

# Neural Camera Simulators

Hao Ouyang\*  
HKUST

Zifan Shi\*  
HKUST

Chenyang Lei  
HKUST

Ka Lung Law  
SenseTime

Qifeng Chen  
HKUST

## Abstract

We present a controllable camera simulator based on deep neural networks to synthesize raw image data under different camera settings, including exposure time, ISO, and aperture. The proposed simulator includes an exposure module that utilizes the principle of modern lens designs for correcting the luminance level. It also contains a noise module using the noise level function and an aperture module with adaptive attention to simulate the side effects on noise and defocus blur. To facilitate the learning of a simulator model, we collect a dataset of the 10,000 raw images of 450 scenes with different exposure settings. Quantitative experiments and qualitative comparisons show that our approach outperforms relevant baselines in raw data synthesis on multiple cameras. Furthermore, the camera simulator enables various applications, including large-aperture enhancement, HDR, auto exposure, and data augmentation for training local feature detectors. Our work represents the first attempt to simulate a camera sensor’s behavior leveraging both the advantage of traditional raw sensor features and the power of data-driven deep learning. The code and the dataset are available at [https://github.com/kenouyang/neural\\_image\\_simulator](https://github.com/kenouyang/neural_image_simulator).

## 1. Introduction

Controllable photo-realistic image generation is a new trending research topic [38]. Most recent works focus on learning a certain scene representation conditioned on different viewpoints [27, 26] or lighting [34, 45]. However, in the physical image formulation pipeline, apart from scene, light, and view angle, the camera settings are also important components, which are yet to be investigated. Modern cameras introduce various settings for capturing sceneries, among which the exposure settings (i.e., exposure time or shutter speed, ISO, and aperture size) are most commonly adjusted. As in Fig. 1, cameras capture the scene radiometric characteristics and record them as raw image data. Capturing with different exposure settings not only leads to luminance changes but also results in different side ef-

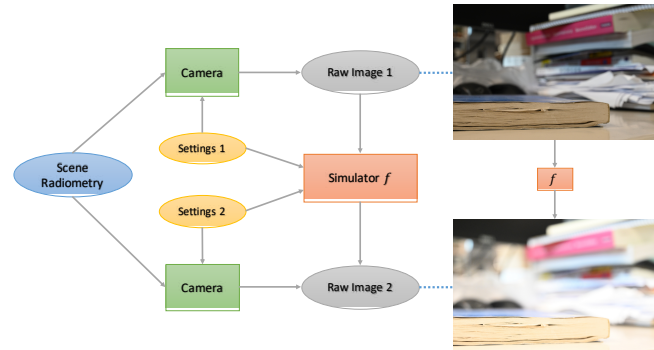


Figure 1. The problem formulation of simulating images with different camera settings. Given the same scene, a camera can capture different images with different camera settings, where the degrees of luminance, noise, and blur vary a lot. Our simulator aims to model the mappings between raw images with different camera settings.

fects: ISO affects the noise presented, and aperture size decides the defocus blur. This paper studies a challenging new task of controllable exposure synthesis with different camera settings. Given the original image, we learn a simulator  $f$  that maps from the old settings to new settings. To capture the essence of changing exposure settings, we directly explore our simulation on raw data rather than monitor-ready images so that any camera signal processing pipeline can be applied [20].

Several features presented in raw data are beneficial for the exposure simulation:

- Raw sensor data directly captures the physical information, such as the light intensity of a scene. To simulate a new illuminance level, we can analyze the change of captured light intensity under new settings based on the physical prior and the design of the modern lens [32].
- The noise distribution model on raw sensor data is relatively robust because of the physical imaging model [14].
- In a close shot scene, the original divisions of blurry and sharp regions provide additional supervision in-

\*Joint first authors

formation. It is possible to learn and locate the blurry regions to magnify the defocus blur.

Therefore, we propose a model that consists of three modules, which progressively learn the change of exposure, noise, and defocus blur. We first adopt an exposure correction module by analyzing the light intensity and refine it using linear regression. In the second module, we utilize the traditional noise-level-function model [13, 14], and train a deep network to further modify the distribution based on real data. Finally, we propose a new attention module to focus on the blurry regions for aperture enhancement. In this paper, we do not consider cases such as removing defocus blur (from large to small aperture in the close shot) or motion blur (with moving objects). They are too scene-dependent to find a general representation.

Since no existing dataset contains enough image sequences that are captured at the same scene with different settings, we collect around 10,000 images of 450 such sequences. The dataset is collected with two modern cameras in diverse environments. Extensive experiments on this dataset demonstrate that our proposed model can progressively generate both visually and statistically similar results at new exposure settings.

The simulator can benefit many low-level computer vision tasks such as generating high-dynamic-range (HDR) photos and enhancing defocus blur. It may also generate realistic images for data augmentation in training deep neural networks. We demonstrate four applications of the simulator: magnifying the defocus blur, generating HDR images, providing environments for training auto-exposure algorithms, and data augmentation for training local feature detectors. The contribution of this paper can be summarized as:

- To the best of our knowledge, we are the first to systematically study the problem of controllable image generation with camera exposure settings, which are important components in the physical image formulation pipeline. By leveraging the physical imaging prior and the power of data-driven deep learning, the model achieves better results than all baselines.
- We demonstrate four applications (large-aperture enhancement, HDR, auto-exposure, and data augmentation) using the proposed simulator.
- We collect a large dataset of raw data sequences of the same scenes under different settings. We believe it can benefit many other computer-vision tasks.

## 2. Related Work

### 2.1. Raw data processing

Learning-based raw data processing has attracted lots of research interests in recent years [7, 41, 44]. Most of the

newly proposed methods focus on generating higher quality monitor-ready images by imitating specific camera image signal processors (ISP). Researchers have explored utilizing a collection of local linear filter sets [18, 21] or deep neural networks [18] to approximate the complex nonlinear operations in traditional ISP. Several follow-up works extend the usage of raw image data to other applications such as low-light image enhancement [7], real scene super-resolution [41], optical zoom adjustment [44] and reflection removal [24, 23]. However, a real camera ISP varies with the camera’s exposure settings [20], which further increases the complexity and leads to the divergence in the training of the learning-based method. In this paper, we focus on the raw-to-raw data generation to make sure that errors in estimating the complicated camera ISP will be excluded.

Previous works have also proposed several ways to synthesize raw data. Gharbi et al. [15] propose to follow the Bayer pattern to assign color pixels from the sRGB images to synthesize mosaicked images. Attracted by the more robust noise model in raw images, Brooks et al. [6] propose to reverse the camera ISP to synthesize raw data for training denoise. However, reverting images from sRGB space is not suitable in our case because guessing the complicated ISP steps (especially the tone curve) introduces additional inaccuracies in simulating the illuminance level. Abdelhamed et al. [1] utilize a flow-based method to synthesize more realistic noise based on camera ISO. Instead of focusing only on noise, our model explores raw data generation in a more general way, which also includes exposure and defocus manipulation.

### 2.2. Aperture supervision

Researchers have proposed a variety of methods for enhancing defocus blur. The most common practice is first to estimate a blurriness map [4, 35, 37, 25, 28] and then refine and propagate it to other pixels. The blurriness map can be retrieved by estimating blur kernels [4], analyzing gradients [35], or learning using deep networks on large-scale datasets [25]. Several follow-up works focus on detecting just-noticeable-blur [31] or blur in small regions [30] by analyzing small patches. Other methods [9, 12] first learn to segment the foreground objects segmentation and followed by re-blurring. Defocus magnification is also associated with depth estimation. Some works [5, 40, 39] render large-aperture images by estimating high-quality depth images as blurriness maps from a single image or stereo image pairs. Srinivasan et al. [33] collect a large dataset of aperture stacks on flowers for inferencing the monocular depth. Instead of estimating the accurate depth value, we train models on image pairs with different apertures for learning how to locate the blurry regions. Moreover, the magnification level on the defocus blur is directly related to the value of the new aperture settings.

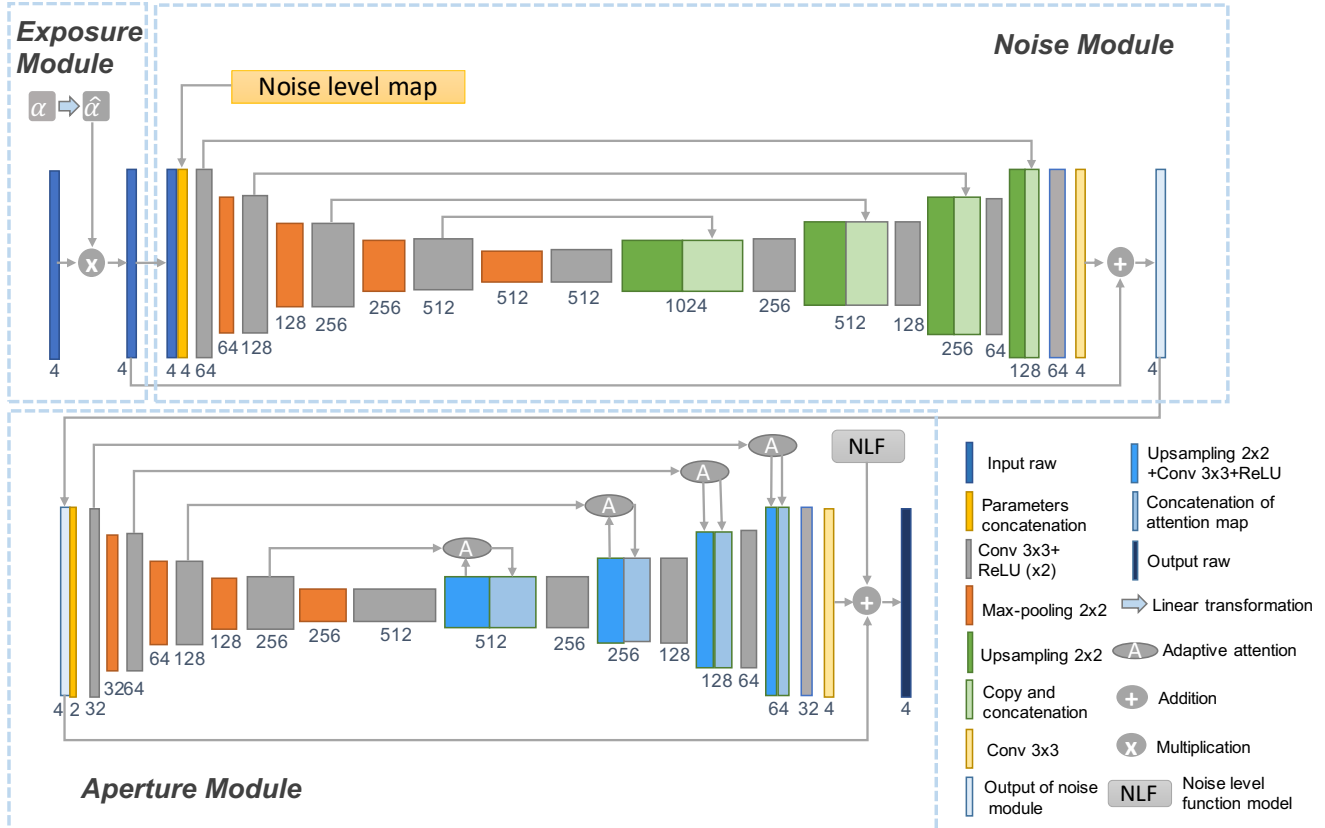


Figure 2. Our simulator synthesizes images under different camera settings based on three modules: exposure module, noise module, and aperture module. The exposure module applies a linear transformation on the input and output camera settings, and thus bringing the brightness of input and output raw to the same level. Noise module takes noise level map of input raw as additional input channels and aims to remove the noise. The aperture module utilizes the input and output aperture parameters to simulate the side effect caused by the aperture. Finally, we adopt the NLF [14] model to model the noise distribution under the output camera setting.

### 3. Method

The overview of our framework is shown in Fig. 2. Please refer to the caption for the general pipeline. Note that we adopt the color processing pipeline from [20] for visualizing raw images in sRGB color space.

#### 3.1. Exposure module

Our exposure correction module aims at adjusting the luminance level of input raw  $I_1 \in \mathbb{R}^{\frac{H}{2} \times \frac{W}{2} \times 4}$  (The raw image is unpacked to a four-channel image where the width and height is half as original  $W$  and  $H$  and normalized between 0 and 1 after black-level subtraction) to the same level of the output  $I_2 \in \mathbb{R}^{\frac{H}{2} \times \frac{W}{2} \times 4}$ . Unlike images in sRGB color space, raw images are linearly dependent on the received photon number. Adjusting camera settings also introduces an approximately proportional relationship on the luminance intensity between two images [29]. Then we respectively analyze the effect of each camera parameter:

**Exposure time** controls the length of time when the shutter is open [3]. The number of photons that reach the sensor is

proportional to the time. Given the input and output exposure time  $t_1$  and  $t_2$ , the multiplier  $\alpha_t$  is defined as  $t_2/t_1$ .

**ISO** controls the light sensitivity of the sensor. Doubling the sensitivity also doubles the recorded light value [29]. Given the input and output ISO  $g_1$  and  $g_2$ , the multiplier  $\alpha_g$  is defined as  $g_2/g_1$ .

**Aperture size** controls the area of the hole which the light can travel through [29]. F-number  $n$  is usually adopted to represent the size and  $\frac{1}{n^2}$  is proportional to the hole area. Given input and output f-numbers  $n_1$  and  $n_2$ , the multiplier  $\alpha_n$  is defined as  $n_1^2/n_2^2$ . In image metadata, the aperture f-number is already approximated (e.g.,  $\sqrt{2}^3$  recorded as 2.8). We may further retrieve the original input and output exposure stop  $s_1$  and  $s_2$  for a more accurate estimation. We have  $s_1 = \text{Round}(6 \log_2(n_1))$  and  $s_2 = \text{Round}(6 \log_2(n_2))$  and the final  $\alpha_n = 2^{(s_1 - s_2)/3}$  (details in the **supplement**).

Combining these three parameters, we get the final  $\alpha$ :

$$\alpha = \alpha_t \alpha_g \alpha_n. \quad (1)$$

The calculated  $\alpha$  may not be accurate enough since camera

settings such as exposure time is not perfectly controlled and has a biased shift. Following the formulation in [2], we can adopt a linear model to further improve the accuracy of our observed data:

$$\hat{\alpha} = w\alpha, \quad (2)$$

$$I_{exp} = clip((I_1 + b) * \hat{\alpha}), \quad (3)$$

where  $clip = \min(\max(y, 0), 1)$  and  $I_{exp}$  is the output of the exposure stage. We initialize  $w$  to 1 and  $b$  to 0. Note that  $b$  is for compensating the inaccuracy of black level. We optimize exposure module using the  $L_1$  loss  $\|I_{exp} - I_2\|_1$ .

**Over-exposed regions** When input  $I_1$  contains pixels that are over-exposed, the real value can be much higher than the recorded value. In this case, the multiplier proposed in the above part does not apply. Thus when training, we exclude pixels with values higher than 0.99.

### 3.2. Noise module

Though noise distribution in a monitor-ready image in sRGB color space can be very complicated because of the non-linear post-processing and compression, the noise in raw color space is a well-studied problem. Noise in raw data is usually represented by heteroscedastic Gaussian composed of two elements: shot noise due to the photon arrival and read noise caused by readout circuitry [14]. Recent works in denoising proves the effectiveness of using this noise level function (NLF) model for noise synthesis [6]. We also adopt this noise model as a prior in our noise module. We first reduce the primary noise in the  $I_{exp}$  and then add back the NLF noise. Modern cameras provide calibrated NLF parameters for each ISO settings under normal temperature. The input read noise  $\lambda_{read}^1$ , input shot noise  $\lambda_{shot}^1$ , output read noise  $\lambda_{read}^2$  and output shot noise  $\lambda_{shot}^2$  can be directly retrieved from the camera metadata. We can represent the observed pixel value  $y_1$  and  $y_2$  in input and output raw data using the true signal  $x_1$  and  $x_2$ :

$$y_1 \sim \mathcal{N}(\mu = x_1, \sigma_1^2 = \lambda_{read}^1 + \lambda_{shot}^1 x_1), \quad (4)$$

$$y_2 \sim \mathcal{N}(\mu = x_2, \sigma_2^2 = \lambda_{read}^2 + \lambda_{shot}^2 x_2), \quad (5)$$

where  $\mathcal{N}$  represents Gaussian distribution. After the exposure correction step, the distribution of pixels in  $I_{exp}$  becomes:

$$\hat{y}_1 \sim \mathcal{N}(\mu = \hat{\alpha}(x_1 + b), \hat{\sigma}_1^2 = \hat{\alpha}^2 \lambda_{read}^1 + \hat{\alpha} \lambda_{shot}^1 x_1). \quad (6)$$

We approximately estimate the noise level map of the input noise and output noise based on the first-stage output  $I_{exp}$  and the modified input noise parameters  $\hat{\sigma}_1$ . The noise level map is concatenated to the input  $I_{exp}$  as an additional prior. We adopt the same U-net structure used in [6] and initialize it using their pre-trained model. As suggested by [43], we

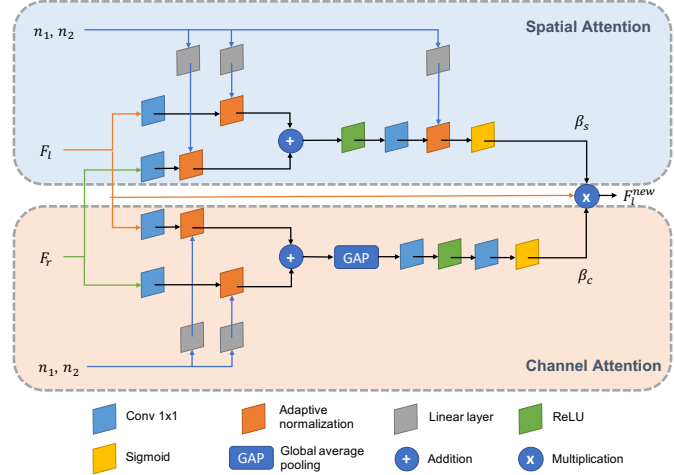


Figure 3. Adaptive attention module comprises two parts: spatial attention module and channel attention module. The input and output aperture f-numbers are used to adjust the parameters in the normalization layer.

empirically find that using residual  $I_2 - I_{exp}$  is more effective in learning the high-frequency information.  $L_1$  loss  $\|I_{rs} - (I_2 - I_{exp})\|_1$  is adopted for this module. We add back  $I_{exp}$  to the learned residual  $I_{rs}$  for retrieving the final output  $I_{ns}$  in this stage. When training this stage, we adopt a special pair selection strategy to ensure that the output raw data contains little noise.

### 3.3. Aperture module

Increasing aperture size not only brightens the images but also enlarges the defocus blur because it shallows the depth-of-field. Recovering details from defocus blur highly depend on scene content. However, its inverse direction, which is also known as defocus magnification, is learnable by enhancing only the blurry regions. Our aperture model is designed to simulate the blur circle according to the new aperture settings. We adopt an adaptive attention module, as shown in Fig. 3, which calibrates the features by considering spatial attention and channel attention at the same time. Uniquely, this module takes the input and output aperture f-number  $n_1$  and  $n_2$  to adjust the parameters of the normalization layer. The adaptive aperture layer is formulated as:

$$\mathcal{A}(x) = (w_\sigma[n_1, n_2] + b_\sigma) \left( \frac{x - \mu(x)}{\sigma(x)} \right) + (w_\mu[n_1, n_2] + b_\mu), \quad (7)$$

where  $w_\sigma$ ,  $w_\mu$ ,  $b_\sigma$  and  $b_\mu$  are parameters to be learnt for linear transformations.  $\mu$  and  $\sigma$  are mean and variance respectively. "[ ]" denotes the concatenation of f-numbers.  $x$  is the instance in the feature maps.

Since we built adaptive attention module on a U-net architecture, the inputs of the adaptive attention module are



original feature maps  $F_l \in \mathbb{R}^{H_l \times W_l \times C_l}$  and gating signal  $F_r \in \mathbb{R}^{H_r \times W_r \times C_r}$ . Utilizing the above adaptive aperture layer, we can design the spatial and channel-wise attention in our case.

Spatial attention module explores the inter-pixels relationships in feature maps  $F_l$  and  $F_r$ , and computes an attention map  $\beta_s$  to rescale feature maps  $F_l$ . We first transform  $F_l$  and  $F_r$  independently with a  $1 \times 1$  convolutional layer and an aperture layer. After a ReLU activation on the addition of the transformed feature maps, another  $1 \times 1$  convolutional layer and aperture layer are applied to get the final spatial attention map  $\beta_s$  followed by a sigmoid activation.

In the channel attention module, channel-wise relationships are captured to rescale  $F_l$  in the channel dimension. After the linear transformations on  $F_l$  and  $F_r$  same as above, a global average pooling layer is used to squeeze the feature maps to a feature vector  $v$  with  $C_l$  dimension. The excitation on  $v$  is completed by two convolutional layers, a ReLU activation, and a sigmoid activation, which gives us the channel attention map  $\beta_c$ . With the spatial and channel attention maps,  $F_l$  will be calibrated as:

$$F_l^{new} = F_l \times \beta_s \times \beta_c. \quad (8)$$

The modified feature  $F_l^{new}$  can provide effective information when concatenating with up-sampled features in the U-net structure. The input and output f-numbers are concatenated as additional channels to the output  $I_{n.s}$  in the noise module. We also adopt residual learning [43] and utilize  $L_1$  loss in this module. Adding the residual learned in the aperture module back to  $I_{n.s}$  gives us the final output of the aperture module. To complete the entire simulation process, we add the synthesized output NLF noise to the output raw data.

## 4. Experiments

### 4.1. Dataset

We collect a new dataset for simulating images with different settings. The dataset contains around 10,000 raw images, which are divided into 450 image sequences in different scenes. In each sequence, we capture 20 to 25 images with different camera settings. ISO, exposure time, and aperture f-number are sampled respectively from the discrete levels in the range [100, 16000], [1/8000s, 4s], and [4.0, 22.0].

The collected dataset contains both indoor and outdoor images taken under different lighting conditions. The illuminance of these scenes varies from 1 lux to 20000 lux. We adapt the camera settings for each scene to capture more information. In brighter scenes, we use lower ISO, shorter exposure time, and smaller aperture size on average for capturing image sequences. In consideration of the diversity

of the dataset, we use two different cameras, DSLR camera Canon 70D and mirrorless camera Nikon Z6, for data acquisition. The captured scene sequence has to be static. We mount cameras on sturdy tripods and utilize corresponding remote control mobile applications for photo shooting to avoid unintended camera movement. We manually filter out all images containing a moving object.

### 4.2. Experimental setup

**Data preparation.** We randomly split the dataset into a training set of 90% of image sequences and a testing set of 10% of image sequences. As pre-processing, we also unpack the Bayer raw pattern to four RRGB channels and deduct the recorded black level. The unpacked raw data is then normalized between 0 and 1. We obtain the camera parameters (ISO, exposure time, and aperture) and pre-calibrated noise level function from the camera meta-data. We also retrieve the color processing pipeline parameters (white balance, tone curve, and color matrix), which we utilize for rendering from raw-RGB to sRGB color space for visualization.

**Pair selection.** We adopt different pair selection strategies to facilitate the training for each module. In the exposure training stage, any random pairs from the same scene sequence can be selected. In the noise manipulation stage, we only choose pairs in which the output has low ISO settings (100, 200, and 400). Similarly, we only choose pairs from small to large aperture sizes to magnify the defocus blur.

**Training details.** We train the exposure module for 5 epochs, the noise module for 30 epochs, and the aperture module for 30 epochs separately. The model is then jointly finetuned for another 10 epochs. The batch size is set to 2. We use Adam Optimizer with a learning rate of 1e-3 for all three modules and reduce the learning rate by a factor of 10 every 20 epochs. The entire training process takes about 48 hours on two NVIDIA RTX 2080 Ti GPUs.

### 4.3. Baselines

Our designed baseline models aim to learn a direct mapping to generate raw images given different camera settings.

**Parameter concatenation (PC).** Image to image translation has shown effectiveness in multiple image generation tasks [8, 17]. In our case, we can easily modify these approaches to adapt to a raw-to-raw translation task. We change the input and output of the model proposed in [8] to unpacked raw images. Other settings, including loss types and network structures, remain the same. The model takes the camera input parameters and output parameters settings as additional input channels. The baselines with and without noise level maps generate similar results.

**Decouple learning (DL).** Another structure applicable to our case is DecoupleNet [11]. The model learns param-

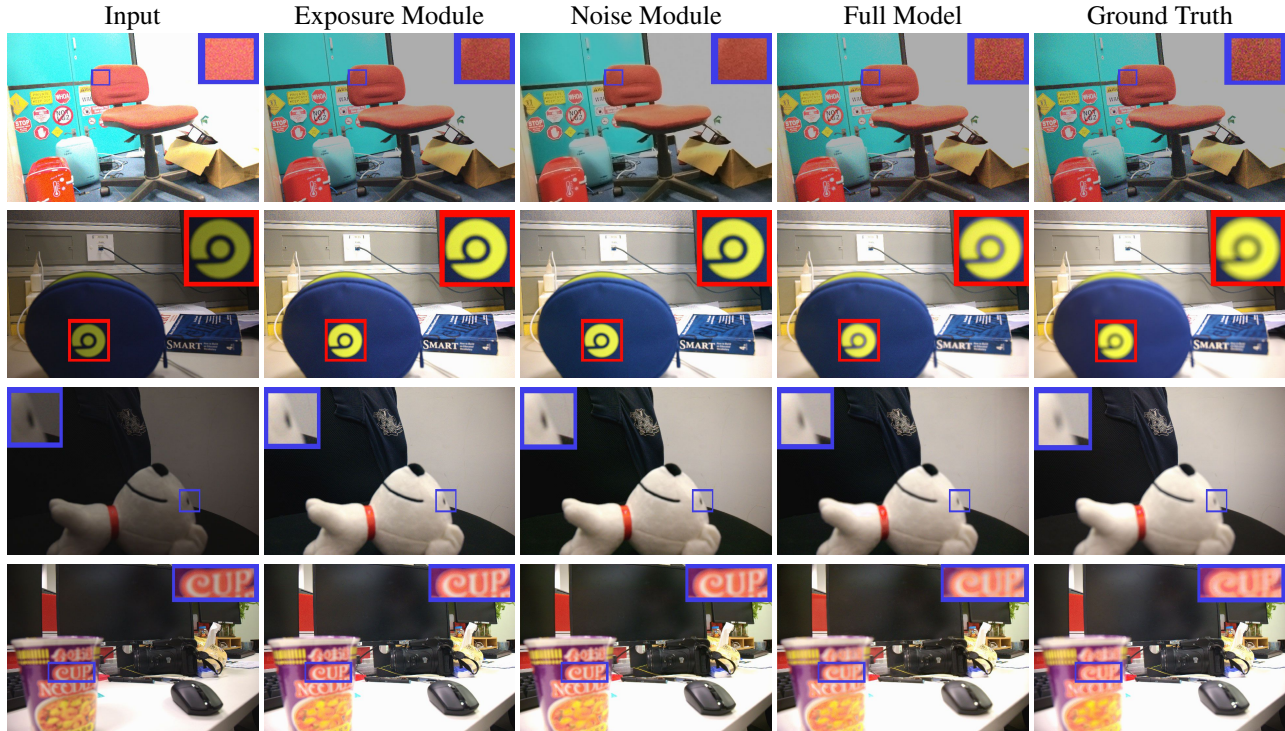


Figure 4. Images generated by each module of our simulator. Zoom-in to view details.

|                         | PC [8] | DL [11] | Ours (EXP) | Ours (NS) | Ours (Full model) |
|-------------------------|--------|---------|------------|-----------|-------------------|
| <b>PSNR (Nikon Z6)</b>  | 18.78  | 34.34   | 35.31      | 35.36     | <b>36.10</b>      |
| <b>SSIM (Nikon Z6)</b>  | 0.542  | 0.879   | 0.903      | 0.911     | <b>0.923</b>      |
| <b>PSNR (Canon 70D)</b> | 18.96  | 24.86   | 33.53      | 33.49     | <b>34.28</b>      |
| <b>SSIM (Canon 70D)</b> | 0.413  | 0.429   | 0.831      | 0.837     | <b>0.846</b>      |

Table 1. PSNR and SSIM of different methods and different stages of our model: PC [8], DL [11], EXP (Ours after exposure module), NS (Ours after noise module), and Full (Our full model).

terized image operators by directly associating the operator parameters with the weights of convolutional layers. Similarly, we need to modify the input and output to raw data. Operator parameters are replaced by input and output camera parameters in our case.

#### 4.4. Image quality evaluation

Fig. 4 visualizes the output of each module of our method, and Fig. 5 provides a qualitative comparison of the visual quality of raw data simulated by different methods. Our simulated raw has higher visual fidelity and fewer artifacts than the baselines. Without prior knowledge of lens design, baseline models struggle to figure out the relationship between camera settings and output raw images. As a result, they generate images with incorrect illuminance and biased color.

Table 1 shows the quantitative analysis of our model and baselines trained on the Nikon Dataset. We also evaluate

the average PSNR and SSIM of the generated raw data after each module. Note that the PSNR and SSIM in step NS do not show significant improvement to step EXP because they highly depend on the target settings of the simulator. For instance, if the input raw is noisy and the target is to add noise, then PSNR and SSIM in step NS can be worse than those of step EXP due to the denoising function of the noise module. More analysis on specific simulation direction (e.g., from low to high ISO) can be found in the **supplement**.

#### 4.5. Cross-dataset validation

We conduct a cross-dataset evaluation to validate the generalization ability of the proposed model. We simulate Canon 70D raw data using the model trained on the Nikon Z6 dataset. Fig. 6 shows that our model generates stable simulation results. Table 1 shows the quantitative results, which implies that baseline methods suffer from severe degradation when testing on different sensors.



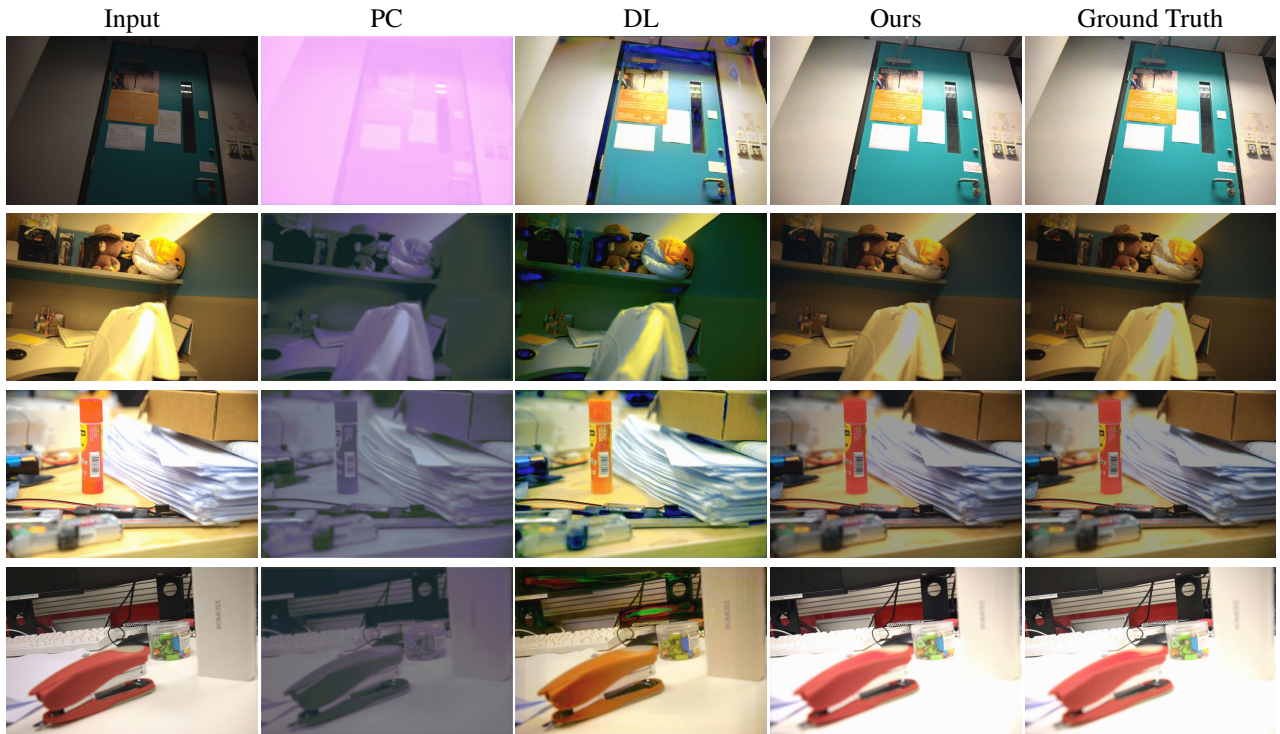


Figure 5. Comparison with different methods. Simple parameter concatenation [8] fails to generate natural images. Decouple learning [11] can generate correct illuminance level but with incorrect color. Our method achieves the most similar result to the ground truth image. Zoom-in to view details.



Figure 6. Cross-dataset validation on the Canon 70D dataset with model trained on the Nikon Z6 dataset. Zoom-in to view details.

## 5. Applications

**Large-aperture enhancement.** The most straightforward application of our proposed structure is to magnify the defocus blur. As shown in Fig. 7, our model can correctly enhance the blurry region given a new aperture setting. Exposure is adjusted to keep the brightness unchanged. Compared with results from [28], our model generates blurry content that is more consistent with the input image.

**HDR images.** As our simulator can generate images with other exposure settings, a very natural application is to generate HDR images. Our HDR pipeline is straightforward: we first simulate multiple output raw images from the in-

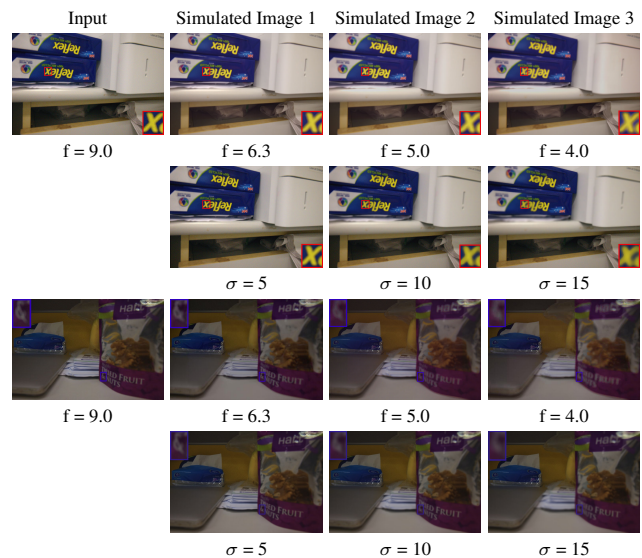


Figure 7. Results of defocus magnification. The first and third rows are generated by our proposed simulator under different aperture settings. The second and fourth rows are generated by [28] with different Gaussian kernels  $\sigma$ . Zoom-in to view details.

put raw using a group of camera exposure settings. We set ISO to be as small as possible for reducing the image noise and calculate the exposure time by shifting original expo-

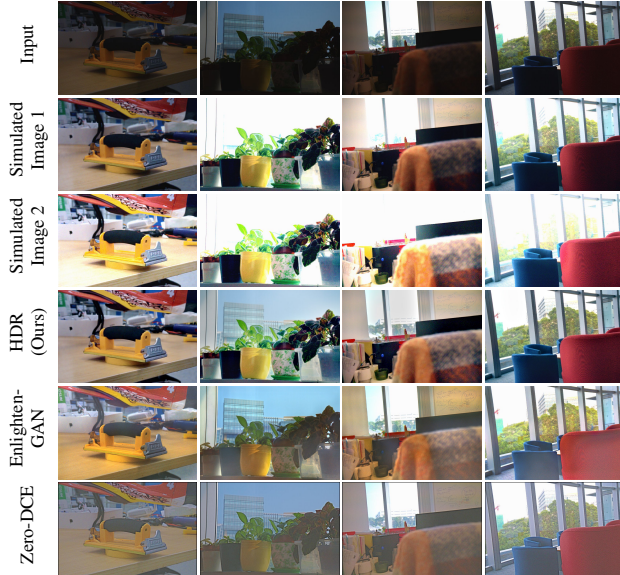


Figure 8. Results of HDR image generation. We generate HDR images by [22]. We pick two representative simulated images for display. Results from EnlightenGAN [19] and Zero-DCE [16] are listed in the last two rows. Zoom-in to view details.

sure in every  $0.5Ev$  stop in the range  $[-4Ev, 4Ev]$ . The output aperture keeps unchanged. We then convert raw data to sRGB-space images and fuse them using the approach proposed by [22] to generate HDR output. In this pipeline, we can fully exploit the information at each exposure level and reduce the noise introduced when brightening dark regions. As Fig. 8 shows, our fused output brings back the missing details and appears more attractive than those generated by recent works [19, 16].

**Auto-exposure mode.** Traditional auto-exposure algorithms rely on light metering based on reference regions. However, metering is inaccurate when no gray objects exist for reference (for example, the snow scenes.) Our proposed model can provide a simulation environment for training the auto-exposure algorithm. We can train the selection algorithm by direct regression or by trial-and-error in reinforcement learning. Any aesthetic standards can be adopted for evaluating the quality of output images with new settings. To validate the feasibility of the proposed auto-exposure pipeline, we build a toy exposure selection model. We pre-defined 64 states with different camera settings for selection and simulate images for each state. We test two different standards for assessing the quality of the images: estimating aesthetic score [36] and detecting the image defections, including noise, exposure, and white balance [42]. We then train a neural network by regression for predicting the score of each state. In Fig. 9, we show the simulation results and the final selected settings (more details in the **supplement**).

**Data augmentation.** We test the feasibility of using the

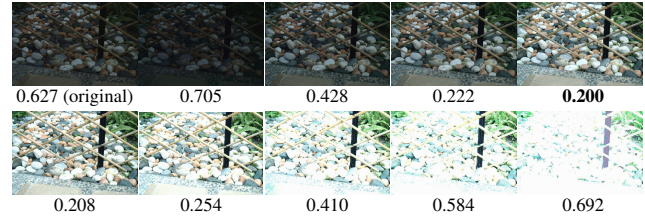


Figure 9. Defection scores (the smaller, the better) for the original and simulated images evaluated with [42]. We pick nine representative simulated images for display.

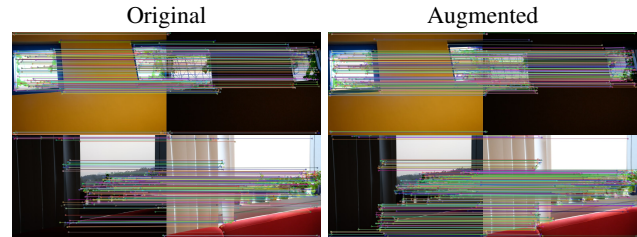


Figure 10. Visual results of local feature matching.

synthesized images as augmented data for training local image features. We conduct experiments using D2-Net [10], which is a local-feature extractor where the descriptor and detector are jointly trained. We use image pairs from the same sequences with homography transform to generate the ground truth matching points. Fifty scenes from the Nikon dataset are used for training, and five scenes are used for testing. We compare the performance of two models: D2-Net trained with only the original augmentation (including color jittering) and D2-Net trained with further augmented data from the simulator. In Fig. 10, the model trained with our augmented data finds more matches in extreme cases. The number of valid matches respectively increases from 253 to 371 and from 358 to 515 for the first and second examples. More details and quantitative results are in the **supplement**.

## 6. Discussion

In this paper, we systematically study the synthesis of one important component in the physical image pipeline: camera exposure settings. We address this novel problem by deep models with physical prior. Experiments demonstrate promising results with correctly modified exposure, noise, and defocus blur. This work also provides many opportunities for future work. The major limitation is that the model only considers static scenes and fails in handling the deblur tasks. In images where no blurry region exists as guidance, our model fails in enhancing the aperture. For noise simulation, we may also adopt a learning-based method [1] rather than NLF functions. We expect future work to yield further improvements in the quality of simulated raw data.



## 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. 2, 8
- [2] Abdelrahman Abdelhamed, Stephen Lin, and Michael S Brown. A high-quality denoising dataset for smartphone cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 4
- [3] Elizabeth Allen and Sophie Triantaphillidou. *The manual of photography and digital imaging*. Focal Press, 2012. 3
- [4] Soonmin Bae and Frédo Durand. Defocus magnification. In *Computer Graphics Forum*, volume 26, pages 571–579. Wiley Online Library, 2007. 2
- [5] Jonathan T Barron, Andrew Adams, YiChang Shih, and Carlos Hernández. Fast bilateral-space stereo for synthetic defocus. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 2
- [6] Tim Brooks, Ben Mildenhall, Tianfan Xue, Jiawen Chen, Dillon Sharlet, and Jonathan T Barron. Unprocessing images for learned raw denoising. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 2, 4
- [7] Chen Chen, Qifeng Chen, Jia Xu, and Vladlen Koltun. Learning to see in the dark. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2
- [8] 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. 5, 6, 7
- [9] Weihai Chen, Fei Kou, Changyun Wen, and Zhengguo Li. Automatic synthetic background defocus for a single portrait image. *IEEE Transactions on Consumer Electronics*, 63(3):234–242, 2017. 2
- [10] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-net: A trainable cnn for joint description and detection of local features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8092–8101, 2019. 8
- [11] Qingnan Fan, Dongdong Chen, Lu Yuan, Gang Hua, Nenghai Yu, and Baoquan Chen. Decouple learning for parameterized image operators. In *Proceedings of the European Conference on Computer Vision*, 2018. 5, 6, 7
- [12] Farshid Farhat, Mohammad Mahdi Kamani, Sahil Mishra, and James Z Wang. Intelligent portrait composition assistance: Integrating deep-learned models and photography idea retrieval. In *Proceedings of the on Thematic Workshops of ACM Multimedia 2017*, pages 17–25, 2017. 2
- [13] Alessandro Foi. Clipped noisy images: Heteroskedastic modeling and practical denoising. *Signal Processing*, 89(12):2609–2629, 2009. 2
- [14] 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. 1, 2, 3, 4
- [15] Michaël Gharbi, Gaurav Chaurasia, Sylvain Paris, and Frédo Durand. Deep joint demosaicking and denoising. *ACM Transactions on Graphics*, 35(6):1–12, 2016. 2
- [16] Chunle Guo, Chongyi Li, Jichang Guo, Chen Change Loy, Junhui Hou, Sam Kwong, and Runmin Cong. Zero-reference deep curve estimation for low-light image enhancement. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 1780–1789, June 2020. 8
- [17] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017. 5
- [18] Haomiao Jiang, Qiyuan Tian, Joyce Farrell, and Brian A Wandell. Learning the image processing pipeline. *IEEE Transactions on Image Processing*, 26(10):5032–5042, 2017. 2
- [19] Yifan Jiang, Xinyu Gong, Ding Liu, Yu Cheng, Chen Fang, Xiaohui Shen, Jianchao Yang, Pan Zhou, and Zhangyang Wang. Enlighten: Deep light enhancement without paired supervision. *arXiv preprint arXiv:1906.06972*, 2019. 8
- [20] Hakki Can Karaimer and Michael S Brown. A software platform for manipulating the camera imaging pipeline. In *European Conference on Computer Vision*. Springer, 2016. 1, 2, 3
- [21] Steven P Linsel and Brian A Wandell. Learning of image processing pipeline for digital imaging devices, Mar. 18 2014. US Patent 8,675,105. 2
- [22] 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. 8
- [23] Chenyang Lei and Qifeng Chen. Robust reflection removal with reflection-free flash-only cues. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2
- [24] Chenyang Lei, Xuhua Huang, Mengdi Zhang, Qiong Yan, Wenxiu Sun, and Qifeng Chen. Polarized reflection removal with perfect alignment in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1750–1758, 2020. 2
- [25] Kede Ma, Huan Fu, Tongliang Liu, Zhou Wang, and Dacheng Tao. Deep blur mapping: Exploiting high-level semantics by deep neural networks. *IEEE Transactions on Image Processing*, 27(10):5155–5166, 2018. 2
- [26] Moustafa Meshry, Dan B Goldman, Sameh Khamis, Hugues Hoppe, Rohit Pandey, Noah Snavely, and Ricardo Martin-Brualla. Neural re-rendering in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6878–6887, 2019. 1
- [27] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *arXiv preprint arXiv:2003.08934*, 2020. 1



- [28] Jinsun Park, Yu-Wing Tai, Donghyeon Cho, and In So Kweon. A unified approach of multi-scale deep and hand-crafted features for defocus estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2, 7
- [29] Sidney F Ray, Wally Axford, and Geoffrey G Attridge. *The Manual of Photography: Photographic and Digital Imaging*. Elsevier Science & Technology, 2000. 3
- [30] Jianping Shi, Xin Tao, Li Xu, and Jiaya Jia. Break ames room illusion: depth from general single images. *ACM Transactions on Graphics*, 34(6):1–11, 2015. 2
- [31] Jianping Shi, Li Xu, and Jiaya Jia. Just noticeable defocus blur detection and estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 2
- [32] Warren J Smith. *Modern lens design*. 2005. 1
- [33] Pratul P Srinivasan, Rahul Garg, Neal Wadhwa, Ren Ng, and Jonathan T Barron. Aperture supervision for monocular depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2
- [34] Tiancheng Sun, Jonathan T Barron, Yun-Ta Tsai, Zexiang Xu, Xueming Yu, Graham Fyffe, Christoph Rhemann, Jay Busch, Paul E Debevec, and Ravi Ramamoorthi. Single image portrait relighting. *ACM Trans. Graph.*, 38(4):79–1, 2019. 1
- [35] Yu-Wing Tai and Michael S Brown. Single image defocus map estimation using local contrast prior. In *2009 16th IEEE International Conference on Image Processing*. IEEE, 2009. 2
- [36] Hossein Talebi and Peyman Milanfar. Nima: Neural image assessment. *IEEE Transactions on Image Processing*, 27(8):3998–4011, 2018. 8
- [37] Chang Tang, Chunping Hou, and Zhanjie Song. Defocus map estimation from a single image via spectrum contrast. *Optics letters*, 38(10):1706–1708, 2013. 2
- [38] Ayush Tewari, Ohad Fried, Justus Thies, Vincent Sitzmann, Stephen Lombardi, Kalyan Sunkavalli, Ricardo Martin-Brualla, Tomas Simon, Jason Saragih, Matthias Nießner, et al. State of the art on neural rendering. *arXiv preprint arXiv:2004.03805*, 2020. 1
- [39] Neal Wadhwa, Rahul Garg, David E Jacobs, Bryan E Feldman, Nori Kanazawa, Robert Carroll, Yair Movshovitz-Attias, Jonathan T Barron, Yael Pritch, and Marc Levoy. Synthetic depth-of-field with a single-camera mobile phone. *ACM Transactions on Graphics (ToG)*, 37(4):1–13, 2018. 2
- [40] Lijun Wang, Xiaohui Shen, Jianming Zhang, Oliver Wang, Zhe Lin, Chih-Yao Hsieh, Sarah Kong, and Huchuan Lu. Deeplens: Shallow depth of field from a single image. *arXiv preprint arXiv:1810.08100*, 2018. 2
- [41] Xiangyu Xu, Yongrui Ma, and Wenxiu Sun. Towards real scene super-resolution with raw images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 2
- [42] 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. 8
- [43] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, 2017. 4, 5
- [44] Xuaner Zhang, Qifeng Chen, Ren Ng, and Vladlen Koltun. Zoom to learn, learn to zoom. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 2
- [45] Hao Zhou, Sunil Hadap, Kalyan Sunkavalli, and David W Jacobs. Deep single-image portrait relighting. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7194–7202, 2019. 1