Technical Appendix

This document supplements the main body of our paper with additional details, discussions, and results. In Section A, we present more details of the moiré pattern dataset collection, including a brief analysis of various previously overlooked factors affecting moiré pattern diversity. In Section B, we will provide a detailed explanation of the two stages involved in implementing UniDemoiré: Moiré Pattern Generator and Moiré Image Synthesis. In Section C, we provide more implementation details of experiments and show more qualitative results. Furthermore, as shown in Section C.3, we performed additional ablation experiments on the blending strategy in the Moiré Image Blending (MIB) module and the design of the upsampling block and the loss function in the Tone Refinement Network (TRN).

A Dataset Capture and Analysis

In this section, we first present a brief introduction of various previously overlooked factors of devices that affect moiré pattern diversity. Then, we provide more details about our capture settings.

A.1 The Impact of Device on Moiré Pattern Diversity

Previous studies (Yu et al. 2022; Yang et al. 2023) have indicated that the geometric correlation between the screen and the camera significantly influences the features of the moiré pattern. However, such studies have overlooked that some aspects of the camera and the screen can also impact the moiré pattern.

For cameras, the two most critical factors affecting the moiré pattern are the CMOS and the lens used. The pixel density of a CMOS sensor (i.e., the number of pixels per unit area) determines its maximum sampling frequency, also known as the Nyquist frequency. The higher the pixel density, the higher the sampling frequency of the sensor and the higher the frequency of the signal that can be sampled, resulting in a higher frequency of moiré produced by the aliasing effect, which impacts the moiré pattern. In addition, the lens's focal length also affects the formation of moiré. In cell phone photography, lenses with shorter focal lengths (e.g., wide lenses/main camera lenses) usually have wider angles of view and can capture more of the scene content. Lenses with longer focal lengths (such as telephoto or telescopic lenses), on the other hand, offer a narrower angle of view and greater magnification for capturing distant details. When the screen is photographed with lenses of different focal lengths, the relative positional relationship between the pixels on the sensor and the pixels on the screen changes, which may cause the moiré pattern to appear or disappear.

Furthermore, the layout and the distance of pixels dots in the panel used can also significantly impact the formation of moiré on the display screen. The frequency of detail that a screen can display depends on how the pixel dots are arranged. Various arrangements result in distinct frequencies of detail, which impacts the formation of moiré patterns. The distance between pixel dots on the screen then affects the shooting distance. Larger pixel dot spacing will make the distance at which the moiré is formed to be photographed farther away. Conversely, the smaller the distance between pixel dots, the closer the distance needed to photograph the molded moiré.

A.2 More Details about Capture Settings

Based on the above analysis, we take screen images through different camera viewpoints to generate diverse moiré patterns. Specifically, we apply six mobile phones and six digital screens, as shown in Table 5 and $6\,(6 \times 6 = 36$ combinations). Figure 7 shows a comparison of some of our captured 4K moiré patterns with some samples from the MoireSpace dataset. It can be seen that our captured moiré patterns are much better than the MoireSpace dataset in terms of the clarity of the moiré texture and the vividness of the colors.

Mobile Phones We chose six mobile phones with varying camera specifications to capture diverse moiré patterns, as shown in Table 5. Our selection criteria included the camera type, CMOS category, and number of megapixels. For the regular main camera with moderate resolution, we picked the iPhone 12 and iPhone 13. For electronic zooming at 2x and 3x, we selected the Honor 90 and Xiaomi 10s, which have high pixels. Additionally, we picked two iPhone 12 Pro and iPhone 15 Pro models with different CMOS specifically for telephoto lenses. These models use the telephoto lens for optical zoom at fixed magnifications of 2x and 3x.

Display Screens To capture a wider variety of moiré patterns in different forms, we selected display screens based on size, panel type, and resolution guidelines to maximize pixel point layouts and spacing on the screen. As shown in Table 6, we have selected three 27-inch IPS panel LED matte screen monitors with a 2K resolution - DELL D2720DS, AOC 27G2G8, and Philips 27E1N5500. This specification is the most common among the available options. The AOC 27G2G8 is a W-LED monitor with an RGBW pixel layout. This IPS screen has white sub-pixels in addition to the standard RGB arrangement, creating a more varied pixel point layout. To capture the moiré pattern on the glossy display, we opted for a 13.3-inch IPS panel with a 2K resolution MacBook Air notebook. Finally, we selected two high-resolution displays: the Xiaomi C34WQBA-RG and the ViewSonic VX2771-4K-HD. These displays were explicitly chosen to capture moiré patterns with smaller pixel dot spacing. The Xiaomi C34WQBA-RG is a 34-inch curved display with an SVA panel and W-LED technology. It boasts a 3K resolution. On the other hand, the ViewSonic VX2771-4K-HD is a 27-inch matte screen display with an IPS panel and LED technology. It offers a standard 4K resolution.

B Further details of our Method

This section will showcase the details of the implementation of our UniDemoiré's Moiré Pattern Generator stage and Moiré Image Synthesis stage.

B.1 Moiré Pattern Generator

The visualization of moiré pattern patches generated using the Moiré Pattern Generator(MPG) is shown in Figure 8.

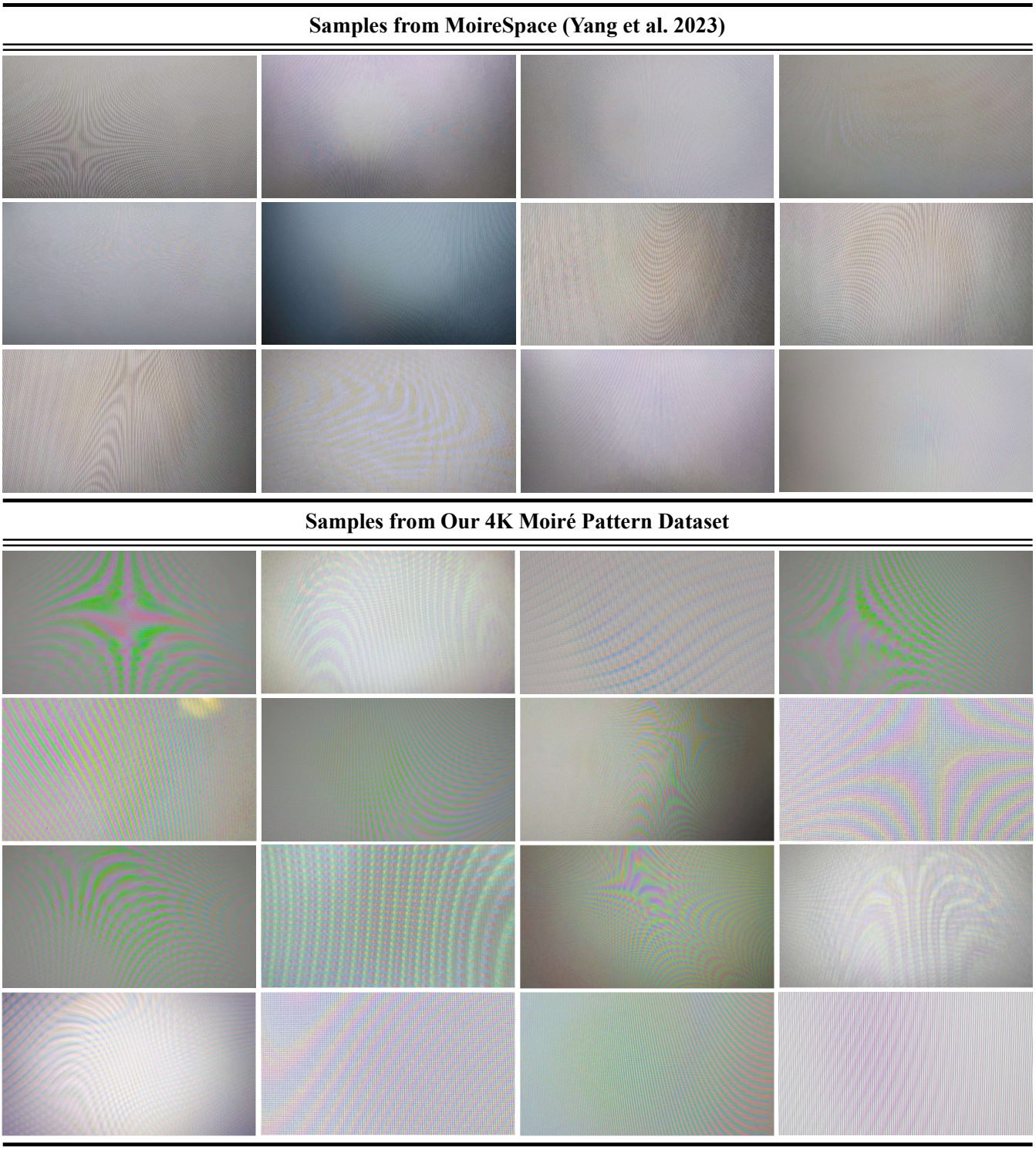

Figure 7: Samples from MoireSpace (Yang et al. 2023) and our 4K moiré pattern dataset.

Mobile Phone	Camera	CMOS	MegaPixel (MP)	Optical format (Inches)	Pixel Size (μm)
iPhone 12	Main	SONY IMX503	12	1/2.55	1.40
iPhone 13	Main	SONY IMX603	12	1/1.88	1.70
Honor 90	Main	ISOCELL HP3	200	1/1.40	0.56
Xiaomi 10s	Main	ISOCELL HMX	108	1/1.33	0.80
iPhone 12 Pro	Main	SONY IMX503	12	1/2.55	1.40
	Telephoto	SONY IMX613 (2x zoom)	12.2	1/3.40	1.00
iPhone 15 Pro	Main	SONY IMX803	48	1/1.28	1.22
	Telephoto	SONY IMX713 (2x/4x zoom)	12	1/3.40	1.00

Table 5: The mobile phone we apply to get the moiré patterns

Digital Screen	Size ($Inches$)	Panel type	Resolution	Brightness (cd/m^2)	Coating
DELL D2720DS	27	IPS(LED)	2560×1440	350	Matte
Macbook Air 2022	13.3	IPS(LED)	2560×1600	500	Glossy
AOC 27G2G8	27	IPS(W-LED)	2560×1440	250	Matte
Philips 27E1N5500	27	IPS(LED)	2560×1440	300	Matte
Xiaomi C34WQBA-RG	34	Curved SVA(W-LED)	3440×1440	300	Matte
ViewSonic VX2771-4K-HD	27	IPS(LED)	3840×2160	350	Matte

Table 6: The screen we apply to get the moiré patterns

Algorithm 1: Data Preprocessing in MPG.

Input: 4K moiré pattern dataset $\mathcal{D}_{mp}$, patch size (w, h).
Output: Selected moiré pattern patch I_{mp}.

while *True* **do**
  Randomly select a 4K moiré pattern $\mathcal{I} \in \mathcal{D}_{mp}$.
  for $i = 1$ *to* n **do**
    1. Multi-Scale Cropping:
      Randomly select probability p_1, p_2.
      if $p_1 \leq 50\%$ **then**
        $I_{mp} \leftarrow$ Random Crop$(\mathcal{I}, w, h)$.
      else if $p_2 \leq 33.33\%$ **then**
        $\mathcal{I} \leftarrow$ Resize$(\mathcal{I}, 2560, 1440)$,
        $I_{mp} \leftarrow$ Random Crop$(\mathcal{I}, w, h)$.
      else if $33.33\% < p_2 \leq 66.66\%$ **then**
        $\mathcal{I} \leftarrow$ Resize$(\mathcal{I}, 1920, 1080)$,
        $I_{mp} \leftarrow$ Random Crop$(\mathcal{I}, w, h)$.
      else $I_{mp} \leftarrow$ Resize$(\mathcal{I}, w, h)$.
    2. Sharpness-Colorfulness selection:
      $G_{mp} \leftarrow$ RGB_to_Gray(I_{mp}),
      $L_{mp}, A_{mp}, B_{mp} \leftarrow$ RGB_to_LAB(I_{mp}),
      Sharpness $\leftarrow \sigma(\mathcal{F} * G_{mp})$,
      Colorfulness $\leftarrow \sqrt{\sigma(A_{mp})^2 + \sigma(B_{mp})^2}$,
      if *Sharpness* $\geq \delta_s$ *and Colorfulness* $\geq \delta_c$ **then**
        return I_{mp}.
  end
end

The details of the data preprocessing and networking implementations of MPG are described below.

The implementation details of data preprocessing The details of our data preprocessing method in the Moiré Pattern Generator(MPG) are described in Algorithm 1. In Multi-Scale Cropping, "Random Crop$(\mathcal{I}, w, h)$" means to randomly crop a patch of size $w \times h$ from $\mathcal{I}$, and "Resize$(\mathcal{I}, w, h)$" means to resize the width and height of $\mathcal{I}$ to w and h directly. In Sharpness-Colorfulness selection, "RGB_to_Gray(I_{mp})" refers to convert I_{mp} to grayscale image G_{mp}, while "RGB_to_LAB(I_{mp})" refers to convert I_{mp} to LAB space and retrieve the corresponding channel matrices L_{mp}, A_{mp}, and B_{mp} respectively. Moreover, "$\mathcal{F} * G_{mp}$" denotes the convolution operation on the grayscale image G_{mp} using the Laplace edge detection operator $\mathcal{F}$. In the actual training process of MPG, we specify $n = 3$ and $w = h = 768$, while δ_s and δ_c are set to 15 and 2, respectively.

The implementation details of Latent Diffusion Model We utilize the Latent Diffusion Model(LDM) (Rombach et al. 2022) as the network component of the Moiré Pattern Generator.

Firstly, we have trained our autoencoder model for moiré patterns according to the method described in (Rombach et al. 2022). Specifically, given a moiré pattern patch $I_{mp} \in \mathrm{R}^{w \times h \times 3}$ that has gone through Multi-Scale Cropping and Sharpness-Colorfulness selection, we utilize an Encoder $\mathcal{E}$ to convert I_{mp} to the latent space $z = \mathcal{E}(I_{mp})$ through multiple downsampling blocks. Simultaneously, we expect the corresponding Decoder $\mathcal{D}$ to reconstruct the moiré pattern from the latent space variable $z : I_{mp} = \mathcal{D}(z) = \mathcal{D}(\mathcal{E}(I_{mp}))$ by using the same upsampling factor. Note that the overall downsampling factor is denoted as $f = h/h_0 = w/w_0$, where h_0 and w_0 are hyperparameters chosen to ensure that f is precisely 2^m, with $m \in \mathrm{N}$. Our loss function is a combination of a perceptual loss function $\mathcal{L}_{rec}$ (Zhang et al. 2018)

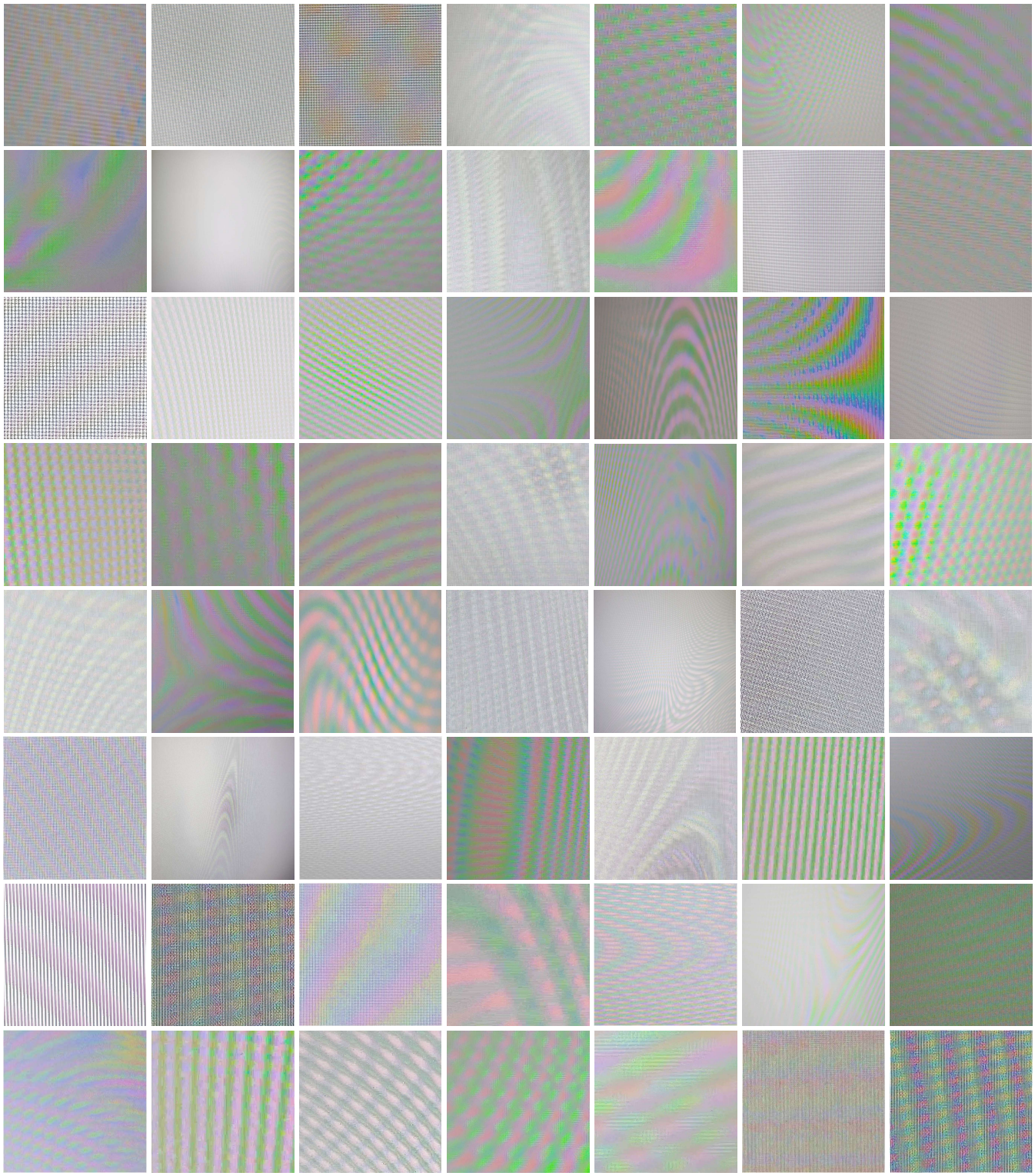

Figure 8: Visualization of sampled patches using our Moiré Pattern Generator.

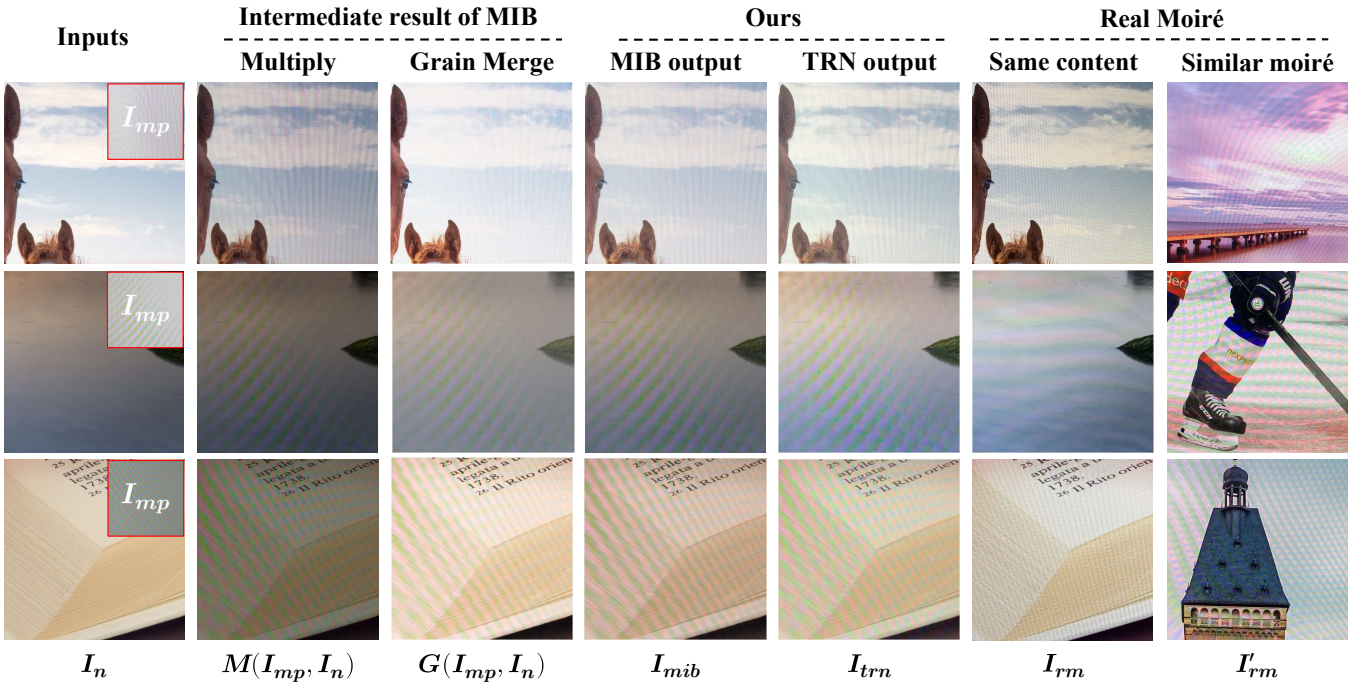

Figure 9: Visualization of our intermediate synthetic results. The final synthesis of I_{trn} best resembles the real moiré images in contrast and brightness distortions.

and patch-based adversarial targets $\mathcal{L}_{adv}$ (Dosovitskiy and Brox 2016; Esser, Rombach, and Ommer 2021; Yu et al. 2021), along with a KL-reg regularization term $\mathcal{L}_{reg}$ where the patch-based discriminator D_ψ we used is optimized to differentiate between the original moiré pattern I_{mp} and the reconstructed moiré pattern $\mathcal{D}(\mathcal{E}(I_{mp}))$. The full objective to train the autoencoder $(\mathcal{E}, \mathcal{D})$ is:

$$\mathcal{L} = \min_{\mathcal{E},\mathcal{D}} \max_{\psi} (\mathcal{L}_{rec}(I_{mp}, \mathcal{D}(\mathcal{E}(I_{mp}))) + \log D_\psi(I_{mp}) - \mathcal{L}_{adv}(\mathcal{D}(\mathcal{E}(I_{mp}))) + \mathcal{L}_{reg}(I_{mp}; \mathcal{E}, \mathcal{D})) \tag{14}$$

The network structure of our $\mathcal{E}$ and $\mathcal{D}$ are the same as the autoencoder in (Rombach et al. 2022). To compress the moiré pattern as much as possible, we use the downsampling factor $f = 32$ and the number of hidden-space channels 64, which gives the latent variable z a dimension of 64 $\times$ 64 $\times$ 24. We adopted 6 downsampling/upsampling blocks in $\mathcal{E}$ and $\mathcal{D}$. Each downsampling/upsampling block contains two layers of ResBlock as well as one layer of multi-head self-attention block, and the list of channel scaling multipliers is [1,1,2,2,4,4]. We trained the autoencoder on 8 NVIDIA A40 GPUs, with a batch size of 2 on each GPU and a learning rate of 4.5e-6. Both the autoencoder ($\mathcal{E}$ and $\mathcal{D}$) and the discriminator (D_ψ) in the loss functions are optimized by Adam (Kingma and Ba 2014) with β_1 = 0.5 and β_2 = 0.9. In total, we trained the autoencoder for 35 Epochs.

Subsequently, we adopt the diffusion model to modify the complex distribution $p(z)$ obeyed by the latent variable z after the $\mathcal{E}$ transformation with the objective function:

$$\mathcal{L} = \mathbb{E}_{\mathcal{E}(I_{mp}),\epsilon,t} \left[\|\epsilon - \epsilon_\theta \left(\alpha_t \mathcal{E} \left(I_{mp} \right) + \sigma_t \epsilon, t \right\|_1 \right] \tag{15}$$

where $\epsilon \sim \mathcal{N}(0, \mathbb{I})$ is the variable sampled by the standard Gaussian distribution, and ϵ_θ is the noisy prediction network parameterized by θ, where we implemented it by using UNet (Ronneberger, Fischer, and Brox 2015) which integrates the time-step conditioning variable t. The α_t is the value at step t of the signal-to-noise ratio.

The network structures of the diffusion model in latent space are the same as those of the unconditional model in LDM (Rombach et al. 2022). For the UNet model ϵ_θ, we set the number of channels to 192, and the encoder and decoder of the UNet contain 4 downsampling/upsampling blocks, and the structure of those blocks are kept the same in the autoencoder. The list of channel scaling multipliers is [1,2,4,8]. The model is trained on 8 NVIDIA A40 GPUs for 50 epochs and optimized by AdamW (Loshchilov and Hutter 2019) with β_1 = 0.9 and β_2 = 0.999. The batch size on each GPU is set to 2, and the learning rate is initially set to 1e-4 and scheduled by linear warmup on the first 10000 steps. The total learning rate complies with the linear multiplication in the LDM (Rombach et al. 2022) based on the number of GPUs and the batch size. We utilize the DDIM sampler (Song, Meng, and Ermon 2022) to accelerate sampling after training, using 200 sampling steps. We sampled 100000 moiré patterns using a single NVIDIA A40 GPU, and some of the samples are shown in Figure 8.

B.2 Moiré Image Synthesis

In this section, we will demonstrate the implementation details of the Moiré Image Synthesis stage that were omitted in our main paper. Additionally, we include more visualizations of the synthesis results in Figure 9.

Figure 10: Examples of the "Checkerboard Artifacts" that occur in the I_{trn} when upsampling with Uformer's transpose convolution (Wang et al. 2022).

Implementations of the Moiré Image Blending For the MIB module, ω_m in Eq. (5) is randomly selected from [0.65, 0.75], while $\omega_g = 1-\omega_m$. The op_m and op_g in Eq. (6) are set to 1.0 and 0.8, respectively. Performance changes resulting from the use of both the Multiply and Grain Merge strategy are detailed in the additional ablation study in Section C.3.

Implementations of the Tone Refinement Network We implement the backbone of our Tone Refinement Network(TRN) using Uformer-T(Tiny) (Wang et al. 2022), where the Transformer Block uses the Locally-enhanced Window (LeWin) Transformer block proposed by Uformer and sets the window size to 8×8. At the same time, we change the encoder depth from {2,2,2,2} to {1,1,1,1}. Performance changes resulting from the use of the Uformer are detailed in the additional ablation study in Section C.3.

In the context of TRN, utilizing the transposed convolutional upsampling block similar to Uformer may lead to the emergence of "Checkerboard Artifacts" in the output I_{trn}, as illustrated in Figure 10. This issue stems from the uneven overlap when transposed convolution is employed during the upsampling process (Odena, Dumoulin, and Olah 2016). As a solution, we utilize the CARAFE upsampling operator (Wang et al. 2019) to replace transposed convolution in Uformer for upsampling. CARAFE effectively addresses the "Checkerboard Artifacts" by predicting diverse up-sampling kernels based on the semantic information of the input feature maps (Wang et al. 2019), thereby contributing to improved feature reorganization within TRN. Performance changes resulting from the use of CARAFE are detailed in the additional ablation study in Section C.3.

We utilize 2 NVIDIA A40 GPUs to train our Tone Refinement Network on UHDM (Yu et al. 2022) for 50 epochs, FHDMi (He et al. 2020) for 25 epochs, and TIP (Sun, Yu, and Wang 2018) for 2 epochs. The learning rate is initially set to 1e-5 and scheduled by cyclic cosine annealing (Loshchilov and Hutter 2016), and models are optimized by Adam (Kingma and Ba 2014) with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. For the input clean natural images, we set the random crop size to 384 × 384, and for the moiré patterns sampled by the Moiré Pattern Generator, we resized their resolution to 384 × 384 as well.

Implementations of loss functions For the loss function Eq. (13) in the main paper, we simply set $\lambda_{per} = \lambda_{color} = 1.0$ and $\lambda_{tv} = 0.1$ to balance the scale of the values during training.

For the perception loss L_{per}, to further validate our assumption in the main paper whether computing the content loss between I_{trn} and I_{mib} directly in the pixel space is less effective, we utilize the $\mathcal{L}_1$ loss instead of the $\mathcal{L}_{per}$ loss for our synthesis network in the additional ablation study in Section C.3.

For the color loss $\mathcal{L}_{color}$, we convert I_{trn} and I_{rm} into RGB-uv histogram feature $H(I_{trn})$ and $H(I_{rm})$ from the log-chrominance space followed by prior work on color constancy (Afifi and Brown 2019; Afifi et al. 2019), which represents the color distribution of those two images. In particular, u and v are used to control the contribution of each color channel in the generated histogram and the smoothness of the histogram bin. Specifically, given an RGB image $I(\mathrm{x})$ where x denotes the pixel point index, we first convert it to

YUV color space:

$$I_y(\mathrm{x}) = \sqrt{I_r^2(\mathrm{x}) + I_g^2(\mathrm{x}) + I_b^2(\mathrm{x})}. \tag{16}$$

and:

$$I_{ur}(i) = \log \frac{I_r(i)+\epsilon}{I_g(i)+\epsilon}; I_{vr}(i) = \log \frac{I_r(i)+\epsilon}{I_b(i)+\epsilon} \tag{17}$$

$$I_{ug}(i) = \log \frac{I_g(i)+\epsilon}{I_r(i)+\epsilon}; I_{vg}(i) = \log \frac{I_g(i)+\epsilon}{I_b(i)+\epsilon} \tag{18}$$

$$I_{ub}(i) = \log \frac{I_b(i)+\epsilon}{I_r(i)+\epsilon}; I_{vb}(i) = \log \frac{I_b(i)+\epsilon}{I_g(i)+\epsilon} \tag{19}$$

where "I_r", "I_g", and "I_b" subscripts refer to the color channels of the image I, ϵ is a small constant added for numerical stability, and (I_{ur}, I_{vr}), (I_{ug}, I_{vg}) and (I_{ub}, I_{vb}) are the uv coordinates of the I_r, I_g, and I_b.

We then generated the unnormalized histogram $H(u, v, c)$ of each color channel $c \in \{r, g, b\}$ according to the HistoGAN (Afifi, Brubaker, and Brown 2021), computed as follows:

$$H(u, v, c) \propto \sum_{\mathrm{x}} k\left(I_{uc}(\mathrm{x}), I_{vc}(\mathrm{x}), u, v\right) I_y(\mathrm{x}), \tag{20}$$

where $k(\cdot)$ is the inverse-quadratic kernel:

$$\begin{aligned} k\left(I_{uc}, I_{vc}, u, v\right) = &\left(1 + \left(\left|I_{uc} - u\right|/\tau\right)^2\right)^{-1} \\ &\times \left(1 + \left(\left|I_{vc} - v\right|/\tau\right)^2\right)^{-1} \end{aligned} \tag{21}$$

where τ is a fall-off parameter to control the smoothness of the histogram's bins. Finally, the histogram features $H(I) \in R^{h \times h \times 3}$ stacked by $H(u, v, c)$ of 3 color channels is normalized to sum to one:

$$H(I) = [\frac{H(u,v,r)}{\sum H(u,v,r)}, \frac{H(u,v,g)}{\sum H(u,v,g)}, \frac{H(u,v,b)}{\sum H(u,v,b)}]. \tag{22}$$

C Experiments

In this section, we will provide a more detailed overview of the experimental setups, present additional visualization results, carry out further ablation studies, and address the limitations of our proposed method.

C.1 Experimental Setups

We implement all the experiments using PyTorch Lightning on multiple NVIDIA A40 GPUs.

Implementation Details of other Comparison Methods For Shooting, we migrated their implementation code from Opencv to PyTorch based on the implementation idea provided by (Niu, Guo, and Wang 2021). Note that the Shooting method produces a distorted composite image after random projective transformation. We maintain the transformation parameter and adjust the clean image accordingly to ensure that the moiré image aligns with the clean image during the subsequent demoireing stage. For UnDeM (Zhong et al. 2024), we directly use their 384×384 moiré image synthesis network trained on UHDM (Yu et al. 2022) and FHDMi (He et al. 2020) and also train their synthesis network on TIP (Sun, Yu, and Wang 2018) in their code framework (Zhong et al. 2024). For MoireSpace (Yang et al. 2023), we utilize the moiré patterns provided by their dataset to obtain the synthesis result by deploying their multiply blending strategy. We resize their moiré patterns to 384×384 for a fair comparison.

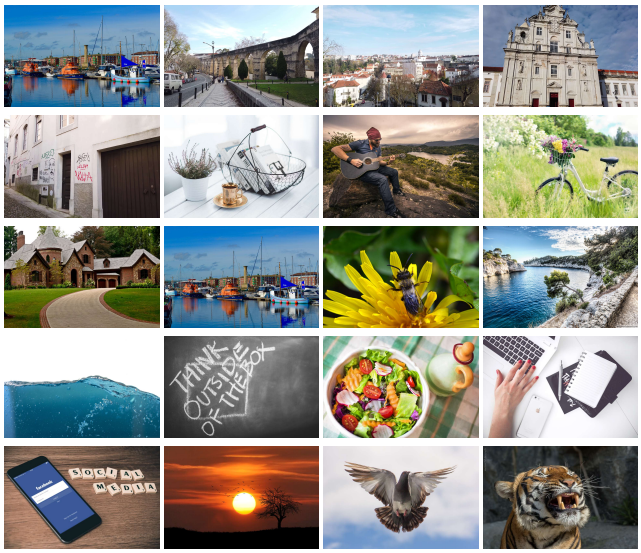

Figure 11: Examples of the MHRNID dataset.

Mixed High-Resolution Natural Image Dataset In the Zero-Shot experiments, we collected a comprehensive Mixed High-Resolution Natural Image Dataset (MHRNID) to avoid data overlap between the training and test sets. The MHRNID dataset consists of the super-resolution datasets DF2K-OST (Wang et al. 2021), the natural image datasets UHD-LOL4K (Wang et al. 2023b), and UHD-IQA (Hosu et al. 2024) collated and incorporated, which contains 26,000 high-definition images. We also provide several visual examples of MHRNID, as shown in Figure 11.

Implementation Details of Demoiréing Models For MBCNN (Zheng et al. 2020) and ESDNet-L (Yu et al. 2022), we followed the experimental settings from (Yu et al. 2022) and (Zhong et al. 2024). We trained for 150 epochs on UHDM (Yu et al. 2022) and FHDMi (He et al. 2020) and 70 epochs on TIP (Sun, Yu, and Wang 2018). Additionally, we trained for 50 epochs on the MHRNID dataset.

C.2 More Qualitative Comparisons

Moiré Image Synthesis The visualization results of synthesis moiré images on the MHRNID dataset using Shooting (Niu, Guo, and Wang 2021), UnDeM (Zhong et al. 2024), and our UniDemoiré are shown in Figure 12. The moiré image produced by our UniDemoiré is notably superior to other synthesis methods in terms of diversity and realism. In comparison, the moiré image generated by the Shooting (Niu, Guo, and Wang 2021) method is excessively

distorted, UnDeM's network (Zhong et al. 2024) is susceptible to anomalies during image generation, and the moiré pattern dataset provided by MoireSpace (Yang et al. 2023) is of subpar quality. Additionally, the multiplication strategy results in a darker synthesized image.

Demoiréing Figure 13 shows the visualization results of zero-shot demoiréing on UHDM (Yu et al. 2022). Additionally, Figures 14 and 15 illustrate the demoiréing results on FHDMi (He et al. 2020) and TIP (Sun, Yu, and Wang 2018) using ESDNet-L (Yu et al. 2022) trained on UHDM (Yu et al. 2022). Our method's model effectively removes moiré artifacts and retains high-frequency details, indicating the strong generalization ability of our proposed UniDemoiré.

C.3 Additional Ablation Study

The results of the additional ablation experiments are in Table 7. where "$\mathcal{L}_{per} \rightarrow \mathcal{L}_1$" denotes replacing the perception loss $\mathcal{L}_{per}$ in the synthesis network with the L1 loss $\mathcal{L}_1$. "Uformer $\rightarrow$ UNet" denotes switching the entire backbone network of the TRN from Uformer to UNet (Ronneberger, Fischer, and Brox 2015).

Components	PSNR↑	SSIM↑	LPIPS↓
ALL	**20.7543**	**0.7653**	**0.2136**
MIB (w/o Multiply)	20.3158	0.7598	0.2328
MIB (w/o Grain Merge)	20.3930	0.7587	0.2414
TRN (w/o CARAFE)	20.4414	0.7408	0.2256
TRN ($\mathcal{L}_{per} \rightarrow \mathcal{L}_1$)	20.1404	0.7447	0.2495
TRN (Uformer $\rightarrow$ UNet)	20.3899	0.7476	0.2413

Table 7: Additional ablation studies. Source: UHDM, Target: FHDMi.

The results of two sets of ablation experiments on layer blending strategies also show that using only one of them leads to distortion of the synthesis results, which in turn affects the model's generalization ability. The results of the "$\mathcal{L}_{per} \rightarrow \mathcal{L}_1$" show that computing the loss function in this way leads to a degradation of the model performance because moiré patterns can disrupt image structures by generating strip-shaped artifacts. The results of the "w/o CARAFE" indicate that using the CARAFE upsampling operator (Wang et al. 2019) yields better fusion performance than the transposed convolution originally employed by Uformer (Wang et al. 2022). Furthermore, the results from the "Uformer $\rightarrow$ UNet" demonstrate that the LeWin Transformer Block within Uformer is more effective at extracting color features from moiré patterns compared to the original UNet architecture.

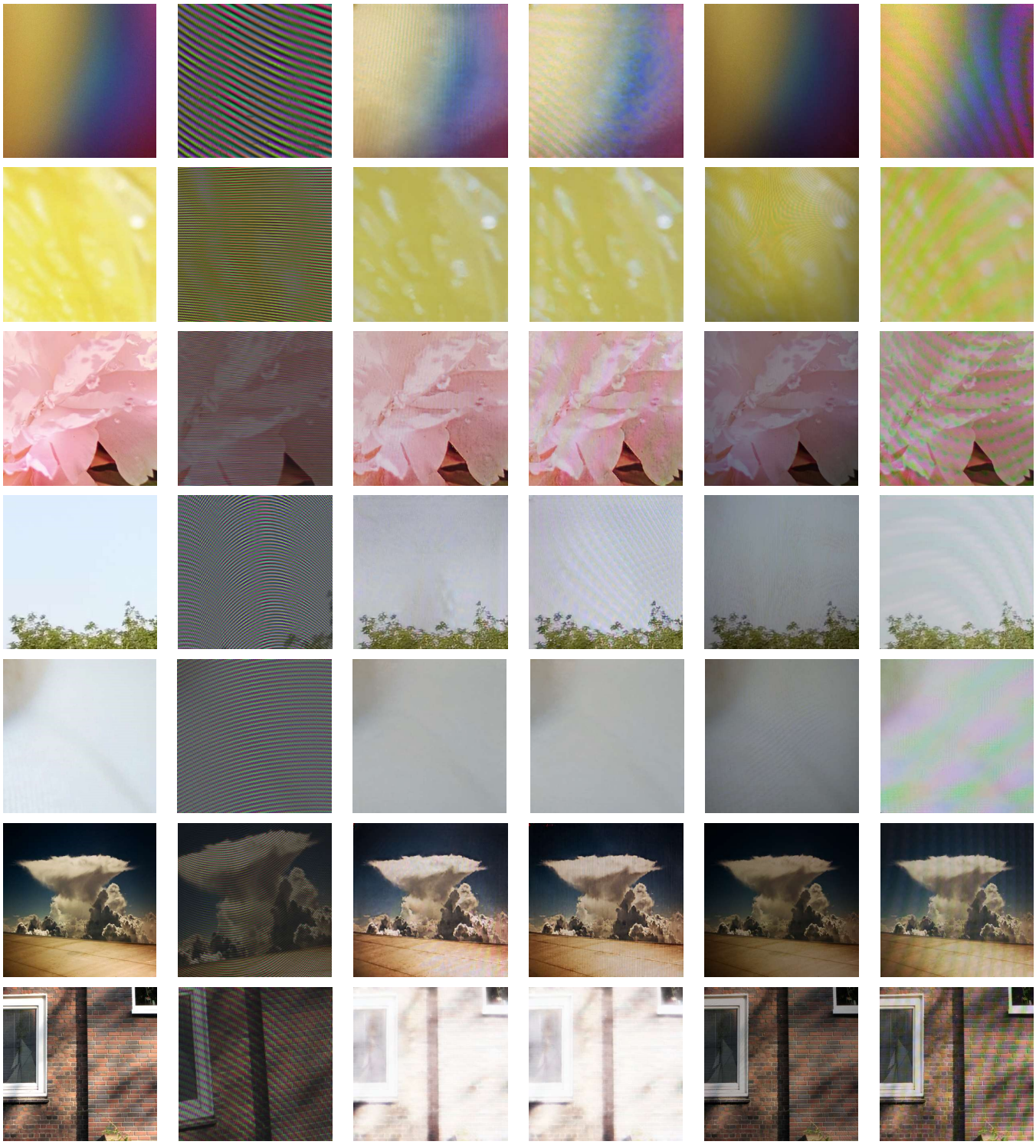

Figure 12: Qualitative comparisons of synthesized moire images were obtained using the shooting method, UnDeM, MoireSpace, and our UniDemoiré.

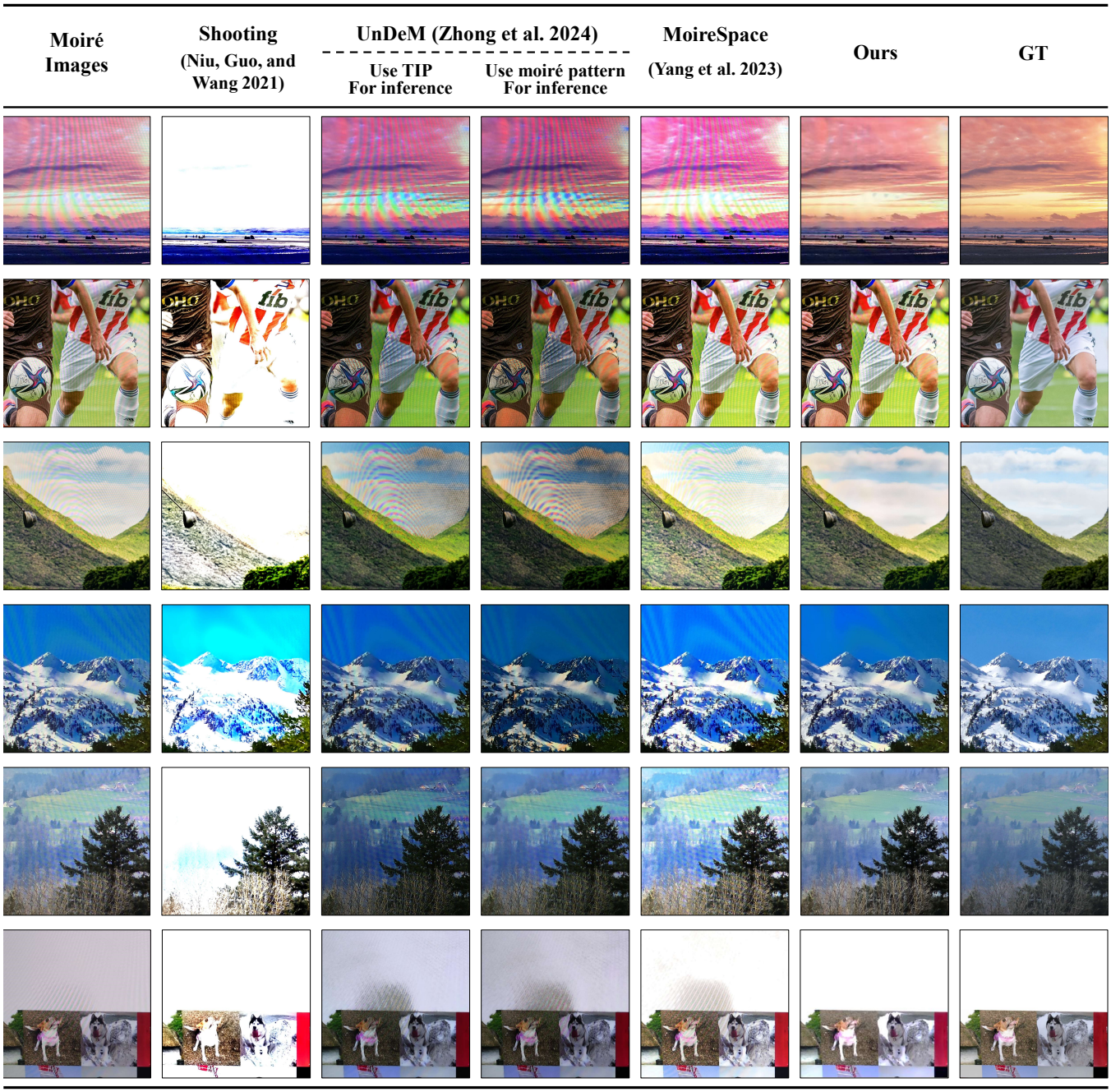

Figure 13: Qualitative comparisons of zero-shot evaluation on the UHDM dataset.

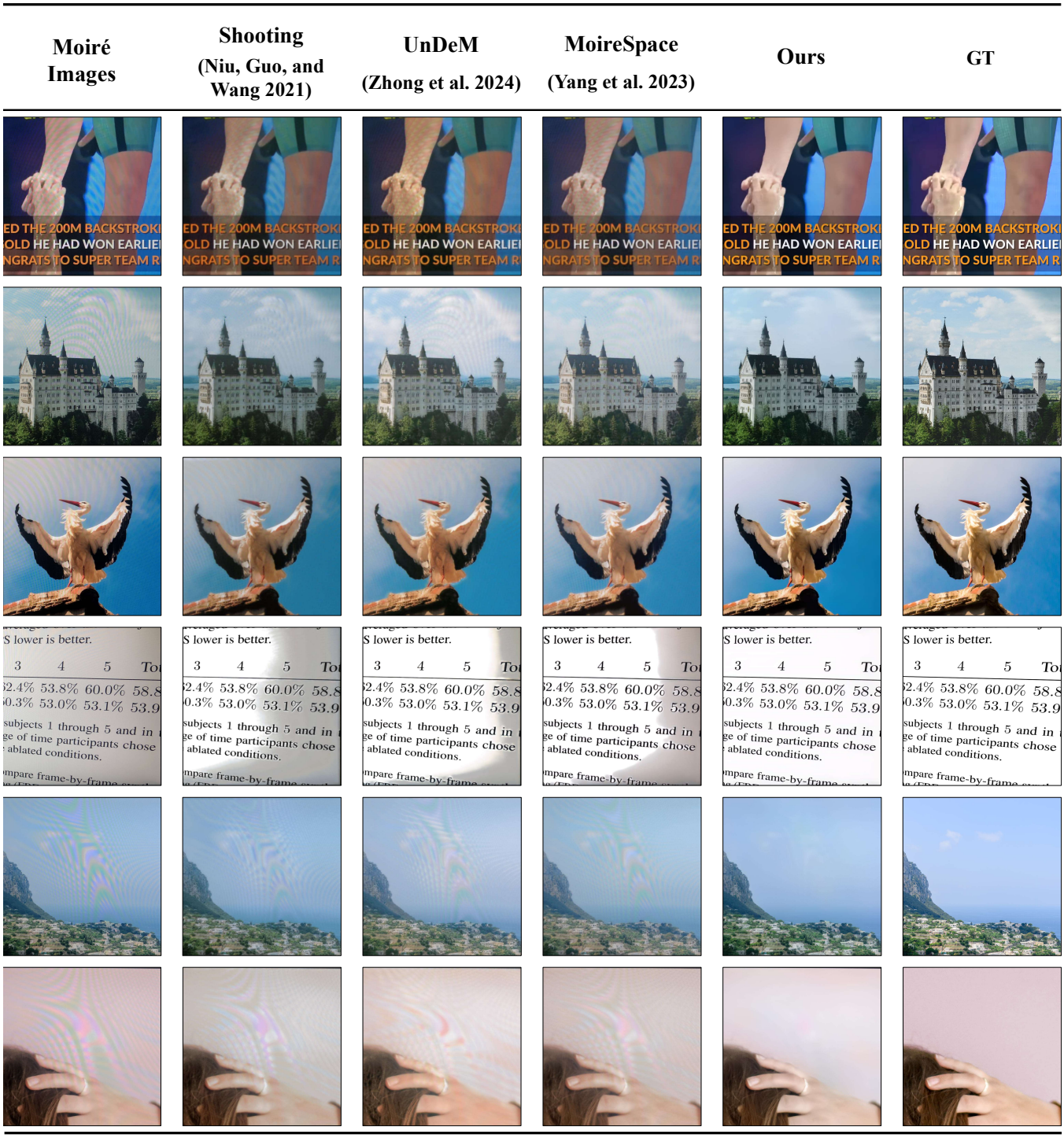

Figure 14: Qualitative comparisons of our models with other state-of-the-art methods on the FHDMi dataset.

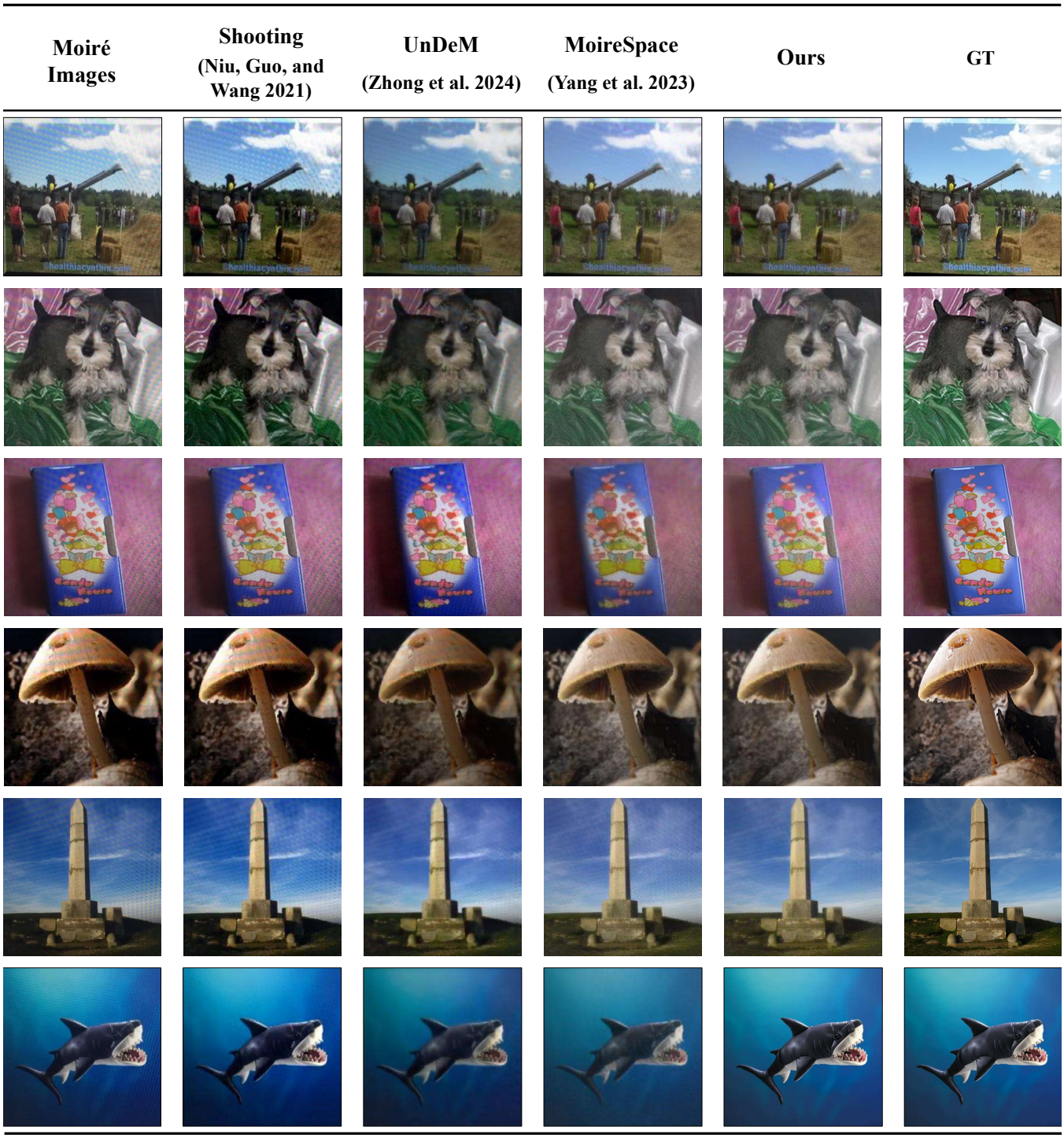

Figure 15: Qualitative comparisons of our models with other state-of-the-art methods on the TIP dataset.

References

Afifi, M.; and Brown, M. S. 2019. Sensor-Independent Illumination Estimation for DNN Models. arXiv:1912.06888.

Afifi, M.; Brubaker, M. A.; and Brown, M. S. 2021. HistoGAN: Controlling Colors of GAN-Generated and Real Images via Color Histograms. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Afifi, M.; Price, B.; Cohen, S.; and Brown, M. S. 2019. When Color Constancy Goes Wrong: Correcting Improperly White-Balanced Images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Dosovitskiy, A.; and Brox, T. 2016. Generating Images with Perceptual Similarity Metrics based on Deep Networks. *Neural Information Processing Systems,Neural Information Processing Systems*.

Esser, P.; Rombach, R.; and Ommer, B. 2021. Taming Transformers for High-Resolution Image Synthesis. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

He, B.; Wang, C.; Shi, B.; and Duan, L.-Y. 2020. FHDe 2 Net: Full High Definition Demoireing Network. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*, 713–729. Springer.

Hosu, V.; Agnolucci, L.; Wiedemann, O.; and Iso, D. 2024. UHD-IQA Benchmark Database: Pushing the Boundaries of Blind Photo Quality Assessment. arXiv:2406.17472.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Loshchilov, I.; and Hutter, F. 2016. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*.

Loshchilov, I.; and Hutter, F. 2019. Decoupled Weight Decay Regularization. arXiv:1711.05101.

Niu, D.; Guo, R.; and Wang, Y. 2021. Morié attack (ma): A new potential risk of screen photos. *Advances in Neural Information Processing Systems*, 34: 26117–26129.

Odena, A.; Dumoulin, V.; and Olah, C. 2016. Deconvolution and Checkerboard Artifacts. *Distill*.

Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; and Ommer, B. 2022. High-Resolution Image Synthesis with Latent Diffusion Models. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ronneberger, O.; Fischer, P.; and Brox, T. 2015. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, 234–241. Springer.

Song, J.; Meng, C.; and Ermon, S. 2022. Denoising Diffusion Implicit Models. arXiv:2010.02502.

Sun, Y.; Yu, Y.; and Wang, W. 2018. Moiré photo restoration using multiresolution convolutional neural networks. *IEEE Transactions on Image Processing*, 27(8): 4160–4172.

Wang, J.; Chen, K.; Xu, R.; Liu, Z.; Loy, C. C.; and Lin, D. 2019. CARAFE: Content-Aware ReAssembly of FEatures. arXiv:1905.02188.

Wang, T.; Zhang, K.; Shen, T.; Luo, W.; Stenger, B.; and Lu, T. 2023. Ultra-high-definition low-light image enhancement: A benchmark and transformer-based method. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 2654–2662.

Wang, X.; Xie, L.; Dong, C.; and Shan, Y. 2021. Real-ESRGAN: Training Real-World Blind Super-Resolution with Pure Synthetic Data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 1905–1914.

Wang, Z.; Cun, X.; Bao, J.; Zhou, W.; Liu, J.; and Li, H. 2022. Uformer: A General U-Shaped Transformer for Image Restoration. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yang, C.; Yang, Z.; Ke, Y.; Chen, T.; Grzegorzek, M.; and See, J. 2023. Doing More With Moiré Pattern Detection in Digital Photos. *IEEE Transactions on Image Processing*, 32: 694–708.

Yu, J.; Li, X.; Koh, J.; Zhang, H.; Pang, R.; Qin, J.; Ku, A.; Xu, Y.; Baldridge, J.; and Wu, Y. 2021. Vector-quantized Image Modeling with Improved VQGAN. *Cornell University - arXiv,Cornell University - arXiv*.

Yu, X.; Dai, P.; Li, W.; Ma, L.; Shen, J.; Li, J.; and Qi, X. 2022. Towards efficient and scale-robust ultra-high-definition image demoiréing. In *European Conference on Computer Vision*, 646–662. Springer.

Zhang, R.; Isola, P.; Efros, A. A.; Shechtman, E.; and Wang, O. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 586–595.

Zheng, B.; Yuan, S.; Slabaugh, G.; and Leonardis, A. 2020. Image demoireing with learnable bandpass filters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3636–3645.

Zhong, Y.; Zhou, Y.; Zhang, Y.; Chao, F.; and Ji, R. 2024. Learning Image Demoireing from Unpaired Real Data. *arXiv preprint arXiv:2401.02719 (AAAI2024)*.