
AgenticIR	34.18	0.9587	0.0411	0.2373	50.59	0.2321
RAR	26.09	0.8239	0.0700	0.3783	51.44	0.3301
Low Light						
LD	18.97	0.7700	0.1590	0.5551	71.56	0.4883
AirNet	21.26	0.8180	0.2370	0.5999	38.58	0.2848
PromptIR	22.42	0.8310	0.1330	0.3062	40.44	0.2820
AutoDIR	22.37	0.8880	0.1290	0.3528	67.21	0.3422
AgenticIR	9.89	0.4590	0.4123	0.2973	48.69	0.2651
RAR	21.75	0.8969	0.0901	0.4179	58.55	0.4235
Super-Resolution						
LD	19.58	0.6032	0.4266	0.3013	37.98	0.1940
AirNet	18.03	0.5751	0.4429	0.3171	26.51	0.1428
PromptIR	19.64	0.6286	0.3967	0.3273	26.71	0.1480
AutoDIR	21.04	0.6818	0.3148	0.3536	37.76	0.2162
AgenticIR	19.57	0.5580	0.5085	0.3393	41.41	0.2322
RAR	22.21	0.7326	0.1270	0.7393	61.12	0.5338

tified degradations (see Fig. 5 in main paper), it may deviate from the annotation-specific ground truth, which can lead to reduced PSNR/SSIM. A similar trend is observed in AgenticIR, which is also iterative: it achieves competitive performance on composite degradations but comparatively lower PSNR/SSIM on single-degradation settings. As shown in Table 5, this effect is particularly evident in deraining, where residual degradations are often not addressed by the annotations. Importantly, the PSNR/SSIM drop remains moderate (up to $\simeq 6\%$), while our method consistently achieves substantial gains (at least $\simeq 16\%$) across perceptual and no-reference metrics.

2.5. Investigation of the Stopping Criterion

We first analyze performance as a function of the number of restoration rounds on both single-degradation tasks and composite degradations (Group C), as shown in Table 6.

Table 6. **Stopping Criterion Analysis**. Results for single and composite degradations.

# rounds	PSNR ↑	SSIM ↑	LPIPS ↓	CLIPIQA ↑	MUSIQ ↑	MINIQA ↑
Single Degradation						
1	26.09	0.8485	0.0672	0.5463	55.78	0.4036
2	24.94	0.8334	0.0760	0.5836	58.15	0.4413
3	24.32	0.8212	0.0836	0.5970	59.31	0.4638
4	23.87	0.8127	0.0896	0.6033	59.91	0.4774
1.3	25.88	0.8378	0.0699	0.5566	56.00	0.4125
Composite Degradation - Group C						
1	19.47	0.6695	0.1562	0.6176	53.43	0.4180
2	19.36	0.6591	0.1459	0.6592	56.92	0.4700
3	19.28	0.6544	0.1477	0.6650	57.84	0.4875
4	19.20	0.6495	0.1508	0.6683	58.33	0.4982
2.94	19.33	0.6579	0.1489	0.6554	56.56	0.4653

Table 7. **Sensitivity Analysis of Iteration Counts.** Results for composite degradations on Group C.

Stopping	Nmax	# rounds	PSNR ↑	SSIM ↑	LPIPS ↓	CLIPIQA ↑	MUSIQ ↑	MINIQA ↑
	4	4	19.20	0.6495	0.1508	0.6683	58.33	0.4982
✓	4	2.87	19.32	0.6578	0.1499	0.6567	56.88	0.4694
	16	16	17.86	0.5851	0.2243	0.6326	58.33	0.5115
✓	16	3.02	19.29	0.6562	0.1519	0.6560	56.97	0.4719

Without the stopping mechanism, increasing the number of iterations may introduce mild oversharpening artifacts, which is reflected by gradual degradation in fidelity metrics (e.g., PSNR/SSIM), particularly under extended iteration budgets. In contrast, when the stopping rule is enabled, the average number of iterations aligns well with task complexity (e.g., 1.3 for Single and 2.94 for Group C), and performance remains stable.

We further increase the maximum allowed iterations to $N_{\text{max}} = 16$ and compare results with and without the stopping rule. As shown in Table 7, without stopping, excessive iterations lead to noticeable degradation in reconstruction quality (e.g., PSNR drops from 19.20 to 17.86 on Group C). However, with the stopping mechanism enabled, both $N_{\text{max}} = 4$ and $N_{\text{max}} = 16$ produce comparable average iteration counts and nearly identical performance. This demonstrates that the termination decision effectively prevents over-restoration and stabilizes the iterative process.

Importantly, the stopping decision is not based on predefined quality thresholds or metric comparisons. Instead, it relies on pairwise image comparison through the LQA model using a fixed prompt template (see Figure 1). By leveraging the same LQA module for both degradation assessment and termination decision, the framework maintains consistency between restoration guidance and stopping control, reducing potential divergence during iterative refinement.

2.6. Additional Qualitative Evaluations

As illustrated in Figure 6 to 9, our qualitative comparisons reveal clear advantages of RAR over AutoDIR [5] and AgenticIR [12] in both efficiency and restoration quality. In terms of efficiency, RAR consistently completes the

iterative restoration process using fewer steps, while AutoDIR [5] frequently generates redundant actions due to its difficulty in reasoning about complex composite degradations, and AgenticIR [12] incurs substantial computational overhead by invoking multiple task-specific restorers at every step. Regarding restoration quality, AgenticIR [12] is fundamentally constrained by its single-task restoration model, which often removing textures together with noise, as observed in Figure 6 (Sample 1, Step 2), leading to irreversible detail loss and a blurred final output. In contrast, RAR leverages a unified image restoration model that jointly models multiple degradation factors within the flow matching, enabling it to more effectively remove heterogeneous degradations while preserving semantic structures and fine textures.

3. Implementation Details

3.1. Training Details

RAR is built upon DepictQA [10] as the LQA backbone and Stable Diffusion 3.5 (SD3.5) [3] as the unified image restoration (UIR) backbone. All experiments are conducted using PyTorch on eight NVIDIA A100 GPUs. The complete training pipeline consists of three primary stages, preceded by an LQA initialization step. First, we initialize the LQA module with pretrained DepictQA [10] weights while keeping the IQA head frozen. To align the visual latent representation with the LQA feature space, we replace DepictQA's original image encoder with the UIR image encoder and fine-tune only the image adapter $\mathcal{A}_I$, which maps UIR latents into the IQA feature domain. After the adapter converges, we unfreeze the IQA head with LoRA for a brief joint refinement stage. This step is optimized using Adam with a batch size of 256, trained for 20 epochs for latent alignment and 1 additional epoch for IQA refinement. The learning rate is set to 3×10^{-4} with a WarmupDecayLR scheduler. Then, We train the embedding-conditioning adapter, $\mathcal{A}_Q$, while keeping all text encoders (CLIP-L, CLIP-G, and T5-XXL) frozen. This stabilizes linguistic conditioning and prevents drift from pretrained language priors. The adapter is trained using Adam with a learning rate of 5×10^{-5} and a cosine annealing scheduler for 20 epochs. Finally, we jointly optimize the full RAR framework, including the LQA module, the adapters, and the pre-tuned UIR backbone, using an iterative training process. We set the number of intermediate assessments to 4 steps and update both restoration inputs and LQA predictions at each step. This stage employs the CAMEWrapper optimizer with a cosine annealing schedule, using a learning rate of 2×10^{-6} and a batch size of 256, trained for 5 epochs.

Algorithm 1: RAR Inference Algorithm

Input:
Degraded image: I_{deg}
Encoder–Decoder: $\mathcal{E}(\cdot), \mathcal{D}(\cdot)$
Latent Quality Assessment: $\text{LQA}(\cdot)$
Unified Image Restoration: $\text{UIR}(\cdot)$
Assessment interval: T
Maximum iterations: $N_{\max}$

Initialization:
Encode image: $z^0_{\text{deg}} = \mathcal{E}(I_{\text{deg}})$
Initial assessment: $Q^0 = \text{LQA}_{\text{identify}}(z^0_{\text{deg}})$
Store previous latent: $z^{\text{prev}} = z^0_{\text{deg}}$
Set iteration counter: $n = 0$

while $n < N_{\max}$ **do**
  Restore phase (1 to T):
  for $t = 1$ **to** T **do**
    $z^{n+1}_{\text{deg}} = \text{UIR}(z^n_{\text{deg}}, t, Q^n)$
    $n = n + 1$
  end
  Assessment phase:
  $d = \text{LQA}_{\text{verify}}(z^{\text{prev}}, z^n_{\text{deg}})$
  if $d = \texttt{STOP}$ **then**
    return $\mathcal{D}(z^{\text{prev}})$
  end
  else
    $Q^n = \text{LQA}_{\text{identify}}(z^n_{\text{deg}})$
    $z^{\text{prev}} = z^n_{\text{deg}}$
  end
end
Output: Final restored image: $\mathcal{D}(z^n_{\text{deg}})$

3.2. Inference Details

During the inference stage, RAR performs iterative image restoration through a restore–assess feedback loop, enabling the outputs to be progressively optimized toward perceptual quality as guided by the LQA module. In addition, we adopt a LQA-based stopping criterion that terminates the iterative process once the latent quality verified by LQA does not improve over the previous iteration. The complete RAR inference pipeline is summarized in Algorithm 1. We set the number of restoration steps to $T = 4$ and the maximum inference iterations to $N_{\max} = 8$ to prevent excessively long refinement loops. We employ the Flow-Euler sampler with logit-normal weighting as the flow matching scheduler, which stabilizes the refinement trajectory during iterative restoration.

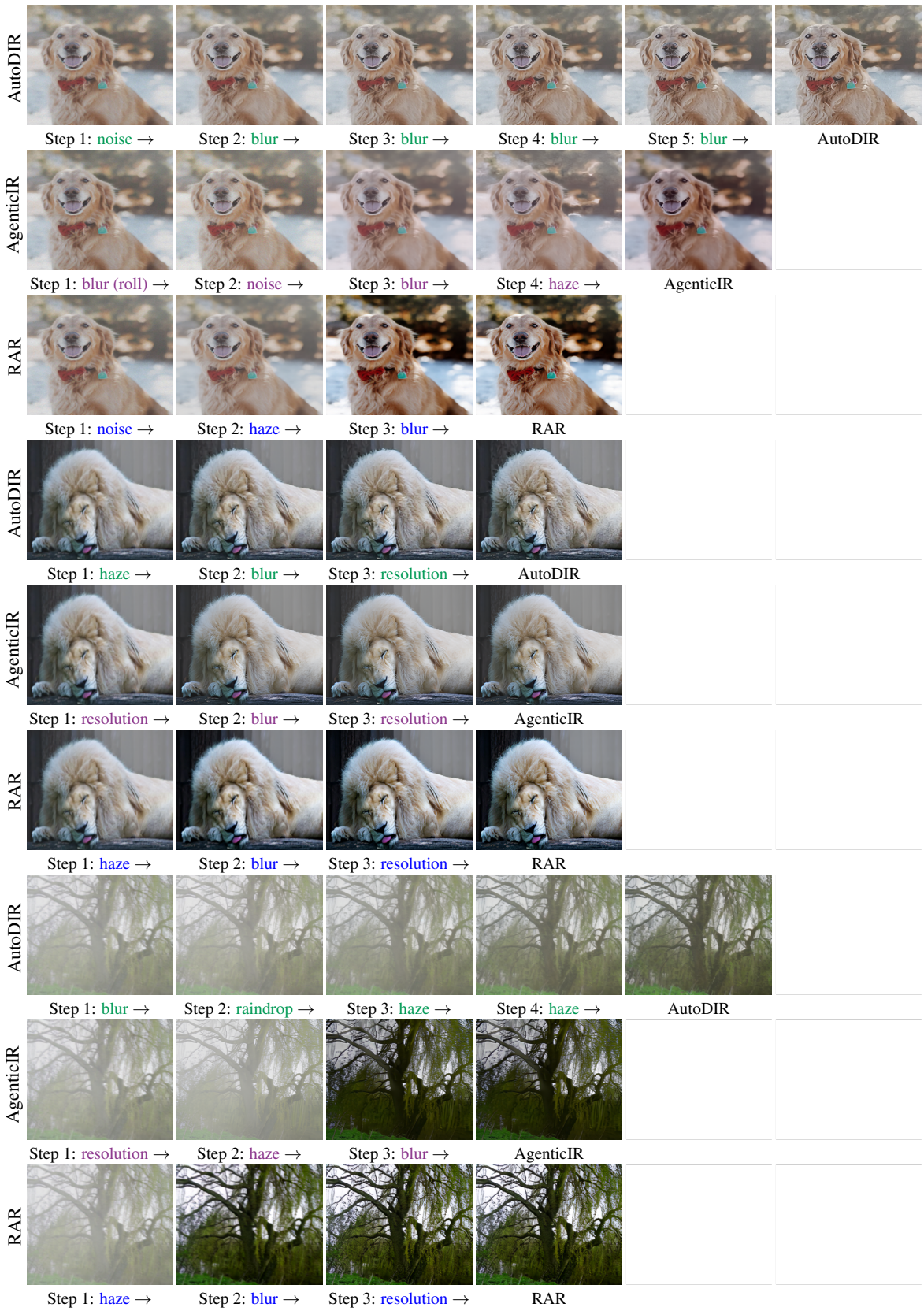

Figure 6. **Qualitative analysis of iterative restoration methods on composite degradations**, including haze, blur, noise, low-resolution conditions.

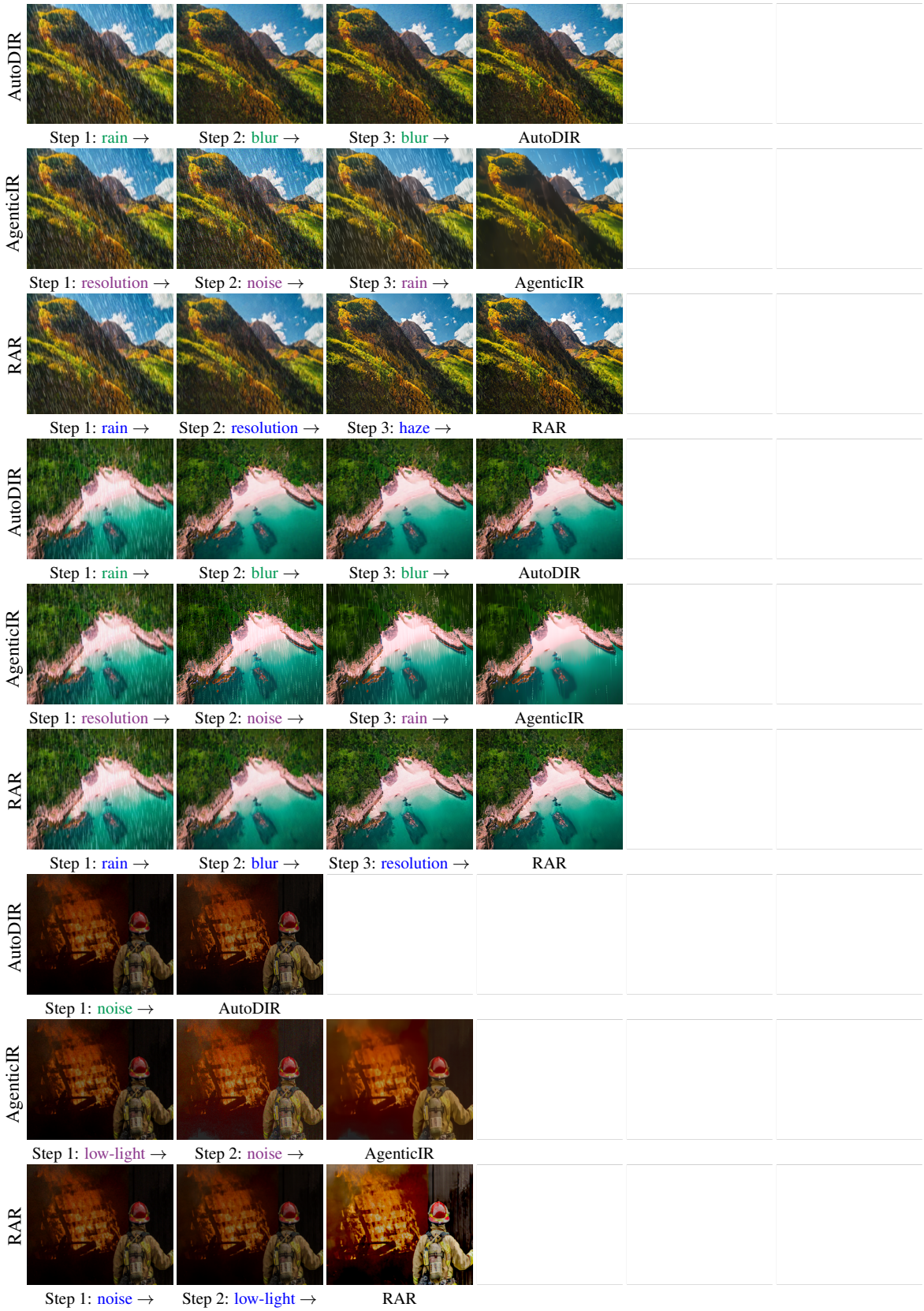

Figure 7. **Qualitative analysis of iterative restoration methods on composite degradations,** including rain, noise, low-resolution, low-light.

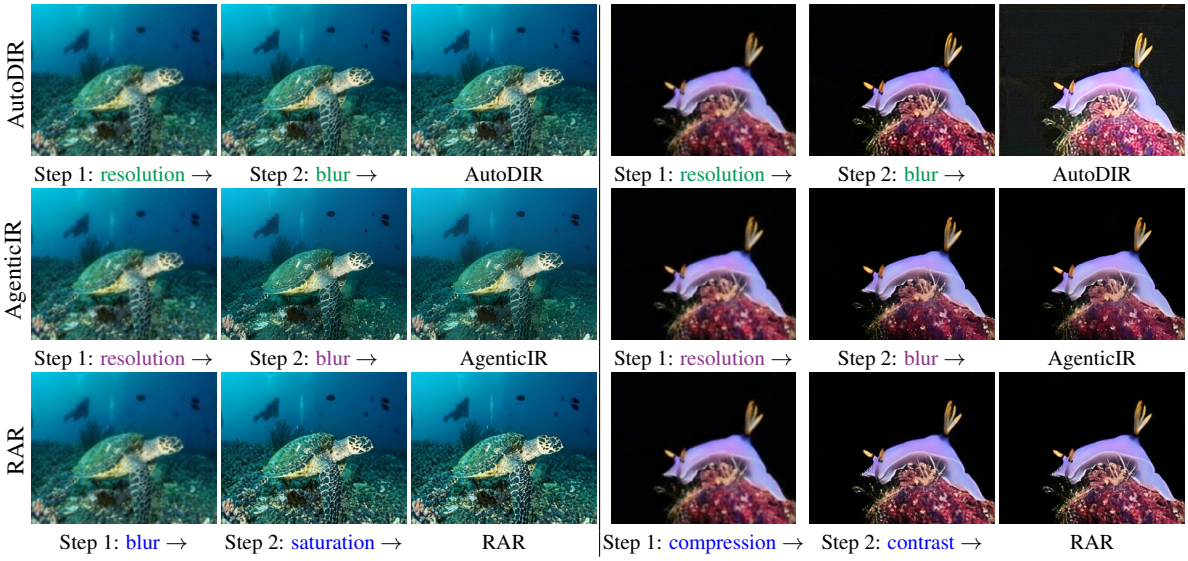

Figure 8. **Qualitative analysis of iterative restoration methods on EUVP [4] dataset.**

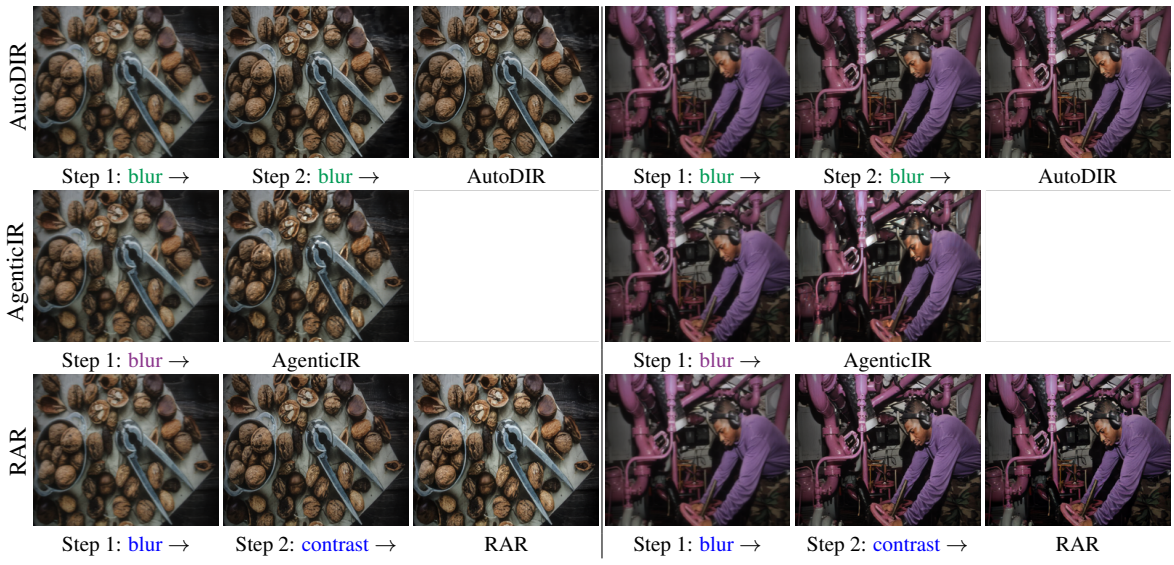

Figure 9. **Qualitative analysis of iterative restoration methods on UDC [11] dataset.**

References

[1] Xiaojie Chu, Liangyu Chen, and Wenqing Yu. Nafssr: Stereo image super-resolution using nafnet. In *IEEE Conference on Computer Vision and Pattern Recognition - Workshops*, 2022. 3

[2] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *ICML*, 2024. 1, 2

[3] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *International Conference on Machine Learning*, 2024. 5

[4] Md Jahidul Islam, Youya Xia, and Junaed Sattar. Fast underwater image enhancement for improved visual perception. *IEEE Robotics and Automation Letters*, 2020. 8

[5] Yitong Jiang, Zhaoyang Zhang, Tianfan Xue, and Jinwei Gu. AutoDIR: Automatic all-in-one image restoration with latent diffusion. In *European Conference on Computer Vision*, 2024. 2, 3, 4, 5

[6] Hanhe Lin, Vlad Hosu, and Dietmar Saupe. Deepfl-iqa: Weak supervision for deep iqa feature learning. *arXiv preprint arXiv:2001.08113*, 2020. 2

[7] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 1

[8] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021. 3

[9] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR*, 2020. 1

[10] Zhiyuan You, Zheyuan Li, Jinjin Gu, Zhenfei Yin, Tianfan Xue, and Chao Dong. Depicting beyond scores: Advancing image quality assessment through multi-modal language models. In *European Conference on Computer Vision*, 2024. 1, 2, 3, 5

[11] Yuqian Zhou, David Ren, Neil Emerton, Sehoon Lim, and Timothy Large. Image restoration for under-display camera. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 8

[12] Kaiwen Zhu, Jinjin Gu, Zhiyuan You, Yu Qiao, and Chao Dong. An intelligent agentic system for complex image restoration problems. *International Conference on Learning Representations*, 2025. 4, 5