Neural Camera Simulators
Supplementary Material

In this supplementary material, we provide additional analysis, implementation details, and experimental results. First, we analyze the effectiveness of each module design and show the quantitative results on raw image simulation. Then, we provide more implementation details of different modules, baselines, and applications.

1. Model Analysis

1.1. Regression model in the exposure module

We analyze the design of the linear regression model in the exposure module. We adopt different b values for each channel in the RGBG Bayer pattern. When training this module, we exclude image pairs with aperture value changes. The final optimized w is 1.0038 and the optimize b is $[-0.0003, -0.0004, -0.0004, -0.0005]$. As we expect, w is close to 1 and b close to 0. We quantitatively analyze the exposure module by calculating the L_2 loss on the test dataset. The mean squared error for simply multiplying the multiplier is 0.7729, and that for applying the exposure module is 0.0033.

1.2. Exposure stops

Unlike many other parameters, the controllable exposure setting in a camera is not continuous but discrete with certain stops. One stop means doubling or halving the amount of light arriving at the sensor. Given a specific scene, the number of combinations of camera parameters at the suitable exposure stop is tremendous (e.g., ISO 100, Exposure time 1s, Aperture 1/5.6 and ISO 200, Exposure time 1/4 s, Aperture 1/4.0 are at the same stop). These combinations enable the possibility of capturing photos at the same exposure stops with different artistic styles. Since modern cameras (including all cameras used for data collecting in this paper) usually adopt one-third fractional f-number stop. Specifically, the difference between each step is $\frac{1}{3}Ev$, where $1Ev$ corresponds to a standard power-of-two exposure step, and one f-stop refers to a factor of $\sqrt{2}$ change in f-number. Thus the f-number steps in this series can be $\sqrt{2}^{s/3}$ where $s \in \mathbb{N}$. We can calculate the corresponding step index of input and output settings to get the more accurate multiplier:

$$s_1 = Round(6\log_2(n_1)), \tag{1}$$

$$s_2 = Round(6\log_2(n_2)), \tag{2}$$

$$\alpha_n = 2^{(s_1 - s_2)/3}. \tag{3}$$

1.3. Attention layers in the aperture module

We trained another baseline U-Net [10] without the attention layer to show the effectiveness of the attention module. The input and output aperture is concatenated as additional channels to the input raw data. As in Fig. 1, without the attention module, the model can not learn how to enhance the subtle blur region.

1.4. Order of modules

In this paper, the order of the three modules is exposure correction, noise adjustment, and aperture enhancement. The first step must be the exposure correction since our experiments show that without this step, the network will focus on learning the change in illuminance while not learning the image details. However, the order of the other two modules can be changed to aperture enhancement first and then noise adjustment. We report the new metrics in Tab. 1.

1.5. Image quality evaluation

We provide quantitative results of different simulation direction in Tab. 2. In each direction, only one camera parameter is changed. PSNR/SSIM is not a suitable metric to represent the quality of simulation direction ISO↑, given the case that the simulated image and the ground-truth image are both noisy images with different random noise. Although the metric of ISO↑ is low, the perceptual quality of ISO↑ is relatively high.

1.6. Deblur

We try to synthesize images from large apertures to small apertures. The result is shown in Fig. 2. Our model can not synthesize the deblurring image correctly.

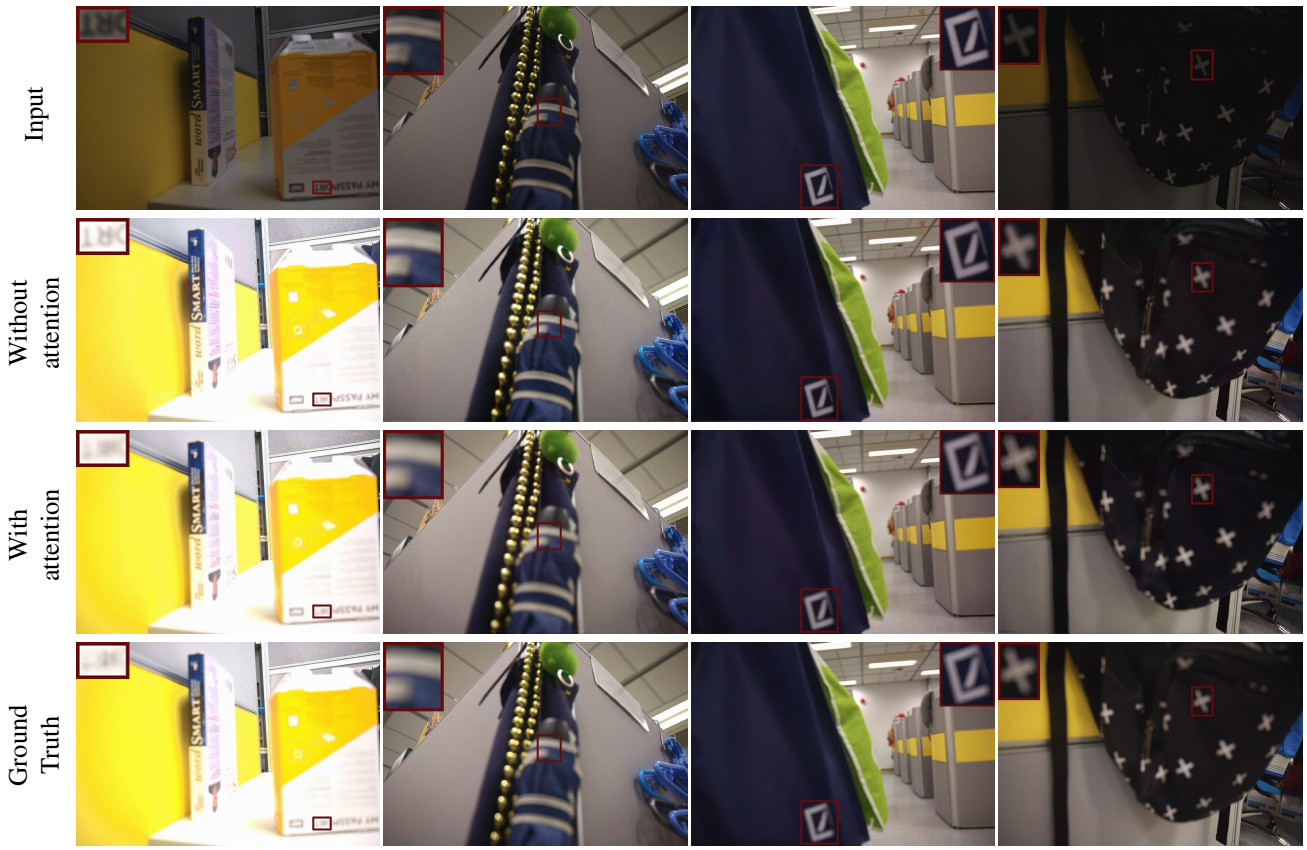


Figure 1. Results of U-Net without and with the attention layer. Images generated without the attention layer cannot magnify the blur region correctly. Our method achieves more similar results to the ground truth image. Best viewed with zoom-in.

	EXP	AP	FULL
PSNR	28.61	28.65	28.90
SSIM	0.861	0.868	0.879

Table 1. PSNR and SSIM of different stages of the model with the order of EXP (exposure module), AP (aperture module), and FULL (full model).

Input Ours GT

Figure 2. Results from large aperture to small aperture.

1.7. HDR baseline

We tried another baseline for HDR experiments: simply multiplying the raw pixels to simulate images with different exposures for HDR generation. The result is shown in Fig. 3, which demonstrates that simple multiplication will introduce more noise.

Input Multiplication Ours

Figure 3. HDR baseline. Zoom in for details.

1.8. NLF noise model

In this section, we discuss the reason that we choose the NLF noise model. The most manageable and accessible noise model is the additive white Gaussian noise model, which adopts a homoscedastic Gaussian assumption. The model assumes that the noise is independent of the image value. The mean of the Gaussian distribution is 0, and the value of the standard deviation decides the noise level. Despite the extensive usage of the additive white

	ISO↑	Exposure time↑	ISO↓	Exposure Time↓	Aperture↑
PSNR	35.47	36.13	37.15	37.64	34.20
SSIM	0.908	0.921	0.939	0.930	0.875

Table 2. PSNR and SSIM of different simulation direction. ↑ means from small to large, and ↓ means from large to small.

Gaussian noise model, it contradicts with the physical imaging principle that the photon noise introduced by photon-counting is dependent on the number of photons. The real noise is mainly composed of two primary sources: signal-independent noise because of the reading and writing of the electronic circuits and signal-dependent noise due to the photon counting. To form a more accurate estimation of the image noise, researchers assume the noise distribution as a mixture of Poisson distribution (for signal-dependent source) and Gaussian distribution (for signal-independent source) [3, 4, 8]. An alternative way [7, 9, 6] to describe this mixture distribution is to regard the Poisson distribution as the part of Gaussian distribution with variance depending on the signal. The heteroscedastic Gaussian model (or noise-level function) has shown the capability to accurately describe the signal-dependent noise. Thus most modern cameras provide a set of calibrated NLF noise parameters to improve the denoising results. Most recent works consider the other minor sources in real noise, such as fixed-pattern noise, defective pixel noise. Researchers adopt a data-driven way utilizing generative adversarial networks [1] or flow-based models [1] training on the large dataset with clean and noisy image pairs. However, these methods require extensive clean-noisy pairs to adapt to the new sensor. We use the NLF model, considering its high accuracy and easy accessibility.

1.9. Vignetting

Currently, our dataset has visually negligible vignetting effects. It would be interesting to investigate and simulate how vignetting changes with different camera settings when vignetting is visible.

2. Implementation

2.1. Baselines

Directly using the original value of camera settings as additional input leads to model divergence of all baseline models PC and DL. To fix this issue, we use the α_g, α_n, and α_t calculated in the exposure module as the prior guidance. We adopt the same U-Net structure as [2] in the PC baseline. In DL baselines, we empirically find that associating the weights of the instance normalization layer other than convolutional layers with the camera parameters leads to better convergence results. We have also tried adding noise level maps to the baselines, and the baselines output similar results when noise level maps are provided.

	Number of Matches		
Difficulty	Easy	Medium	Hard
Original	1307	849	514
Augmented	1638	1235	835

Table 3. Quantitative results of feature match

2.2. Auto-exposure mode

The auto-exposure selection algorithm is trained on simulated data rather than captured data because it is time-consuming to manually capture enough real data to train the algorithm well. Our simulator offers an opportunity to avoid tedious data collection process.

In the auto-exposure toy model, we focus only on the auto-exposure mode under a normal lighting condition (i.e., not including dark and extremely bright environment). Thus the selected 64 camera settings state have low ISO (from 100 to 800) and relatively short exposure time (from 0.005s to 0.5s). These settings usually cover the relatively good camera settings for capturing photos with correct exposure. We tested two image scoring standards. The first one is the NIMA model [11], which is trained on a dataset focusing on the aesthetic score of an image. However, in our case, it tends to give the under-exposure image a higher score because their training data contains a lot of high aesthetic score low-light photographs. The other one is the defect detection model [12], which is trained on a dataset concentrating on the detecting image defect, including exposure, noise, and saturation. We found the pre-trained model quite accurate to detect over-exposure and under-exposure. Hence we adopt this defection score in our model. We select 200 images from 50 scenes. For each image, we generate 64 simulated images and get the predicted defection score as ground truth for learning.

We directly adopted the ResNet50 structure for training the score prediction. The last layer is replaced with a linear layer that outputs 64 scores. We empirically find that utilizing KL-divergence as a loss function yields the best prediction result.

2.3. Data augmentation

2.3.1 Data generation

We discuss the data generation for training the d2-net in this section. Initially, since our dataset is captured on static scenes, the matching points of each pair from the same sequence are the points with the same coordinates. However, to increase the robustness of the training process and prevent overfitting, we augment each training pair with a random perspective transform. The transformation matrix is decided by solving the equation from the original coordinates of four corners to the new coordinates. Each corner is shifted by a random number in the range $[-150, 150]$ in each axis. To train with the augmented data, we randomly synthesize 100 new images with different settings using the data from the original scene. The learning rate is $1e-3$, and the model is trained for 30 epochs.

3. More results

3.1. Simulation

More results of the simulation and error maps for each module are shown in Fig. 4.

3.2. HDR

More results of HDR are demonstrated in Fig. 5.

3.3. Feature matching

More qualitative results are demonstrated in Fig. 6. For the quantitative comparison, we randomly sampled 50 pairs from the testing sequences for each category: easy, medium, and hard. Easy case denotes that the exposure stop difference between two images in the pair is 1 or below 1 stop, the medium case is 1 to 3 stops, and the hard case is above 3 stops. The quantitative results are shown in Tab. 3. We observe the improvement of the augmented method, especially for difficult cases.

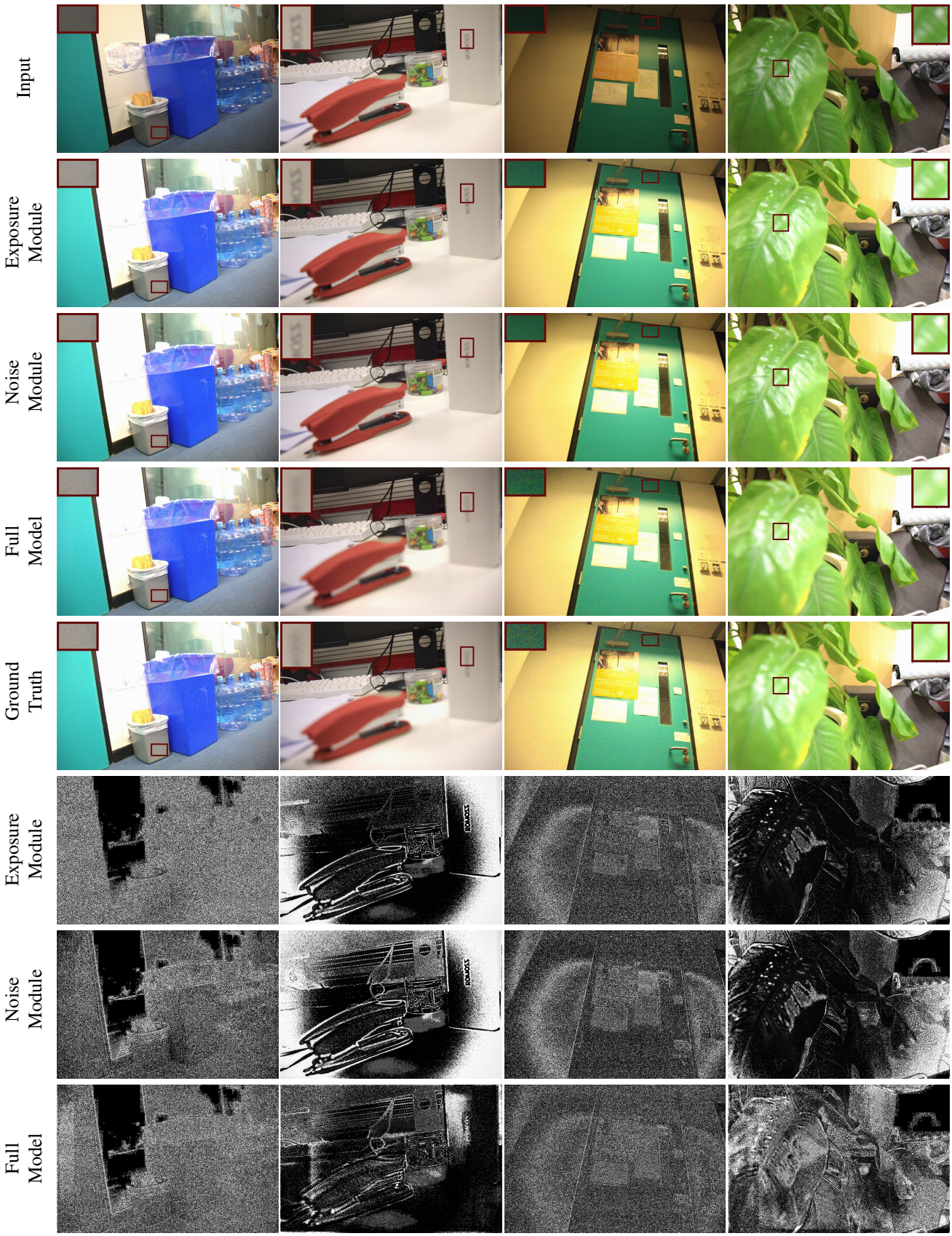


Figure 4. Images generated by each module of our simulator. Best viewed with zoom-in.

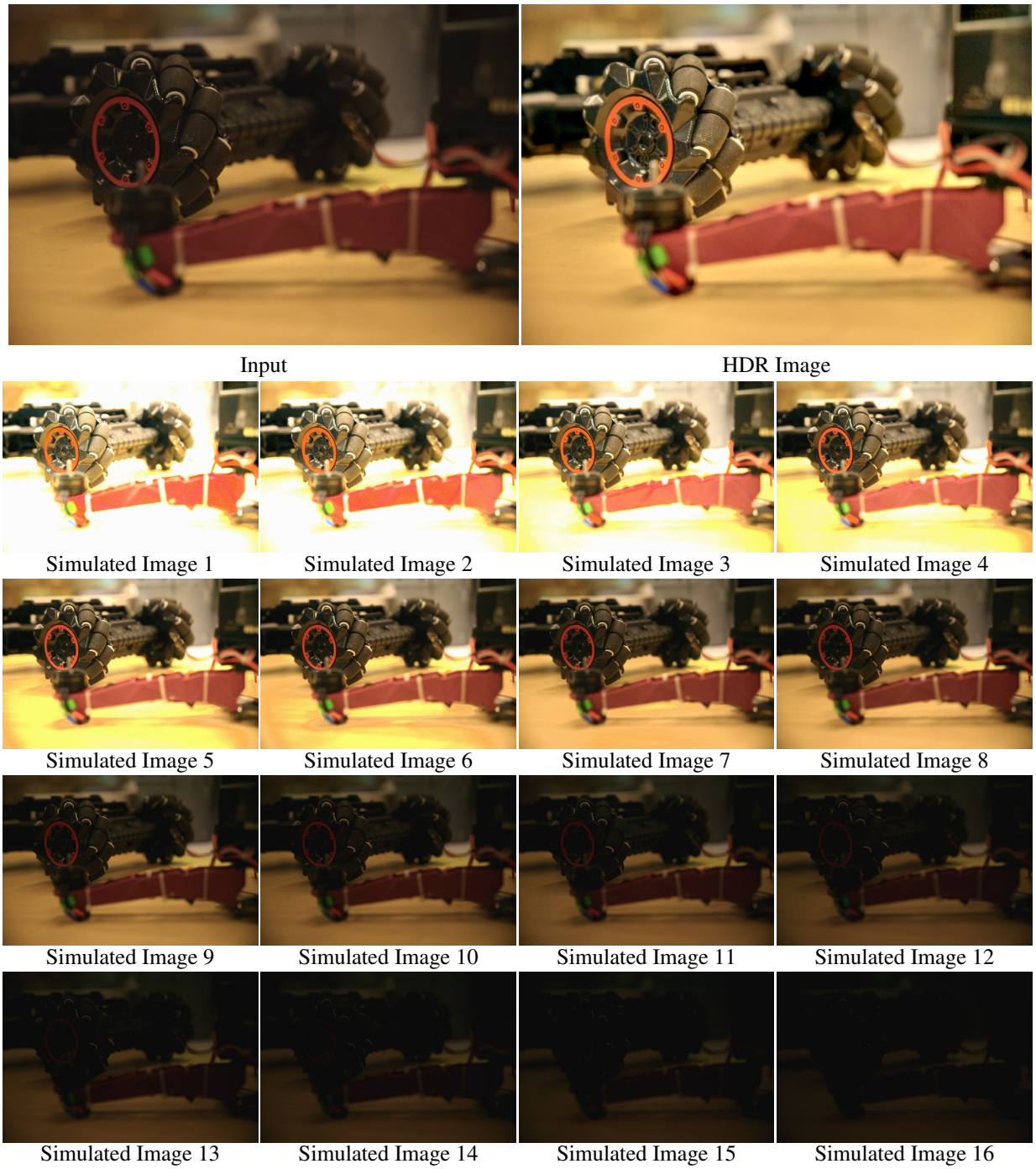
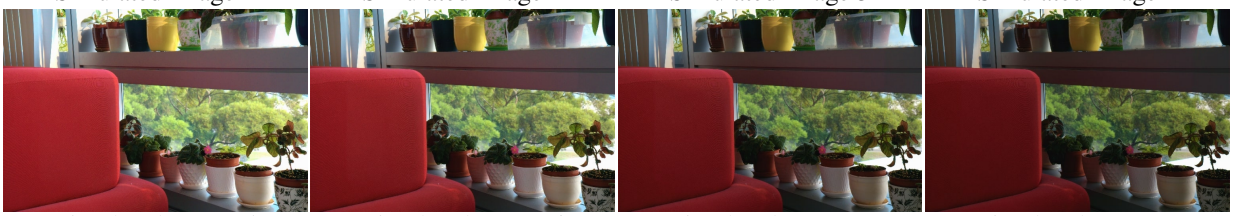


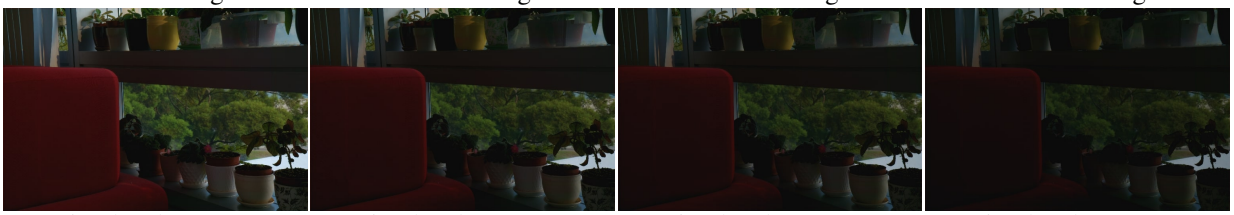
Figure 5. Results of generated HDR images by algorithm [5].

Input HDR Image

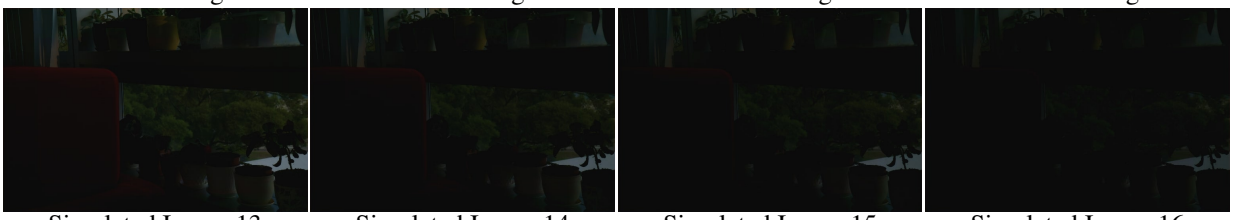
Simulated Image 1 Simulated Image 2 Simulated Image 3 Simulated Image 4



Simulated Image 5 Simulated Image 6 Simulated Image 7 Simulated Image 8



Simulated Image 9 Simulated Image 10 Simulated Image 11 Simulated Image 12



Simulated Image 13 Simulated Image 14 Simulated Image 15 Simulated Image 16

Figure 5. Results of generated HDR images by algorithm [5].

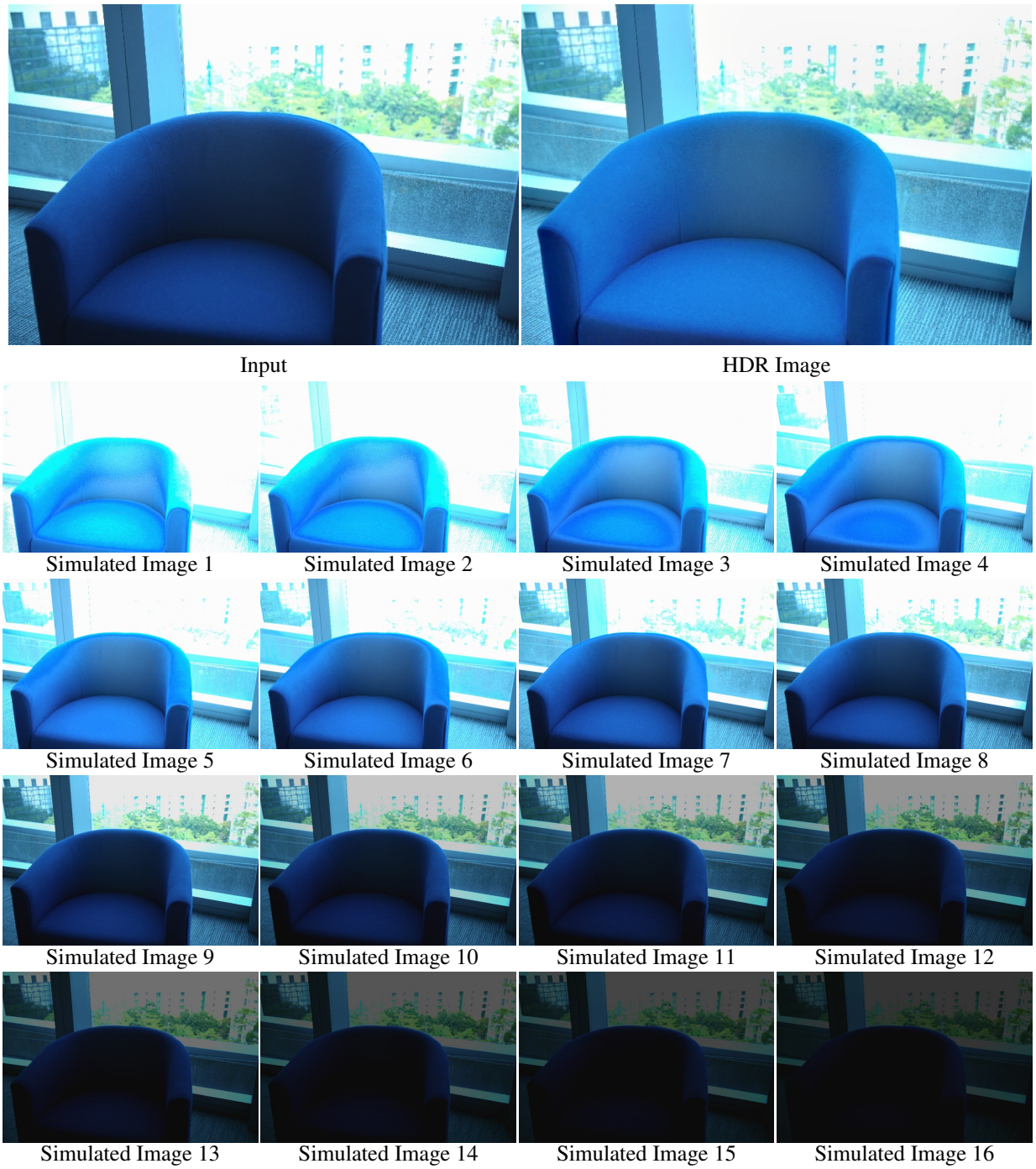
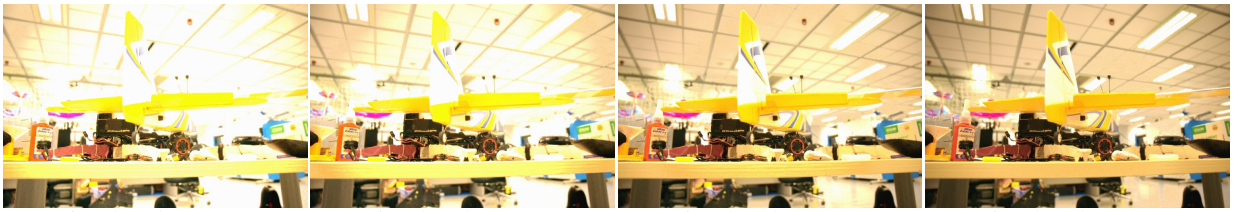


Figure 5. Results of generating HDR image by algorithm [5].

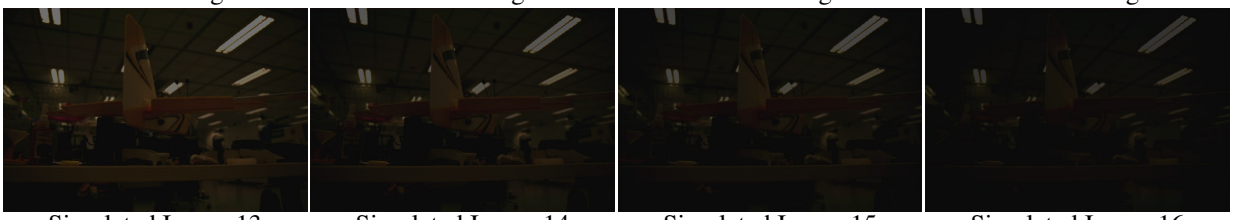
Input HDR Image



Simulated Image 1 Simulated Image 2 Simulated Image 3 Simulated Image 4

Simulated Image 5 Simulated Image 6 Simulated Image 7 Simulated Image 8

Simulated Image 9 Simulated Image 10 Simulated Image 11 Simulated Image 12



Simulated Image 13 Simulated Image 14 Simulated Image 15 Simulated Image 16

Figure 5. Results of generated HDR images by algorithm [5].

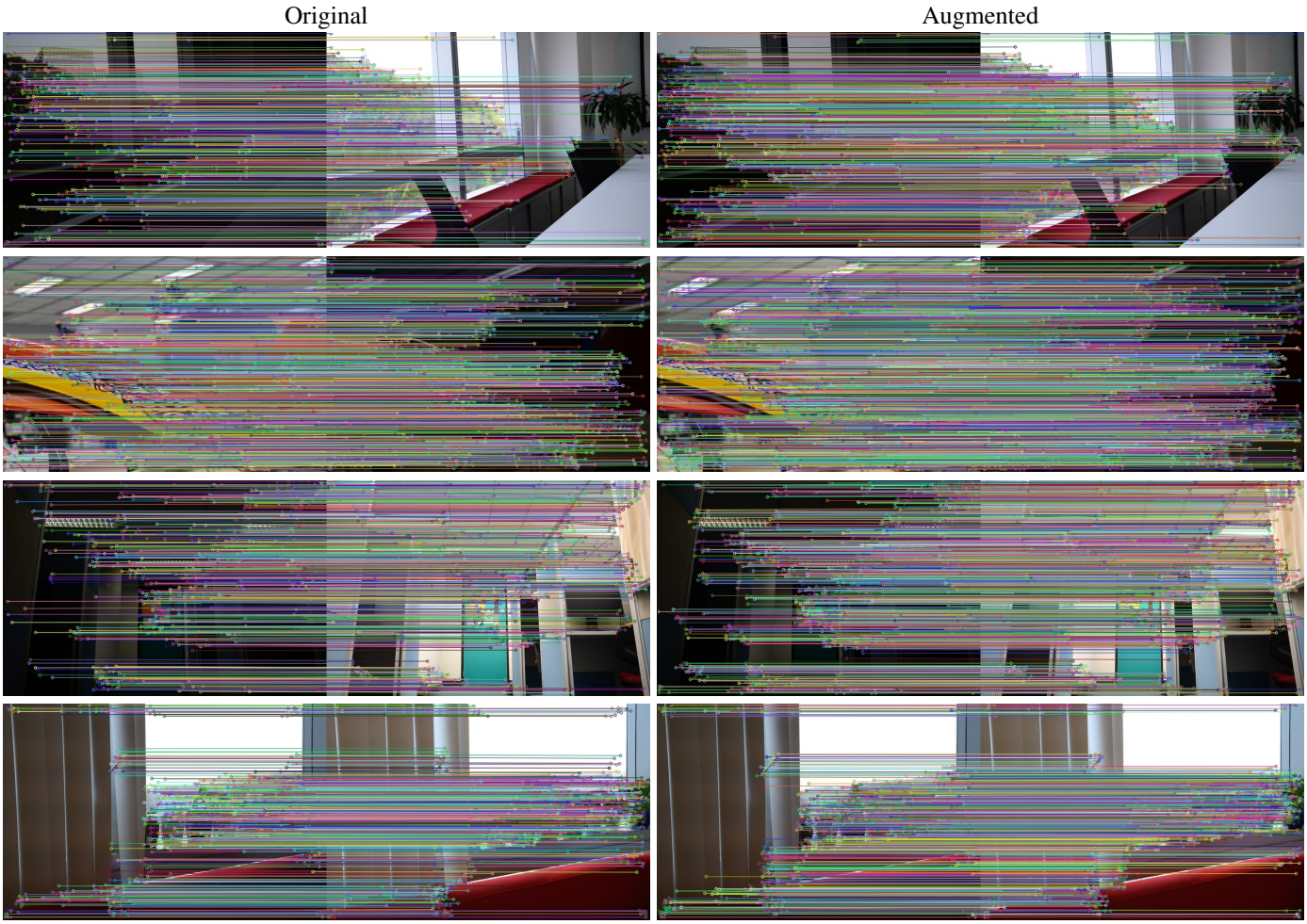


Figure 6. Visual results of local feature matching.

References

[1] Abdelrahman Abdelhamed, Marcus A Brubaker, and Michael S Brown. Noise flow: Noise modeling with conditional normalizing flows. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 3

[2] Qifeng Chen, Jia Xu, and Vladlen Koltun. Fast image processing with fully-convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, 2017. 3

[3] Alessandro Foi. Clipped noisy images: Heteroskedastic modeling and practical denoising. *Signal Processing*, 89(12):2609–2629, 2009. 3

[4] Alessandro Foi, Mejdi Trimeche, Vladimir Katkovnik, and Karen Egiazarian. Practical poissonian-gaussian noise modeling and fitting for single-image raw-data. *IEEE Transactions on Image Processing*, 17(10):1737–1754, 2008. 3

[5] S. Lee, J. S. Park, and N. I. Cho. A multi-exposure image fusion based on the adaptive weights reflecting the relative pixel intensity and global gradient. In *2018 25th IEEE International Conference on Image Processing*, Oct 2018. 6, 7, 8, 9

[6] Ce Liu, Richard Szeliski, Sing Bing Kang, C Lawrence Zitnick, and William T Freeman. Automatic estimation and removal of noise from a single image. *IEEE transactions on pattern analysis and machine intelligence*, 30(2):299–314, 2007. 3

[7] Xinhao Liu, Masayuki Tanaka, and Masatoshi Okutomi. Practical signal-dependent noise parameter estimation from a single noisy image. *IEEE Transactions on Image Processing*, 23(10):4361–4371, 2014. 3

[8] Markku Makitalo and Alessandro Foi. Optimal inversion of the generalized anscombe transformation for poisson-gaussian noise. *IEEE transactions on image processing*, 22(1):91–103, 2012. 3

[9] Amr M Mohsen, Michael F Tompsett, and Carlo H Sèquin. Noise measurements in charge-coupled devices. *IEEE Transactions on Electron Devices*, 22(5):209–218, 1975. 3

[10] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015. 1

[11] Hossein Talebi and Peyman Milanfar. Nima: Neural image assessment. *IEEE Transactions on Image Processing*, 27(8):3998–4011, 2018. 3

[12] Ning Yu, Xiaohui Shen, Zhe Lin, Radomir Mech, and Connelly Barnes. Learning to detect multiple photographic defects. In *2018 IEEE Winter Conference on Applications of Computer Vision*. IEEE, 2018. 3