When Schrödinger Bridge Meets Real-World Image Dehazing with Unpaired Training

Supplementary Material

1. Introduction

Our supplementary materials offer additional details and experimental results that further support our method. These can be summarized as follows:

- We provide detailed information of the training process and our method;
- We conduct further ablation studies to validate the effectiveness of our method;
- We illustrate more qualitative and quantitative results to demonstrate the superior performance of the proposed method.

2. Detail information of the training process

2.1. Estimation of Entropy

Following the UNSB [4], we employ mutual information to estimate the entropy of a general random variable as follows:

$$I(X, Y) = H(X) - H(X \mid Y), \tag{1}$$

where $H(X)$ denotes the entropy of X, $I(X, X)$ denotes the mutual information, and $H(X \mid Y)$ is the conditioned entropy. When we set Y to X, Eq. 1 can be transformed into:

$$I(X, X) = H(X) - H(X \mid X), \tag{2}$$

where $H(X \mid X) = 0$ since knowing X leaves no uncertainty about X. Thus, the mutual information $I(X, X)$ equals the entropy $H(x)$.

Inspired by the work [1], we employ a neural network T_θ parametered by $\theta \in \Theta$ to approximate mutual information to arbitrary accuracy as follows:

$$I_\Theta(X, Z) := \sup_{\theta \in \Theta} \left(\mathbb{E}_{P_{XZ}}[T_\theta] - \log \mathbb{E}_{P_X \otimes P_Z}[e^{T_\theta}] \right). \tag{3}$$

Following this paradigm, we estimate the entropy $H(q_\theta(x(t_i), x_1))$ by setting $X = Z$.

2.2. Network architecture

In this section, we provide a detailed description of the network architecture. Specifically, we employ a UNet used in [3] as our generator $G : q_\theta(x_1 \mid x(t_i))$. Differently, our generator does not utilize dynamic snake convolution, as it does not enhance the ability to capture fine details in hazy images. Moreover, we take the time step t_i as input along with $x(t_i)$ since our generator shares a parameter for all time steps t_i.

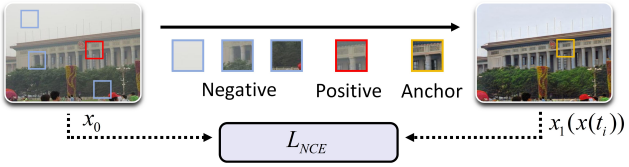

Figure 1. Illustration of PatchNCE regularization.

For the discriminator, we employ a Markovian discriminator from [4] as the local discriminator and a CLIP-based discriminator [5] as the global discriminator. Additionally, we employ a UNet employed in [6] as the transmission map refined network. The architecture of T_θ, which is used to estimate entropy, is the same as the local discriminator.

2.3. PatchNCE Regularization

As illustrated in Figure 1, a representative patch from the dehazed image (denoted by a yellow border) serves as an anchor, while its spatially corresponding patch in the hazy input (marked with a red border) is designated as the positive sample. Concurrently, all non-corresponding patches within the hazy image (highlighted with blue borders) are treated as negative samples. This contrastive formulation effectively maintains structural fidelity by encouraging the generator to preserve original image textures while removing haze artifacts.

2.4. High-Frequency Detail Regularization

We introduce high-frequency detail regularization, including Discrete Fourier Transform loss, SSIM loss, and Sobel Gradient loss. Specifically, we convert both hazy and dehazed images into the frequency domain through Discrete Fourier Transform (DFT), and extract high-frequency components. These components are then transformed back to the spatial domain via Inverse DFT (IDFT), establishing a high-frequency regularization that ensures the high-frequency details of the dehazed image match those of the original image. Moreover, we employ SSIM [2] and Sobel Gradient loss to ensure that the restored details remain structurally aligned with the source image's inherent characteristics. The Discrete Fourier Transform loss and Sobel Gradient loss can be expressed as:

$$L_F = L_2(F(x_0), F(x_1(x(t_i)))), \tag{4}$$

$$L_S = L_2(S(x_0), S(x_1(x(t_i)))), \tag{5}$$

where F and S denote the Discrete Fourier Transform and Sobel Gradient operation. x_0 and $x_1(x(t_i))$ denote the hazy

Intervals	1	3	5	10
FID ↓	75.339	73.171	69.796	**66.541**
NIQE ↓	3.895	3.789	**3.743**	3.751
MUSIQ ↑	56.764	55.880	**59.256**	58.062
MANIQA ↑	0.134	0.131	**0.150**	0.132
PSNR ↑	18.284	18.508	**18.829**	18.617
SSIM ↑	0.820	0.825	0.838	**0.839**
LPIPS ↓	0.293	0.276	**0.223**	0.251
VSI ↑	0.948	0.948	**0.961**	0.958

Table 1. Ablation study of number of intervals.

image and generated dehazed image. The use of SSIM loss is the same as in reference [3].

2.5. Training Details

We implement our method within the Pytorch framework using Python 3.10, utilizing the Adam optimizer with a batch size of 4 for network training. We train our framework for 100K iterations, with β_1 set to 0.9, β_2 set to 0.999, and a learning rate of 2×10^{-5}. All experiments are conducted on a single 3090 GPU. The training sample is resized to 256×256, and we implement horizontal flipping for data augmentation. For SB training and simulation, we discretize the unit interval [0, 1] into 5 intervals with uniform spacing. Additionally, τ is set to 0.01, λ_{SB} is set to 1, λ_p is set to 1, λ_{NCE} is set to 1, λ_{phy} is set 0.5, and λ_{hfd} is set to 0.5.

3. Ablation Study

3.1. Ablation Study of Number of Intervals

To investigate the impact of the number of intervals, we conduct experiments with varying numbers of intervals. As shown in Table 1, setting this parameter to 5 yields the best overall quality performance. Additionally, compared to setting it to 10, using 5 intervals results in a reduction in training costs, making it a more efficient choice.

3.2. Ablation Study of Weight

We conduct a series of ablation studies to validate the impact of varying weights of detail-preserving regularization, including physical prior regularization (PPR) and high-frequency detail regularization (HFDR). It is important to note that PatchNCE regularization is essential in our framework. As illustrated in Figure 2 and Figure 3, while different weight settings may outperform our method on certain metrics, from the perspective of overall image quality, setting the weights of PPR and HFDR to 0.5 yields the most suitable results.

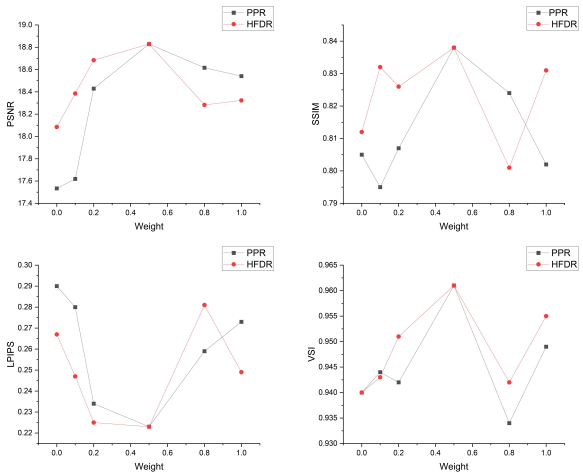

Figure 2. Ablation study of Weight.

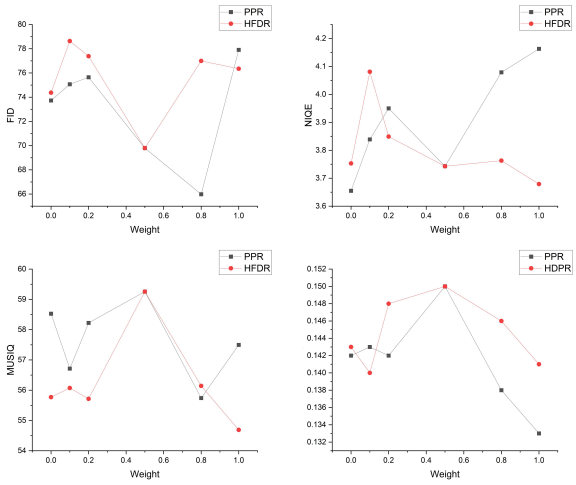

Figure 3. Ablation study of Weight.

3.3. Ablation study of NFEs

The trained generator can produce dehazed images from any given $x(t_i)$. This capability allows us to iteratively generate dehazed images with varying numbers of function evaluations (NFEs) using the trained generator and Eq. 6.

$$\begin{aligned} p(x(t_{j+1}) \mid x(t_j), x_1(x(t_j))) \sim \\ \mathcal{N}(x(t_{j+1}); \mu(t_{j+1}), \sigma(t_{j+1})), \\ \mu(t_{j+1}) = s(t_{j+1})x_1(x(t_j)) + (1 - s(t_{j+1}))x(t_j), \\ \sigma(t_{j+1}) = s(t_{j+1})(1 - s(t_{j+1}))\tau(1 - t_j)I. \end{aligned} \tag{6}$$

Consequently, we investigate the relationship between NFEs and the quality of the generated images. While a higher NFE is generally more advantageous for image translation tasks, as validated in UNSB [4], tasks requiring strict pixel-wise alignment, such as image dehazing, may suf-

Figure 4. Ablation study of NFEs.

	URHI				IHAZE			
Method	FID↓	NIQE↓	MUSIQ↑	MANIQA↑	PSNR↑	SSIM↑	LPIPS↓	VSI↑
DCP	63.701	4.129	55.966	0.143	11.055	0.592	0.374	0.930
MBFormer	55.023	4.174	57.293	<u>0.163</u>	16.092	0.776	0.232	0.939
C2PNet	55.970	4.368	56.542	0.161	16.427	<u>0.780</u>	0.234	0.944
KANet	56.921	3.839	57.774	0.144	<u>16.667</u>	0.759	0.435	0.953
DEANet	56.606	4.233	56.306	0.155	10.191	0.459	0.420	0.913
Diff-Plugin	54.669	4.555	53.913	0.122	15.513	0.756	0.267	0.943
OneRestore	56.011	4.476	53.698	0.131	16.537	**0.788**	<u>0.224</u>	0.942
SGDN	65.937	5.225	46.817	0.100	16.074	0.775	0.239	**0.959**
CUT	53.511	4.299	58.224	0.147	14.456	0.646	0.379	0.919
UNSB	<u>47.294</u>	4.232	55.064	0.153	16.486	0.649	0.360	0.936
YOLY	60.651	4.289	54.896	0.147	15.555	0.704	0.447	0.943
RefineDNet	59.701	**3.668**	<u>58.540</u>	0.138	16.571	0.765	0.291	0.945
D4	58.877	4.372	55.958	0.160	13.844	0.605	0.281	0.940
D4+	59.743	3.942	55.807	0.147	15.072	0.726	0.299	0.950
Ours	**46.613**	<u>3.813</u>	**60.865**	**0.170**	**16.764**	0.766	**0.201**	<u>0.951</u>

Table 2. Quantitative results on URHI and IHAZE. The best results are denoted in **bold**, and the second-best results are <u>underlined</u>.

fer from decreased fidelity with increased NFE. As shown in Figure 4 and Figure 5, increasing the NFE can lead to greater distortion, thereby reducing the overall quality of the dehazed images. To strike a balance between perceptual quality and distortion in the generated dehazed images, we opt for an NFE value of 1. This choice ensures optimal fidelity and minimizes unwanted distortions, resulting in high-quality dehazed images.

4. Experimental Results

In this section, we provide additional qualitative and quantitative results to further demonstrate the superior performance of the proposed method.

Table 2 illustrates the quantitative results of URHI and IHAZE. Figures 6, 7, 8, 9, 10, 11, and 12 present visual comparisons with several state-of-the-art methods on Fattal's dataset, Haze2020, RTTS, and URHI. As shown, the proposed method achieves satisfactory performance, producing high-quality images with natural color and high contrast. For IHAZE, our method does not achieve optimal performance on all metrics. This is due to the fact that IHAZE is an indoor hazy dataset generated using a haze machine,

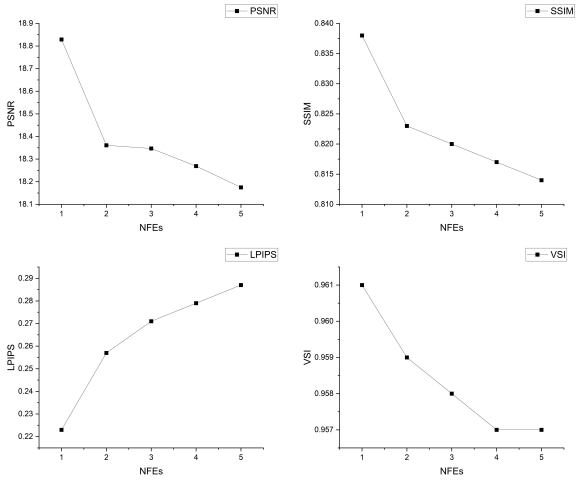

Figure 5. Ablation study of NFEs.

Method	Params↓	FLOPs↓	mAP↑
DCP	-	-	0.6489
MBFormer	77.43M	88.1G	0.6480
C2PNet	7.17M	352.9G	0.6462
KANet	55.25M	4.4G	0.6441
DEANet	**3.65M**	34.1G	0.6429
Diff-Plugin	942.61M	1.8T	0.6255
OneRestore	5.98M	11.3G	0.6418
SGDN	10.97M	52.9G	0.6395
CUT	11.37M	64.1G	0.4618
UNSB	14.68M	62.7G	0.6029
YOLY	-	-	0.6122
RefineDNet	65.80M	75.4G	0.6400
D4	10.70M	**2.2G**	0.6398
D4+	10.70M	**2.2G**	0.6425
Ours	15.83M	15.8G	**0.6506**

Table 3. Quantitative results of model efficiency.

while our training data consists exclusively of outdoor images. Given that haze predominantly occurs in outdoor environments in real-world scenarios, we believe our method remains the most effective for processing real-world hazy images.

Moreover, we compare the proposed method with the others in terms of model parameters, and FLOPs (Table 3). We do not include DCP and YOLY in this comparison because DCP is a prior-based method, and YOLY is a training-free method that requires hundreds of iterations to generate a dehazed image, taking several minutes. As evident from the table, although not the most optimal method in terms of efficiency, our model's number of parameters and FLOPs surpass those of most other approaches. Our approach ensures the best performance while achieving efficient dehazing. Additionally, we show the detection results of RTTS detected by YOLO, validating that our dehazed images perform well in downstream tasks. This demonstrates the practical benefits of our method in real-world applications.

References

[1] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In *International conference on machine learning*, pages 531–540. PMLR, 2018. 1

[2] Dominique Brunet, Edward R Vrscay, and Zhou Wang. On the mathematical properties of the structural similarity index. *IEEE Transactions on Image Processing*, 21(4):1488–1499, 2011. 1

[3] Xuanzhao Dong, Vamsi Krishna Vasa, Wenhui Zhu, Peijie Qiu, Xiwen Chen, Yi Su, Yujian Xiong, Zhangsihao Yang, Yanxi Chen, and Yalin Wang. Cunsb-rfie: Context-aware unpaired neural schrodinger bridge in retinal fundus image enhancement. In *Proceedings of the Winter Conference on Applications of Computer Vision (WACV)*, pages 4502–4511, 2025. 1, 2

[4] Beomsu Kim, Gihyun Kwon, Kwanyoung Kim, and Jong Chul Ye. Unpaired image-to-image translation via neural schrödinger bridge. In *The Twelfth International Conference on Learning Representations*, 2024. 1, 2

[5] Nupur Kumari, Richard Zhang, Eli Shechtman, and Jun-Yan Zhu. Ensembling off-the-shelf models for gan training. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10651–10662, 2022. 1

[6] Shiyu Zhao, Lin Zhang, Ying Shen, and Yicong Zhou. Refinednet: A weakly supervised refinement framework for single image dehazing. *IEEE Transactions on Image Processing*, 30:3391–3404, 2021. 1

Figure 6. Visual comparison of samples from Fattal's dataset.

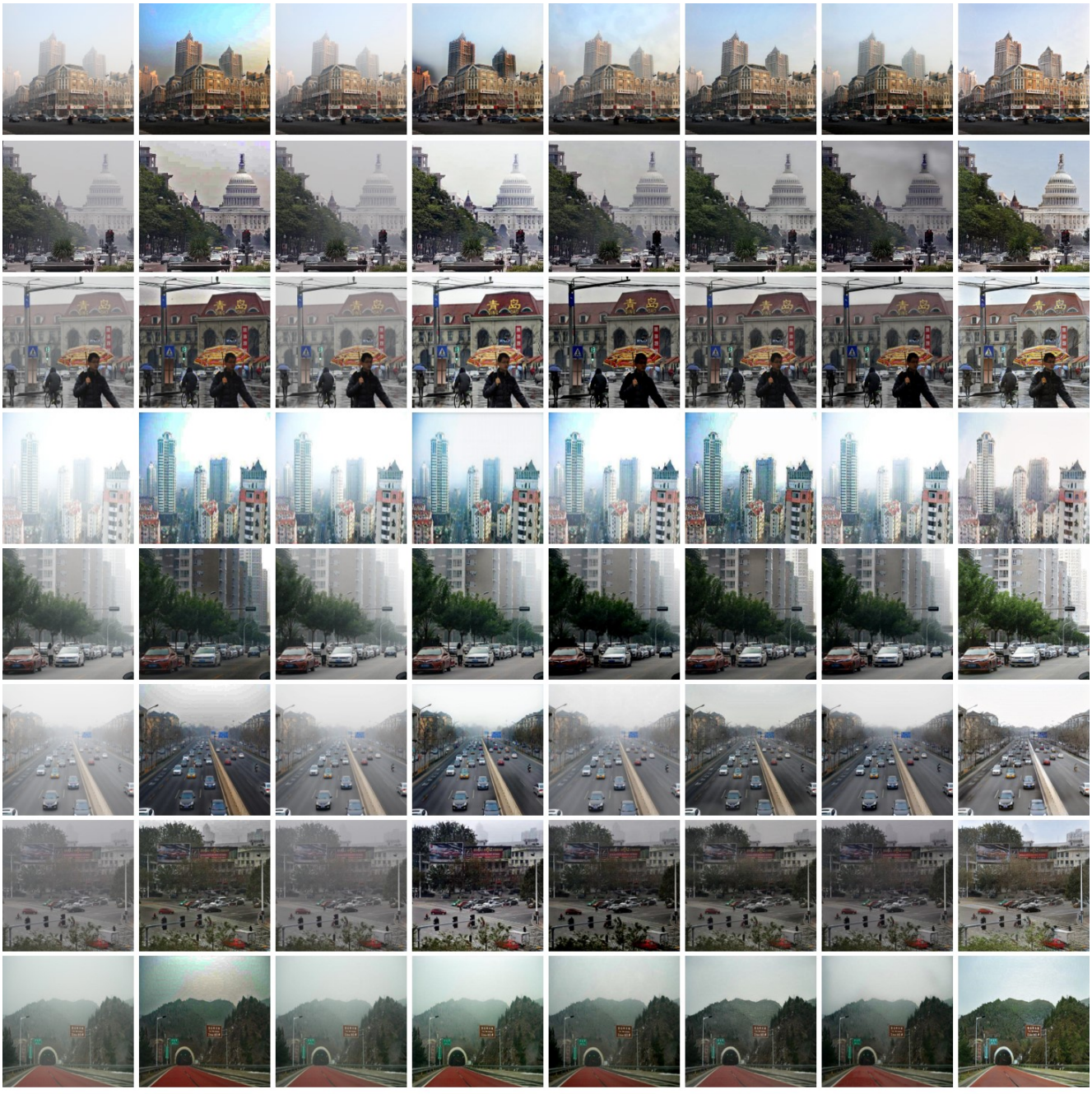

Figure 7. Visual comparison of samples from Haze2020.

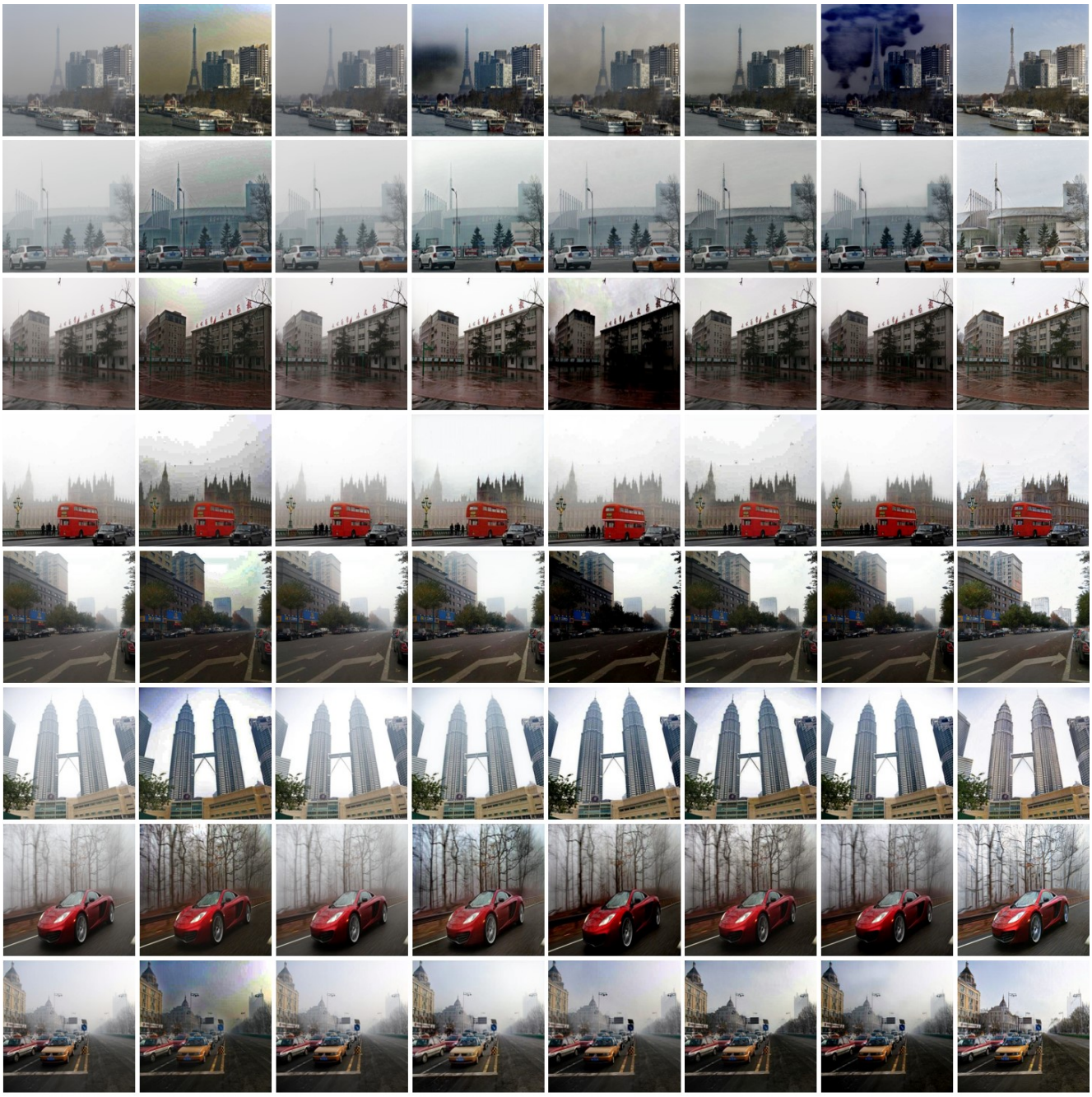

Figure 8. Visual comparison of samples from Haze2020.

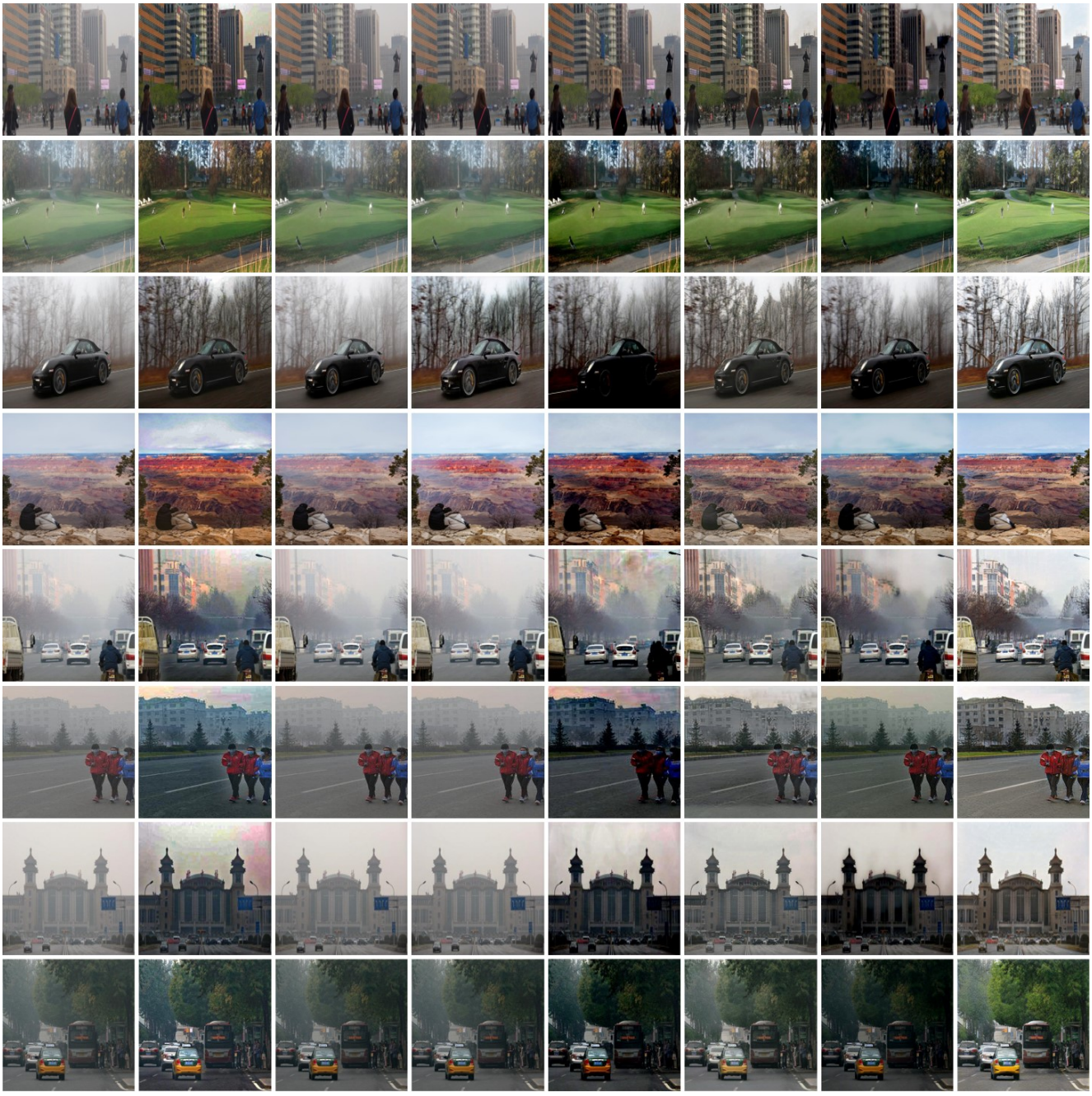

Figure 9. Visual comparison of samples from RTTS.

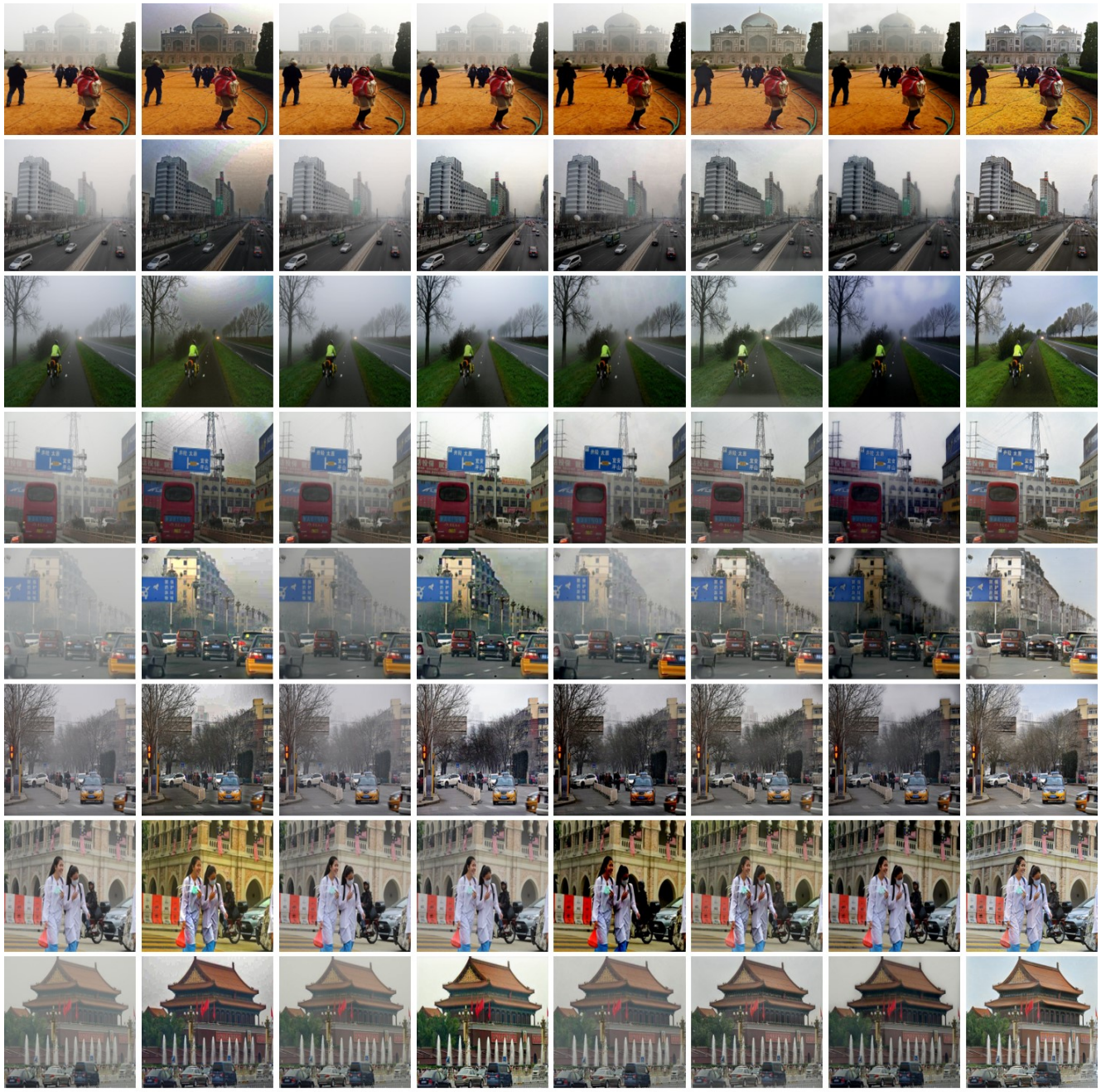

Figure 10. Visual comparison of samples from RTTS.

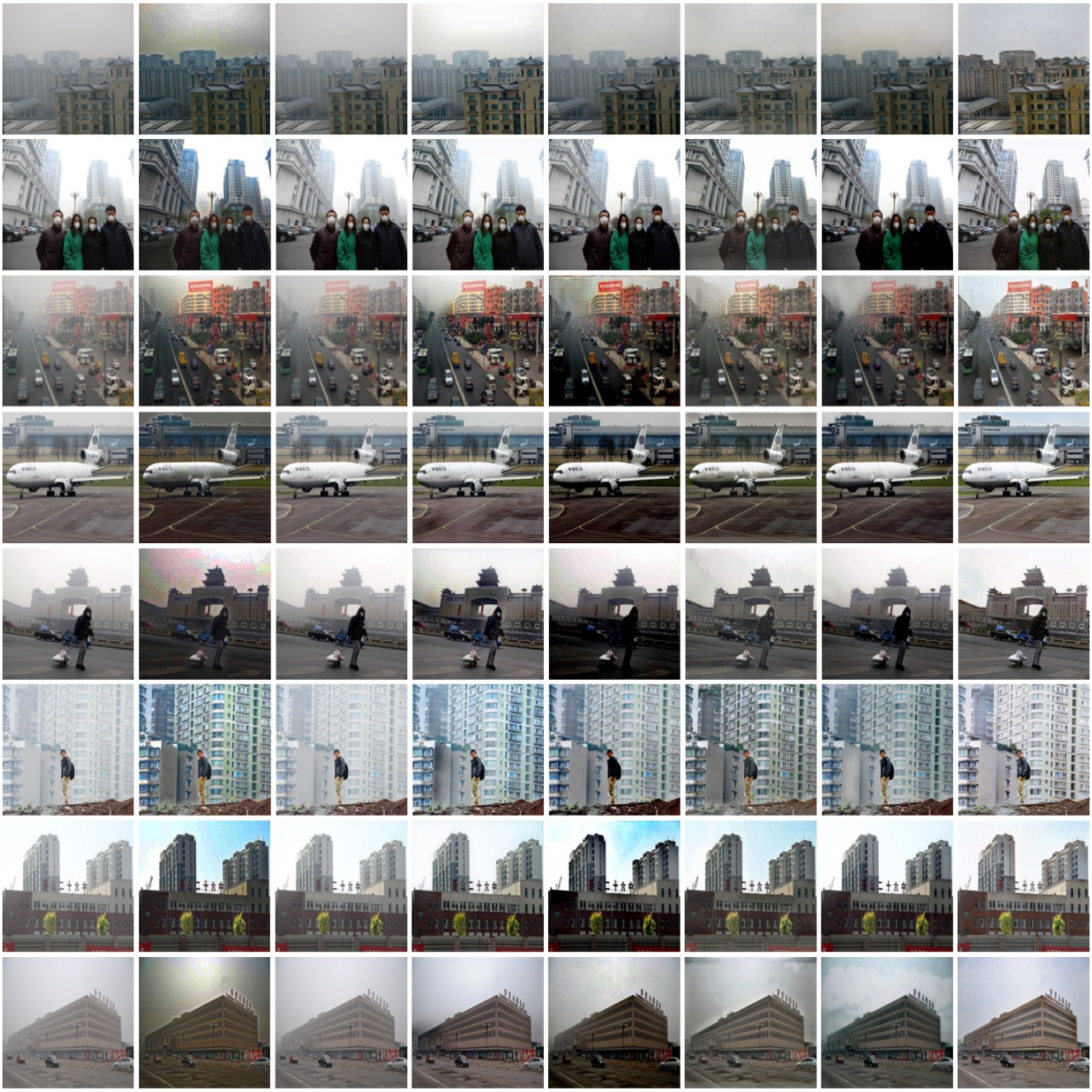

Figure 11. Visual comparison of samples from URHI.

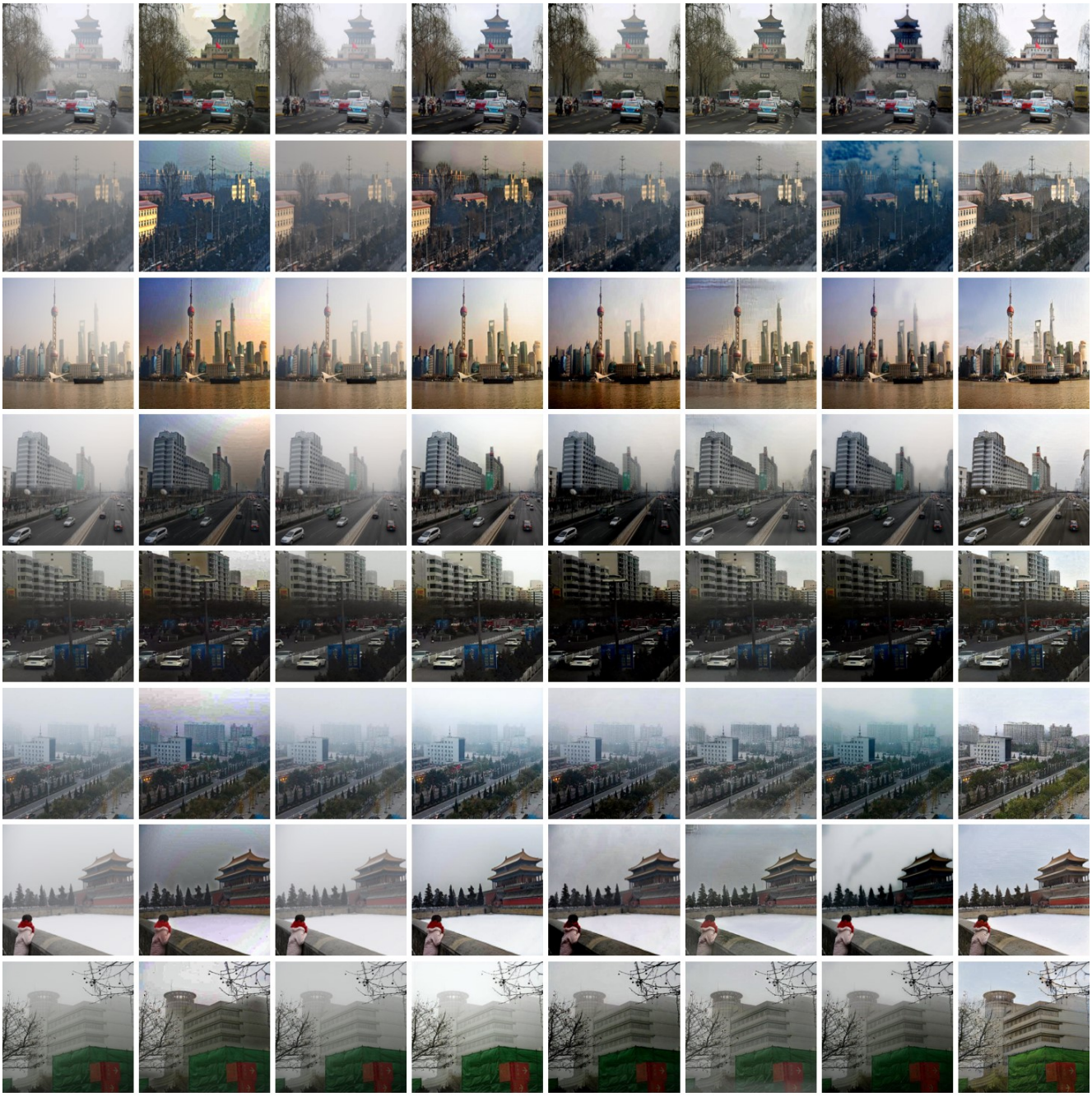

Figure 12. Visual comparison of samples from URHI.