SUPPLEMENTARY MATERIAL OF DIFFUSE AND REFINE LATENT PRIOR WITH TRANSFORMERS FOR NEURAL ISP

7. IMPLEMENTATION DETAILS

As mentioned in § 3.3, the training of D&R includes two stages. For Stage 1, D&R consists of the latent encoder (LE) and the regression-based Transformer. We directly adopt the architecture of LE in [16], and the details are given in the following. The ground-truth latent prior $\mathbf{z} \in \mathbb{R}^{N \times C'}$ is generated by LE with $N = 256$ and $C' = 256$. As for the regression-based Transformer, we apply Restormer, a 4-level encoder-decoder Transformer architecture. From Level 1 to Level 4, we set the number of Transformer blocks as $[3, 5, 6, 6]$, which are denoted as $[L_1, L_2, L_3, L_4]$ in Fig. 4, the number of channels as $[48, 96, 192, 384]$, *i.e.*, $C = 48$, and the attention heads as $[1, 2, 4, 8]$. Besides, there are 4 blocks in the refinement stage, *i.e.*, $L_r = 4$. The channel expansion factor is 2. For Stage 2, D&R consists of the regression-based Transformer, $\mathrm{LE}_{\mathrm{DT}}$, Diffusion Transformer (DT) and Refinement Transformer (RT). The architecture of $\mathrm{LE}_{\mathrm{DT}}$ is the same as that of LE. For DT, there are 4 DT blocks, *i.e.*, $L_{\mathrm{DT}} = 4$, and the iteration number $T = 32$. As for RT, there are 2 RT blocks, *i.e.*, $L_{\mathrm{RT}} = 2$, and the token number of the learned latent bottleneck $\mathbf{h}$ is 128, *i.e.*, $N' = 128$. All these hyperparameters are chosen empirically.

Latent Encoder (LE). As shown in Fig. 1, given the demosaicked RAW image $\mathbf{I}_{\mathrm{Dem}} \in \mathbb{R}^{H \times W \times 3}$ and its corresponding ground-truth counterpart $\mathbf{I}_{\mathrm{GT}} \in \mathbb{R}^{H \times W \times 3}$, we first concatenate them along the channel dimension and feed them into LE to generate the latent prior $\mathbf{z} \in N \times C'$. Here H and W represent the image height and width, while N and C' are the token number and channel dimensions of $\mathbf{z}$. Importantly, the token number N is a constant much smaller than $H \times W$. The compression ratio ($\frac{H \times W}{N}$) is much higher than that applied in previous latent diffusion. Therefore, the computational burden of the subsequent latent diffusion model is effectively reduced. The details of LE are depicted in Fig. 5, which contains L residual blocks. The architecture of $\mathrm{LE}_{\mathrm{DT}}$ is the same as LE except that the input channel of the first convolutional layer is 3 instead of 6, because $\mathrm{LE}_{\mathrm{DT}}$ takes as input only the demosaicked RAW image $\mathbf{I}_{\mathrm{Dem}} \in \mathbb{R}^{H \times W \times 3}$. We set $N = 256$, $C' = 256$, and $L = 5$.

Hierarchical Integration Module (HIM). To effectively integrate the latent prior and intermediate feature of the regression-based Transformer, we adopt HIM. As illustrated in Fig. 4, the HIM is placed in front of each encoder and decoder. For each HIM, cross-attention is computed between the latent prior and intermediate features for feature fusion. This module allows the information in the latent prior to be aggregated into features of the regression-based Transformer. Specifically, as shown in Fig. 6, given the intermediate feature $\mathbf{X}_{in} \in \mathbb{R}^{\hat{H} \times \hat{W} \times \hat{C}}$, we reshaped it as tokens $\mathbf{X}_r \in \mathbb{R}^{\hat{H}\hat{W} \times \hat{C}}$; where $\hat{H} \times \hat{W}$ is spatial resolution and $\hat{C}$ denotes channel dimension. Then we linearly project $\mathbf{X}_r$ into $\mathbf{Q} \in \mathbb{R}^{\hat{H}\hat{W} \times \hat{C}}$ (*query*). Similarly, we project the latent prior $\bar{\mathbf{z}}_i \in \mathbb{R}^{\hat{N} \times C'}$ as $\mathbf{K} \in \mathbb{R}^{\hat{N} \times \hat{C}}$ (*key*) and $\mathbf{V} \in \mathbb{R}^{\hat{N} \times \hat{C}}$ (*value*). The cross-attention is formulated as:

$$\begin{aligned} \mathbf{Q} = \mathbf{W}_Q \mathbf{X}_r, \mathbf{K} = \mathbf{W}_K \bar{\mathbf{z}}_i, \mathbf{V} = \mathbf{W}_V \bar{\mathbf{z}}_i, \\ \mathrm{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{SoftMax}(\mathbf{Q}\mathbf{K}^T / \sqrt{\hat{C}}) \cdot \mathbf{V}, \end{aligned} \tag{3}$$

where $\mathbf{W}_Q \in \mathbb{R}^{\hat{C} \times \hat{C}}$, $\mathbf{W}_K \in \mathbb{R}^{C' \times \hat{C}}$, and $\mathbf{W}_V \in \mathbb{R}^{C' \times \hat{C}}$ represent learnable parameters of linear projections without bias. As vanilla multi-head self-attention, we separate channels into multiple "heads" and calculate the attention operations. Note that Fig. 6 depicts the situation with a single head and omits some details for simplification. Finally, we reshape and project the output of cross-attention, and add it with $\mathbf{X}_{in}$ to derive the output feature $\mathbf{X}_{out} \in \mathbb{R}^{\hat{H} \times \hat{W} \times \hat{C}}$. For $\bar{\mathbf{z}}_1$, $\hat{N} = 256$; for $\bar{\mathbf{z}}_2$, $\hat{N} = 64$; for $\bar{\mathbf{z}}_3$, $\hat{N} = 1$.

Global Color Mapping (GCM) module. For optical flow estimation, GCM's output $\mathbf{I}_{\mathrm{GCM}}$ is required to satisfy two prerequisites: *(i)* $\mathbf{I}_{\mathrm{GCM}}$ should imitate the color of the ground-truth image $\mathbf{I}_{\mathrm{GT}}$ for diminishing the severe color inconsistency; *(ii)* the spatial position of the pixels should keep the same as the input demosaicked RAW image $\mathbf{I}_{\mathrm{Dem}}$. Thus, as shown in Fig. 7, GCM adopts a Spatially Preserving Network (SPN) as its backbone, which is composed of a stack of 1×1 convolutional layers, and further employs a GuideNet to extract the color information from $\mathbf{I}_{\mathrm{GT}}$ as the condition for the backbone. Besides, for the purpose of anti-vignetting, GCM takes a 2D coordinate map $\mathbf{I}_{\mathrm{Coord}}$ as another input. With $\mathbf{I}_{\mathrm{GCM}}$, we use PWC-Net [25] to estimate the optical flow for warping $\mathbf{I}_{\mathrm{GT}}$. The warped sRGB image $\mathbf{I}_{\mathrm{GT}}^{\mathrm{w}}$ can then be adopted as the supervision for training GCM, the regression-based Transformer and LE. In Eq. 1 and Eq. 2 of the submission, we introduce $\mathbf{m}$ as a mask indicating valid positions of the optical flow. Here each element m_i of $\mathbf{m}$ is defined as

$$m_i = \begin{cases} 1, & [\mathcal{W}(\mathbf{1}, \Psi)]_i \geq 1 - \epsilon \\ 0, & \text{otherwise} \end{cases}, \tag{4}$$

where $\mathcal{W}$ is a warping function, $\mathbf{1}$ denotes an all-1 matrix, Ψ is the estimated optical flow, ϵ is a threshold set to 0.001, and $[\cdot]_i$ denotes the i-th element of a matrix.

Diffusion Transformer (DT). The hidden dimension of DT is 256. DT has 4 DT blocks. Within each DT block, the number Multi-Head Self Attention (MHSA) heads is 4, and the mlp ratio of the Pointwise FeedForward (PFF) is 4.0.

Refinement Transformer (RT). RT is made up of 2 RT blocks, a final cross-attention module, and a learned latent bottleneck. Within each RT block, the number of cross-attention heads is 1, and the hidden dimension per head is 128; the number of MHSA heads is 4, and the hidden dimension per head is 64; the hidden dimension for two PFFs is 256. For the final

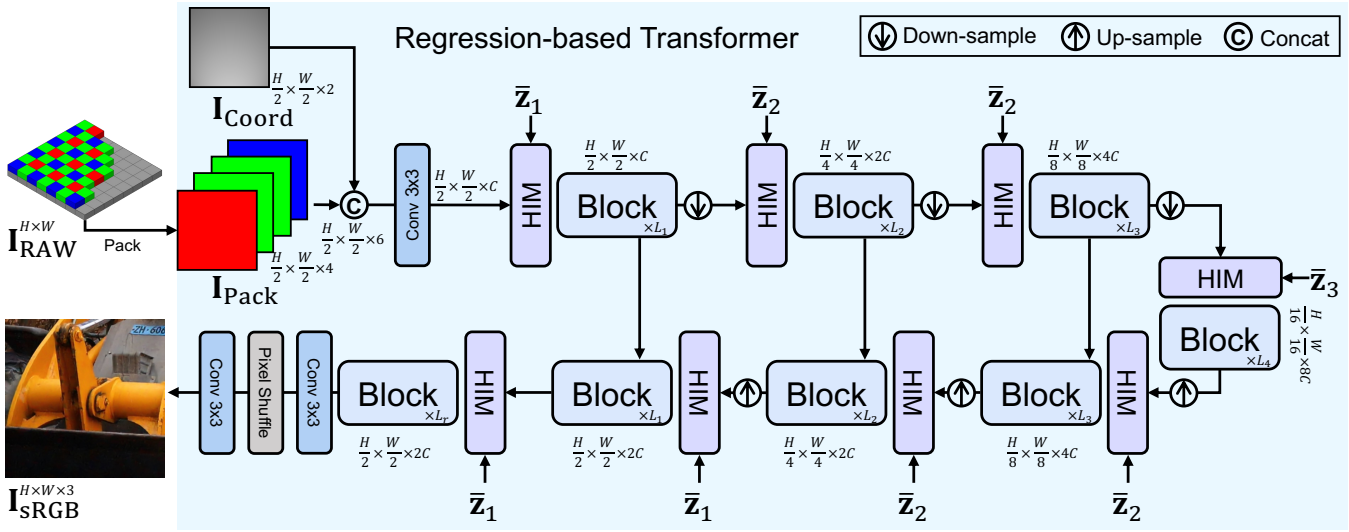

Fig. 4. The regression-based Transformer adopts hierarchical encoder-decoder architecture equipped with HIM (abbr. Hierarchical Integration Module) and takes as input the concatenation of the packed RAW image $\mathbf{I}_{\text{Pack}}$ and a coordinate map $\mathbf{I}_{\text{Coord}}$ (for handling anti-vignetting [14]). HIM integrates the multi-scale latent priors by calculating the cross-attention between the prior and the intermediate feature of the regression-based Transformer.

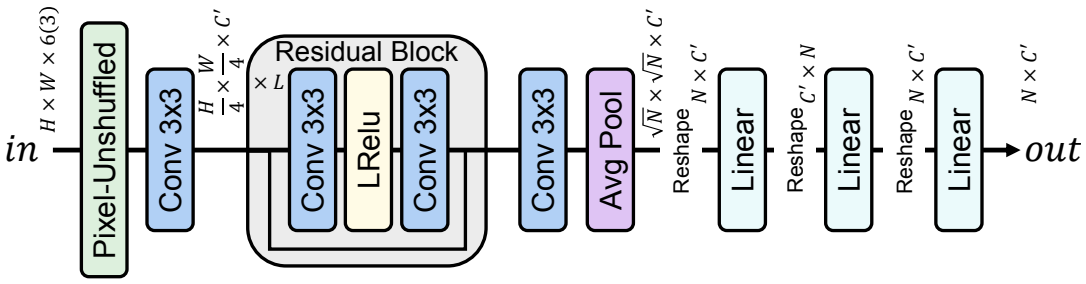

Fig. 5. The architecture of LE and LE_{DM}.

cross-attention module, the head number is 1, and the hidden dimension per head is 128. For the learned latent bottleneck, its token number is 128.

D&R (Tiny). From Level 1 to Level 4, we set the number of Transformer blocks as $[3, 3, 4, 4]$ and the number of channels as $[40, 80, 160, 320]$. Besides, there are 3 blocks in the refinement stage. We set the dimensions of the latent prior as $N = 16$ and $C' = 256$. Accordingly, we set the token number of the learned latent bottleneck in RT as $N' = 8$. The iteration number of DT is set as $T = 12$. All the other hypeparameters are kept the same as those of D&R.

8. DIFFUSION MODEL (DM)

In Stage 2, DT based DM is trained to predict the latent prior $\hat{\mathbf{z}}$ which mimics the ground-truth latent prior $\mathbf{z}$ generated in Stage 1. Our DM is based on conditional denoising diffusion probabilistic model [26]. DM involves a forward diffusion process and a reverse denoising process.

In the forward diffusion process, given a ground-truth image, we first adopt the latent encoder (LE) trained in Stage 1 to generate the ground-truth latent prior $\mathbf{z}$. Let $\mathbf{z}_0 = \mathbf{z}$. We take $\mathbf{z}_0$ as the starting point of the forward Markov process, and gradually add Gaussian noise to it over T iterations as follows:

$$\begin{aligned} q(\mathbf{z}_{1:T} \mid \mathbf{z}_0) &= \prod_{t=1}^{T} q(\mathbf{z}_t \mid \mathbf{z}_{t-1}), \\ q(\mathbf{z}_t \mid \mathbf{z}_{t-1}) &= \mathcal{N}(\mathbf{z}_t; \sqrt{1-\beta_t}\mathbf{z}_{t-1}, \beta_t \mathbf{I}), \end{aligned} \tag{5}$$

where $t = 1, \ldots, T$; $\mathbf{z}_t$ represents the noisy latent prior at the t-th step; $\beta_{1:T} \in (0, 1)$ are hyperparameters that control the variance of the noise; $\mathcal{N}$ denotes the Gaussian distribution. Through iterative derivation with reparameterization, Eq. (5) can be written as:

$$\begin{aligned} q(\mathbf{z}_t \mid \mathbf{z}_0) &= \mathcal{N}\big(\mathbf{z}_t; \sqrt{\bar{\alpha}_t}\mathbf{z}_0, (1-\bar{\alpha}_t)\mathbf{I}\big), \\ \alpha_t &= 1 - \beta_t, \quad \bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i. \end{aligned} \tag{6}$$

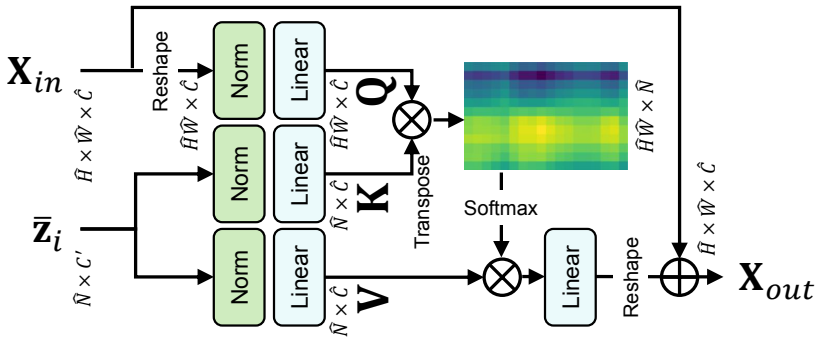

Fig. 6. The architecture of HIM.

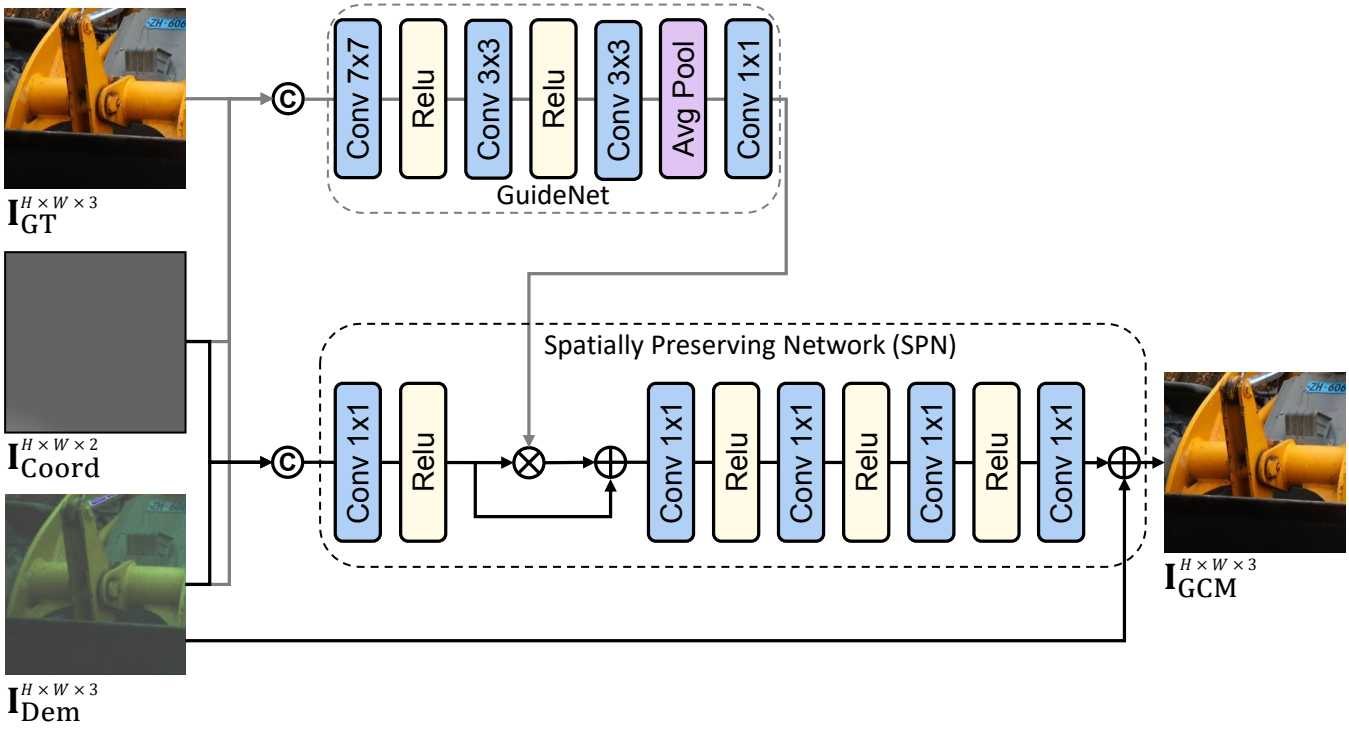

Fig. 7. The architecture of GCM.

In the reverse process, we aim to predict the latent prior from a pure Gaussian distribution. The reverse process is a T-step Markov chain that runs backwards from $\mathbf{z}_T$ to $\mathbf{z}_0$. Specifically, for the reverse step from $\mathbf{z}_t$ to $\mathbf{z}_{t-1}$, we use the posterior distribution as:

$$q(\mathbf{z}_{t-1} \mid \mathbf{z}_t, \mathbf{z}_0) = \mathcal{N}\big(\mathbf{z}_{t-1}; \boldsymbol{\mu}_t(\mathbf{z}_t, \mathbf{z}_0), \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t\mathbf{I}\big), \\ \boldsymbol{\mu}_t(\mathbf{z}_t, \mathbf{z}_0) = \frac{1}{\sqrt{\alpha_t}}(\mathbf{z}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}), \tag{7}$$

where $\boldsymbol{\epsilon}$ represents the noise in $\mathbf{z}_t$, and is the only uncertain variable. Following [26], we adopt a denoising network $\boldsymbol{\epsilon}_\theta$ to estimate the noise $\boldsymbol{\epsilon}$ for each step. Since DM operates in the latent space, we utilize another latent encoder ($\mathrm{LE}_{\mathrm{DT}}$), with the same architecture as LE. $\mathrm{LE}_{\mathrm{DT}}$ compresses the demosaicked RAW image into latent space to get the conditional latent feature $\mathbf{c}$. The denoising network predicts the noise conditioned on $\mathbf{z}_t$ and $\mathbf{c}$, *i.e.*, $\boldsymbol{\epsilon}_\theta(\mathbf{z}_t, \mathbf{c}, t)$. With the substitution of $\boldsymbol{\epsilon}_\theta$ in Eq. (7) and setting the variance to $(1-\alpha_t)$, we get

$$\mathbf{z}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\big(\mathbf{z}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{z}_t, \mathbf{c}, t) + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_t\big), \tag{8}$$

where $\boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{I})$. By iteratively sampling $\mathbf{z}_t$ using Eq. (8) T times, we can predict the latent prior $\hat{\mathbf{z}}$ as shown in Fig. 1 (a). $\hat{\mathbf{z}}$ is then used to guide the regression-based Transformer, *i.e.*, $\bar{\mathbf{z}}_1 = \hat{\mathbf{z}}$. Notably, since the distribution of the latent space ($\mathbb{R}^{N\times C'}$) is much simpler than that of images ($\mathbb{R}^{H\times W\times C}$), the latent prior can be generated with a small number of iterations.

Training DM means training denoising network $\boldsymbol{\epsilon}_\theta$. Previous works train DM by optimizing the weighted variational bound with the following training objective:

$$\nabla_\theta \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{z} + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, \mathbf{c}, t)\|_2^2, \tag{9}$$

where $\mathbf{z}$ and $\mathbf{c}$ are the ground-truth latent prior and conditional latent feature defined above; $t \in [1, T]$ is a random time-step; $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$ denotes the sampled noise. However, the objective in Eq. (9) only trains DM. Since the slight deviation between the predicted latent prior $\hat{\mathbf{z}}$ and the ground-truth latent prior $\mathbf{z}$, directly combining DM with the regression-based Transformer could cause a mismatch, which restricts the RAW-to-sRGB mapping performance.

To overcome this issue, we jointly train DM and the regression-based Transformer. Specifically, for each training iteration, we use the ground-truth latent prior $\mathbf{z}$ to generate the noisy sample $\mathbf{z}_T$ through Eq. (5). As the time-step T is small in latent space, we then run the complete T iteration reverse processes (Eq. (8)) to predict the latent prior $\hat{\mathbf{z}}$ which is used to guide the regression-based Transformer through HIM ($\bar{\mathbf{z}}_1 = \hat{\mathbf{z}}$).

9. TRAINING DETAILS

We train D&R with AdamW optimizer with β_1=0.9 and β_2=0.99. For Stage 1, the total training iterations are 300K. The initial learning rate is set as 2×10^{-4} and stays the same for the first 100K iterations. Then it gradually reduced to 1×10^{-6} with the cosine annealing. For Stage 2, we adopt the same training settings as in Stage 1. For both stages, the batch size is set as 8. Moreover, we apply random rotation and flips for data augmentation. We use PyTorch to implement our models. The training of each stage of D&R lasts 4 days on an A100 GPU.

Table 3. Quantitative results on SID dataset. "-" means that the corresponding papers do not report a specific metric value. "*" marks methods that require the clean RAW image as the supervision signal in addition to the target sRGB image. (red: best, blue: 2nd best)

Methods	Params	Sony			Fuji		
		PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
SID [27]	7.7 M	28.96	0.787	0.356	26.66	0.709	0.432
DID [Supp2]	2.5 M	29.16	0.785	0.368	-	-	-
SGN [Supp3]	19.2 M	29.28	0.790	0.370	27.41	0.720	0.430
LLPackNet [Supp4]	1.2 M	27.83	0.755	0.541	-	-	-
RRT [Supp5]	0.8 M	28.66	0.790	0.397	26.94	0.712	0.446
EEMEFN [Supp6]	40.7 M	29.60	0.795	0.350	27.38	0.723	0.414
LDC [Supp7]	8.6 M	29.56	0.799	0.359	27.18	0.703	0.446
MCR [Supp8]	15.0 M	29.65	0.797	0.348	-	-	-
RRENet [Supp9]	15.5 M	29.17	0.792	0.360	27.29	0.720	0.421
DNF* [Supp10]	2.8 M	30.62	0.797	0.343	28.71	0.726	0.391
RAWMamba* [Supp11]	6.2 M	30.76	0.810	0.328	29.02	0.743	0.382
D&R	31.6 M	31.10	0.814	0.261	29.07	0.743	0.346
D&R (Tiny)	17.1 M	30.93	0.811	0.260	29.13	0.744	0.348

10. EXPERIMENTS ON SID DATASET

SID dataset contains 5094 extremely low-light RAW images with corresponding normal-light reference sRGB images taken by two cameras: Sony A7S2 with Bayer sensor and a resolution of 2832 × 4240, and Fuji X-T2 with X-Trans sensor and a resolution of 4000 × 6000. The exposure time of the low-light image varies from 0.1s to 0.033s, and the reference images are captured 100 to 300 times longer than the exposure time of the low-light images. Table 3 shows the quantitative comparisons between D&R[1] and the existing methods. Note that although D&R does not use the clean RAW image as the additional supervision signal, our D&R and D&R (Tiny) still outperforms the previous state-of-the-art methods RAW-Mamba [Supp11] and DNF [Supp10] by a noticeable margin, especially on the LPIPS metric which demonstrates the good perceptual quality. For D&R and D&R (Tiny), we directly adopt the hyperparameters selected for the ZRR dataset, so further parameter sweeping may improve the performance of D&R on the SID dataset.

11. VISUAL RESULTS

We provide visual comparisons with the existing methods in Fig. 8, Fig. 9, Fig. 10 and Fig. 11. We provide visual comparisons among our methods and the ablative versions in

[1]Since the misalignment issue is not obvious on SID, we did not use the GCM loss (2) and the mask in (1) when training D&R on SID.

Fig. 12, Fig. 13 and Fig. 14. It can be observed that in Fig. 12, D&R is on par with D&R (GAN); in Fig. 13, D&R yields more image details than D&R (GAN); in Fig. 14, D&R (GAN) generates sharper textures than D&R. D&R and D&R (GAN) consistently outperform the ablative versions.

12. REFERENCES

[1] Chen Chen, Qifeng Chen, Jia Xu, and Vladlen Koltun, "Learning to see in the dark," in *CVPR*, 2018.

[2] Paras Maharjan, Li Li, Zhu Li, Ning Xu, Chongyang Ma, and Yue Li, "Improving extreme low-light image denoising via residual learning," in *ICME*, 2019.

[3] Shuhang Gu, Yawei Li, Luc Van Gool, and Radu Timofte, "Self-guided network for fast image denoising," in *ICCV*, 2019.

[4] Mohit Lamba, Atul Balaji, and Kaushik Mitra, "Towards fast and light-weight restoration of dark images," in *BMVC*, 2020.

[5] Mohit Lamba and Kaushik Mitra, "Restoring extremely dark images in real time," in *CVPR*, 2021.

[6] Minfeng Zhu, Pingbo Pan, Wei Chen, and Yi Yang, "EEMEFN: low-light image enhancement via edge-enhanced multi-exposure fusion network," in *AAAI*, 2020.

[7] Ke Xu, Xin Yang, Baocai Yin, and Rynson W. H. Lau, "Learning to restore low-light images via decomposition-and-enhancement," in *CVPR*, 2020.

[8] Xingbo Dong, Wanyan Xu, Zhihui Miao, Lan Ma, Chao Zhang, Jiewen Yang, Zhe Jin, Andrew Beng Jin Teoh, and Jiajun Shen, "Abandoning the bayer-filter to see in the dark," in *CVPR*, 2022.

[9] Haofeng Huang, Wenhan Yang, Yueyu Hu, Jiaying Liu, and Ling-Yu Duan, "Towards low light enhancement with RAW images," *TIP*, 2022.

[10] Xin Jin, Linghao Han, Zhen Li, Chun-Le Guo, Zhi Chai, and Chongyi Li, "DNF: decouple and feedback network for seeing in the dark," in *CVPR*, 2023.

[11] Xianmin Chen, Peiliang Huang, Xiaoxu Feng, Dingwen Zhang, Longfei Han, and Junwei Han, "Retinex-rawmamba: Bridging demosaicing and denoising for low-light RAW image enhancement," *Arxiv*, 2024.

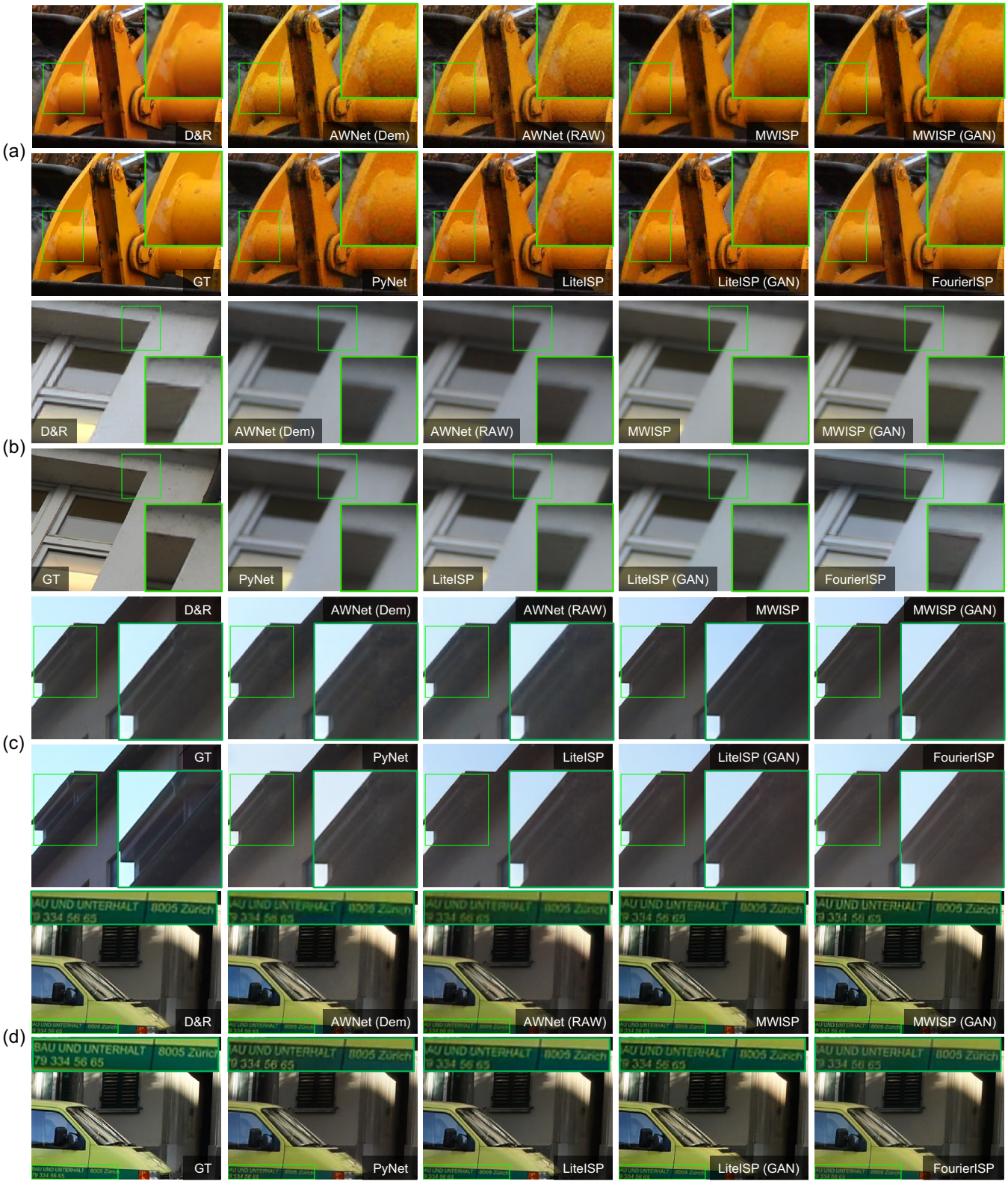

Fig. 8. Visual comparisons on ZRR dataset. Please zoom in for better observation.

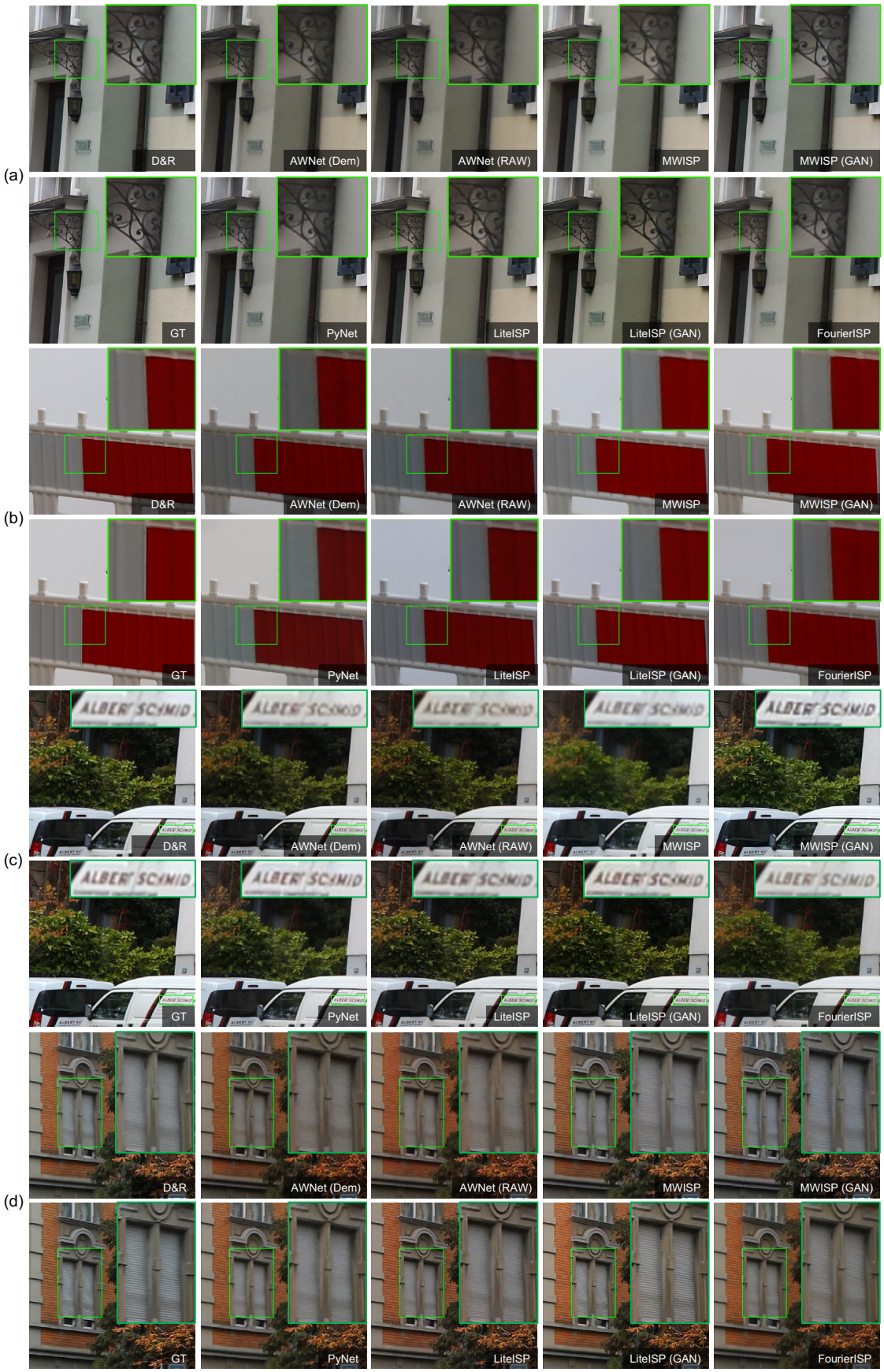

Fig. 9. Visual comparisons on ZRR dataset. Please zoom in for better observation.

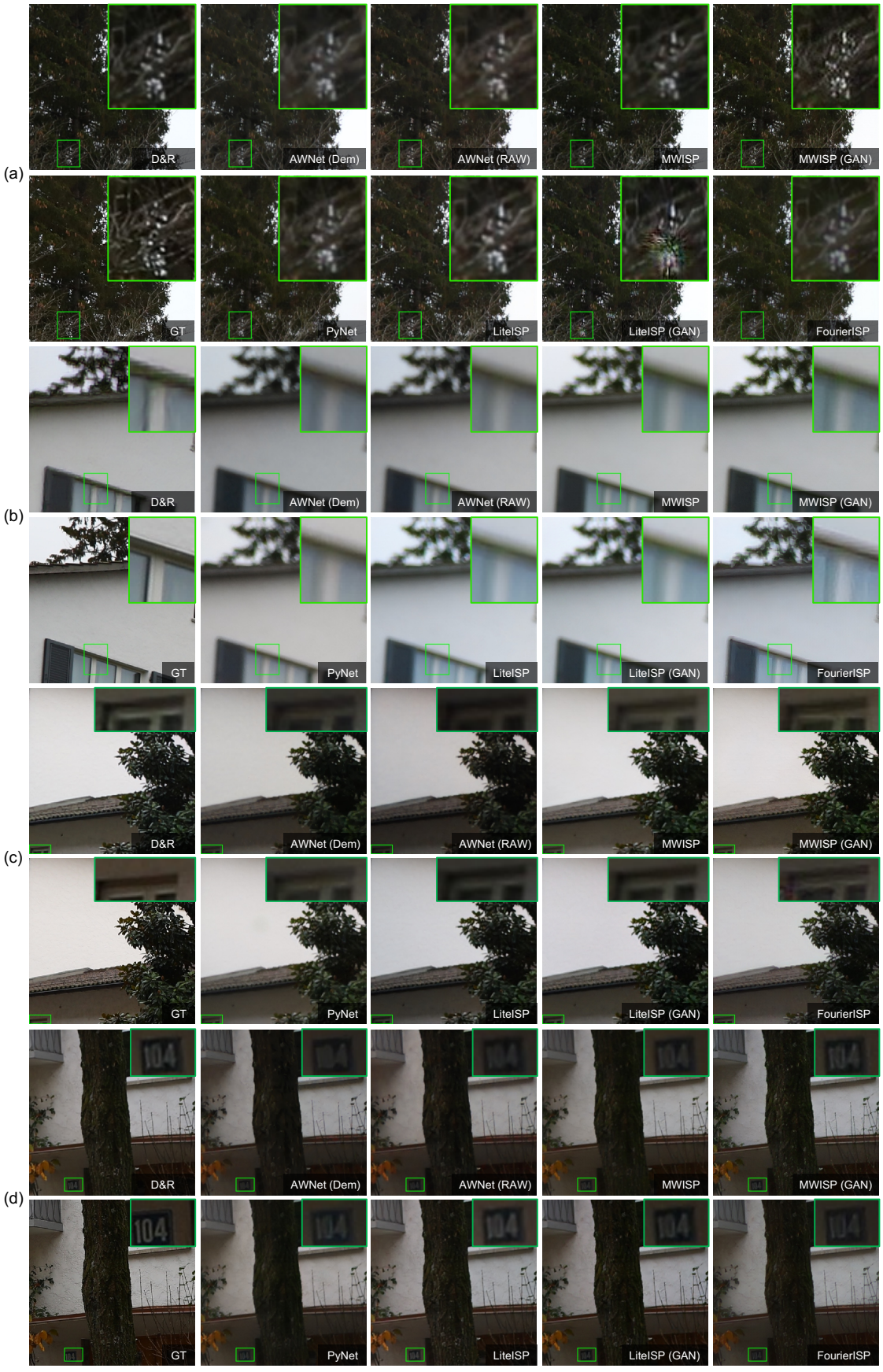

Fig. 10. Visual comparisons on ZRR dataset. Please zoom in for better observation.

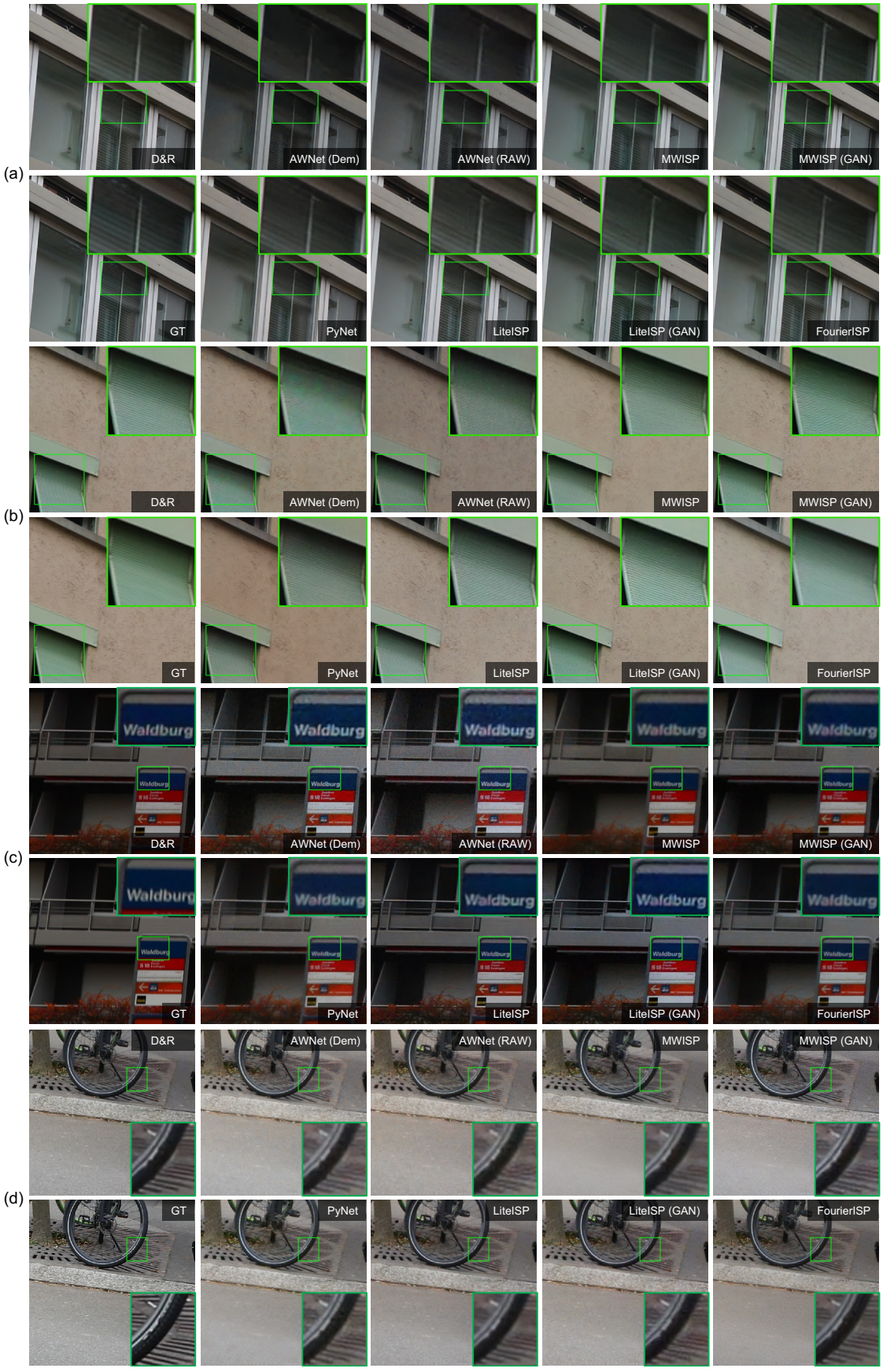

Fig. 11. Visual comparisons on ZRR dataset. Please zoom in for better observation.

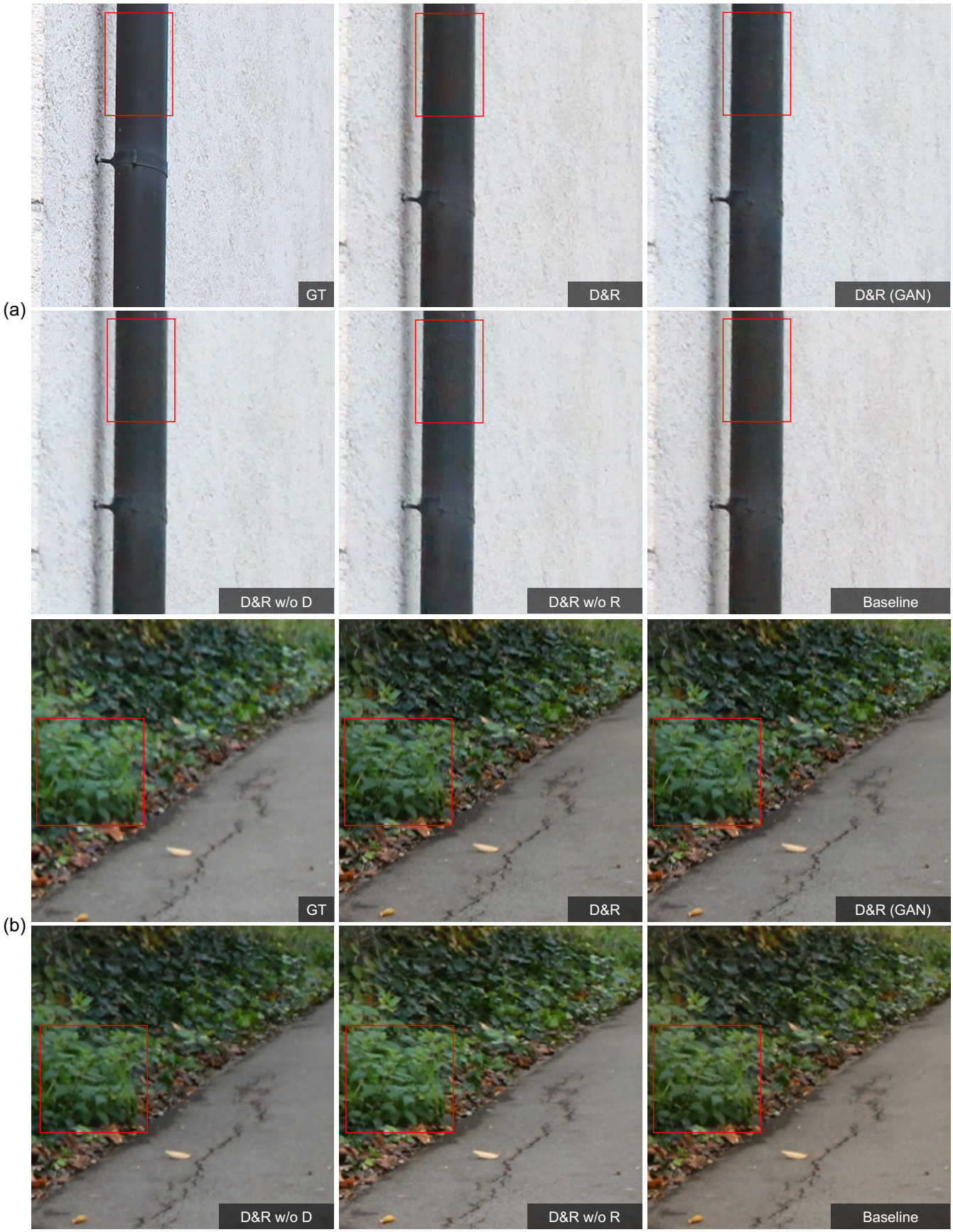

Fig. 12. Ablative visual comparisons. Please zoom in for better observation.

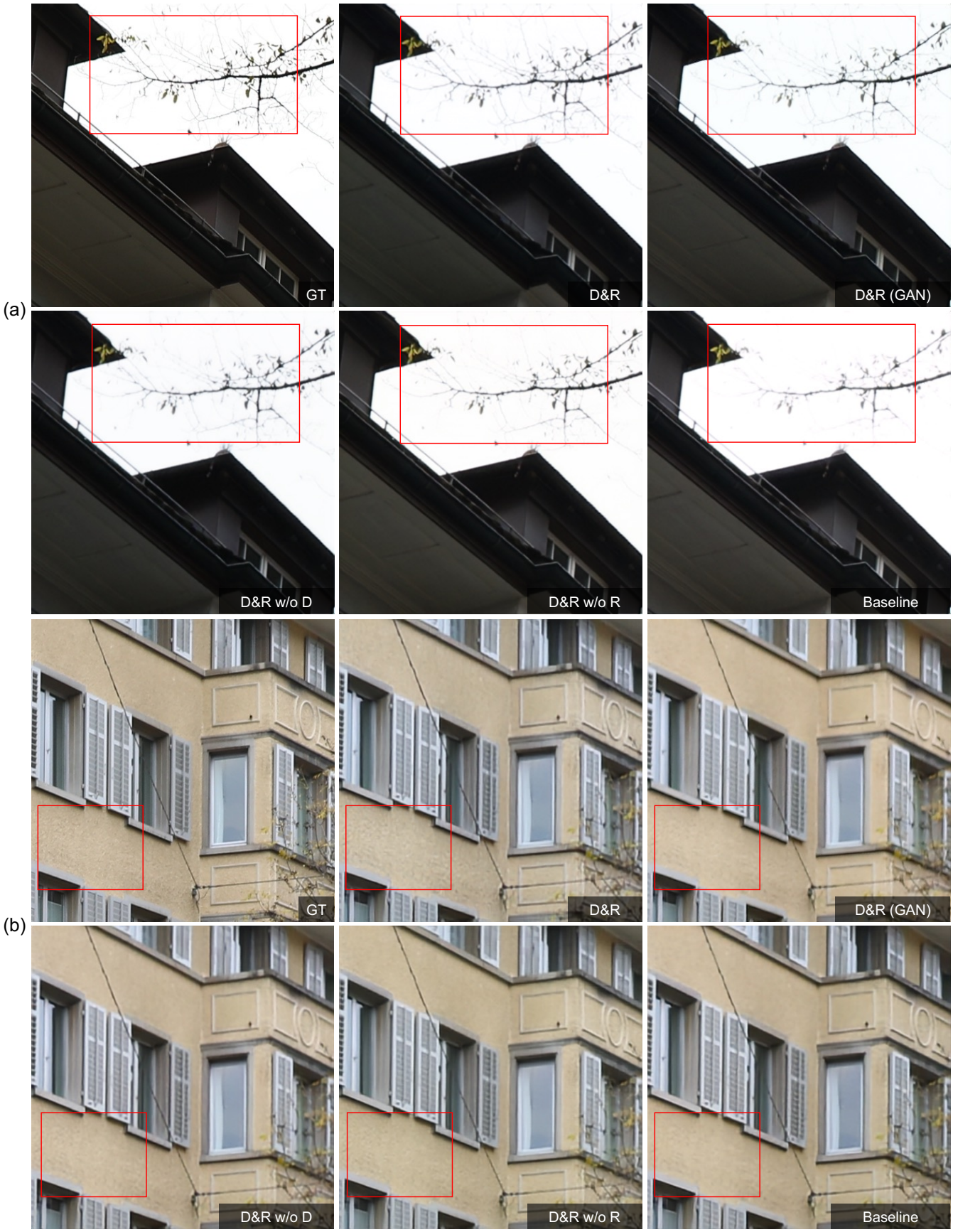

Fig. 13. Ablative visual comparisons. Please zoom in for better observation.

Fig. 14. Ablative visual comparisons. Please zoom in for better observation.