Supplyment Material for Manuscript: Efficient Depth-and Spatially-Varying Image Simulation for Defocus Deblur

Xinge Yang[1]* Chuong Nguyen[2] Wenbin Wang[2] Kaizhang Kang[1]
Wolfgang Heidrich[1] Xiaoxing Li[2]

KAUST[1] Meta Reality Labs[2]

In this Supplemental Material, we provide more implementation details (Sec. 1), more experimental results (Sec. 2), and extra discussion about the limitations (Sec. 3) for our proposed the approach.

1. Implementation details

1.1. Detailed synthetic dataset generation

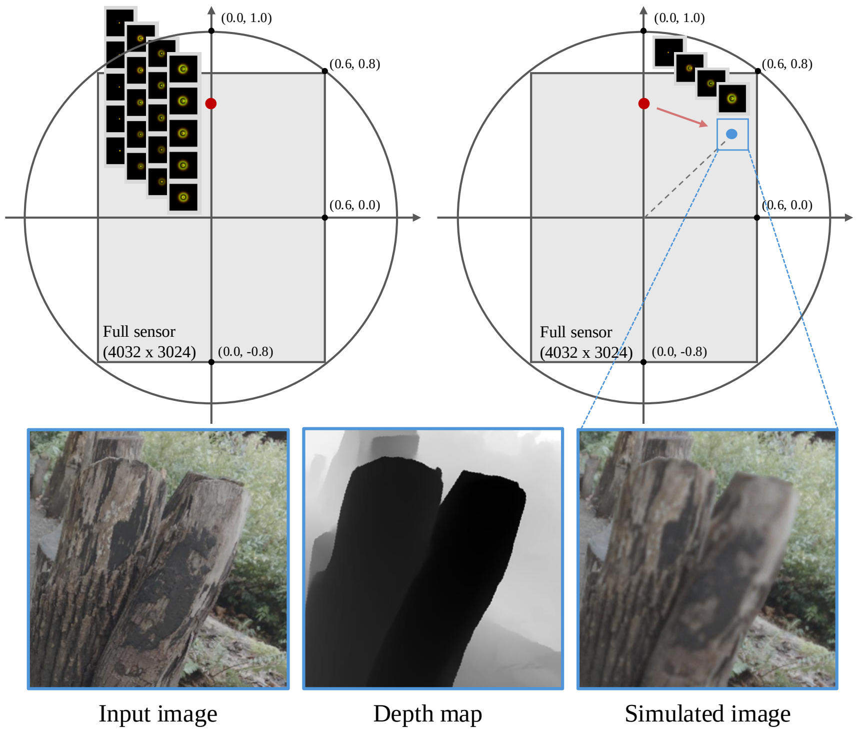


Figure S1. **Implementation details for spatially varying depth-varying defocus blur simulation.** Top left: before the experiments, PSFs are computed in ZEMAX for various fields of view (along the y-axis) and depths. Top right: during each iteration, a random rotation is applied to sample a low-resolution image patch at a random spatial position. The center position of the sampled image patch is determined using both rotation and distortion parameters. PSFs are also rotated according to the angle to account for rotational symmetry. Bottom: input all-in-focus images and the estimated depth map are used for depth-varying defocus image simulation.

*Work partially done during Meta internship. Inquiries can be directed to: xinge.yang@kaust.edu.sa

During the network training stage, we generate smaller image patches (512×512) and use positional channels to guide the network for spatially varying defocus blur. This setting is from the consideration of both speed, including image simulation and network training, and memory consumption, as directly simulating and training on 12-megapixel (4032×3024) full-resolution images is almost impossible. Before training, point spread functions (PSFs) at different fields of view (FoV) are loaded from previous simulation results obtained using the optical design software ZEMAX [9]. We sample PSFs along the y-axis (Fig. S1) from the center ($0°$) to the full diagonal FoV ($54.4°$). During each iteration, we first randomly sample a FoV and a rotation angle between 0 and 2π radians. We then rotate the PSFs to obtain depth-varying PSFs at the desired positions. Next, we rotate points around the origin point from the y-axis and apply distortion to determine the center positions of the target image patches. Note that the rotation applied to the PSFs is counterclockwise, whereas the rotation for determining the center positions is clockwise. We compute distortion only for the center positions and sample uniform positional grids as positional channels, consistent with the inference stage for 12-megapixel images, where uniform positional grids are also sampled as auxiliary channels. For each training image patch, we use the estimated depth map to interpolate for the depth-varying defocus blur simulation. The simulated image is then concatenated with auxiliary channels for network reconstruction.

1.2. Data augmentation

In our experiments, data augmentation is applied at various stages of the pipeline to enhance training robustness and improve real-world performance. The need for data augmentation arises from both image distribution variability and hardware imperfections.

Image operations. Basic image augmentations are performed on the RGB inputs to address the limited image distribution. Identical augmentation techniques, such as random rotation, flipping, scaling, and cropping, are applied to both the RGB images and the depth maps. Additionally, for RGB images, pixel-level operations including adjustments in color, saturation, and brightness are implemented.

Unprocessing. Random perturbations following a Gaussian distribution are introduced to unprocessing parameters, including the gamma value in the inverse gamma correction step and gain values in inverse white balance. This approach accounts for the unknown image signal processing (ISP) pipelines used to produce the final images, especially in online text datasets [5, 6] for OCR fine-tuning, and the varying sensor response curves of different cameras. By incorporating augmentation into the unprocessing pipeline, the unprocessed RAW signals derived from RGB images more accurately mimic real captured RAW signals.

Post-processing. Our network reconstructs blurry and noisy RAW signals, while the loss functions are computed in the post-processed RGB image space to prioritize RGB image quality. In this stage, Gaussian perturbations are applied to various ISP parameters, such as the gain values in the auto white balance step and the gamma value in gamma correction. Specifically, the center gamma value is set to 2.0 to emphasize dark regions during network training [2]. Augmenting the post-processing pipeline ensures that the final image quality generalizes better across diverse and complex ISP algorithms.

Depth scaling. The DepthAnythingV2 [8] model provides only relative depth estimations for an image, whereas our training pipeline requires absolute depth maps for depth-varying image simulation. We randomly scale relative depths ranging from 0 to 1 to absolute depths within the minimum and maximum desired ranges using various strategies: linear mapping, square mapping, exponential mapping, and constant mapping. For each mapping method, random perturbations are applied to the function parameters. This augmentation enhances the network's ability to generalize to different real-world depth scenes.

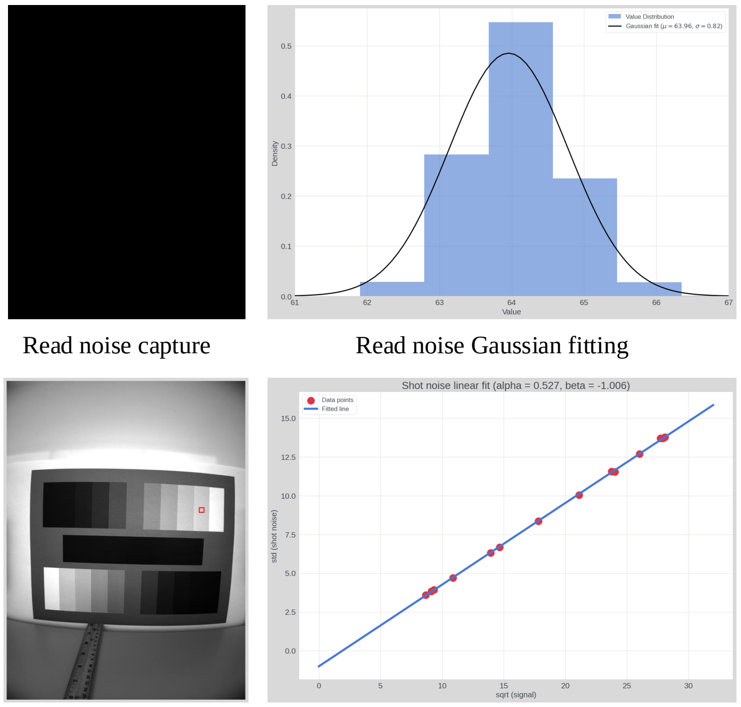
PSF perturbation. A small random Gaussian blur is applied to the Point Spread Functions (PSFs) during image simulation. This PSF augmentation accounts for potential errors in lens manufacturing and assembly, which typically result in larger PSFs compared to ideal designs.

1.3. Noise calibration

The smart glass camera used in our experiments is equipped with a Sony IMX681 sensor, which supports three analogue gains corresponding to ISO 50, 400, and 800. Other ISO values are achieved through digital gain applied to these base ISO settings. For instance, ISO 100 is obtained by digitally amplifying ISO 50 with a gain factor of 2. In our experiments, we restrict ISO values to the range of 50 to 400. We believe this range is reasonable for effectively demonstrating the utility of the ISO channel without introducing excessive complexity. At the beginning of all experiments, we calibrate the noise statistics, including both read noise and shot noise. During this calibration stage, we utilize the top-right green channel of the RAW Bayer signals, resulting in the monochromatic image shown in Fig. S2.

Read Noise. For read noise calibration, we set the ISO to 50 and cover the camera with a black mask (Fig. S2). We then continuously capture 100 RAW images. The camera sensor has a black level of 64, allowing negative noise signals to be recorded. We calculate the standard deviation of the pixel values across these 100 captures and use this value as the standard deviation of the Gaussian read noise in the image simulation stage.

Shot Noise. To calibrate shot noise, we first print a grayscale chart with regions of varying gray levels (Fig. S2). We set the ISO to 50 and adjusted the exposure time to ensure that the RAW signal values span a wide range from 64 (black level) to 1023 (the upper bound of the 10-bit sensor). After fixing the exposure time, we continuously capture 100 RAW images. In the shot noise calibration stage, we select small patches from different dark regions in the images, compute the average signal intensity, and determine the corresponding noise standard deviation. We then subtract the read noise and regress the square root of the signal intensity against the noise standard deviation, resulting in the curve depicted in Fig. S2. This regression curve aligns well with the noise model employed in our experiments.



Read noise capture

Read noise Gaussian fitting

Shot noise capture

Shot noise fitting

Figure S2. **Calibration of read and shot noise for the camera sensor.** Top: read noise is calibrated by capturing 100 dark RAW images at ISO 50, followed by the computation of the standard deviation. Bottom: shot noise is calibrated by capturing 100 RAW images of a grayscale chart with varying gray levels. A regression is performed to fit the relationship between the square root of signal intensities and the corresponding standard deviations, after subtracting the read noise. The red box in the image corresponds to a red dot in the curve.

1.4. Data capture setup

In the experiments, the camera is mounted on a tripod to eliminate motion blur, which is not the focus of this study. The camera operates in auto-exposure mode, with ISO settings determined by the internal algorithm during capture. Objects are placed at short distances ranging from 10 to 30 cm, while the background can extend far from the camera. Images are captured in both indoor and outdoor environments. Fig. S3 illustrates the setup for capturing short-distance images and performing 3D reconstructions.

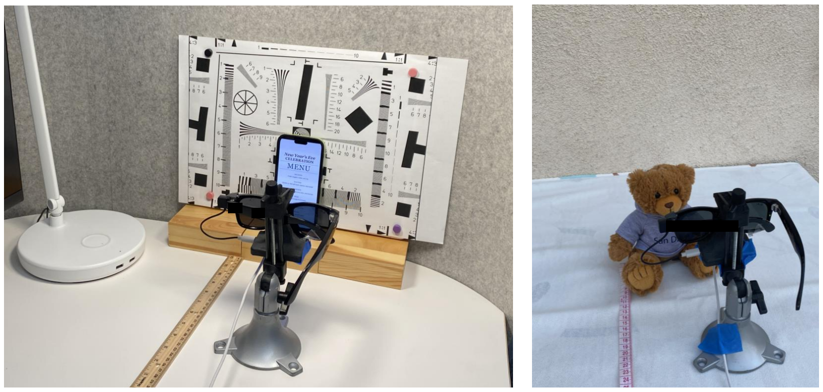


Figure S3. Experimental setup for capturing short-distance images and 3D reconstructions in both indoor and outdoor environments.

2. Extra experimental results

2.1. Extra real-world comparison examples

In this subsection, we present additional experimental results comparing our approach with existing algorithms, including the classical Polyblur method [4] and the pretrained state-of-the-art single-image defocus deblurring network LaDKNet [7]. For Polyblur, we fine-tuned hyperparameters to achieve optimal results by testing approximately five different hyperparameter sets and selecting the best outcomes. LaDKNet does not natively support 12-megapixel input images; therefore, we cropped the full-resolution images into smaller patches, processed them individually, and then recombined them to obtain the full-resolution output. The results indicate that classical algorithms fail to produce high-fidelity images, resulting in significant deblur artifacts, such as halos. Additionally, networks trained on public defocus deblur datasets [1] cannot be directly applied to images captured by different cameras due to variations in optical and noise properties. Significant noise artifacts are observable in the LaDKNet results. In contrast, the network trained with our synthetic dataset exhibits minimal halo artifacts and effectively removes noise.

We also include comparisons with different synthetic dataset methods: image simulation in RGB space; RAW space without auxiliary channels or depth-varying simulation; RAW space with auxiliary channels but without depth-varying simulation; and our method, which incorporates both auxiliary channels and depth-varying simulation. The results are reported in Fig. S4, showcasing both planar and 3D scenes in indoor and outdoor environments.

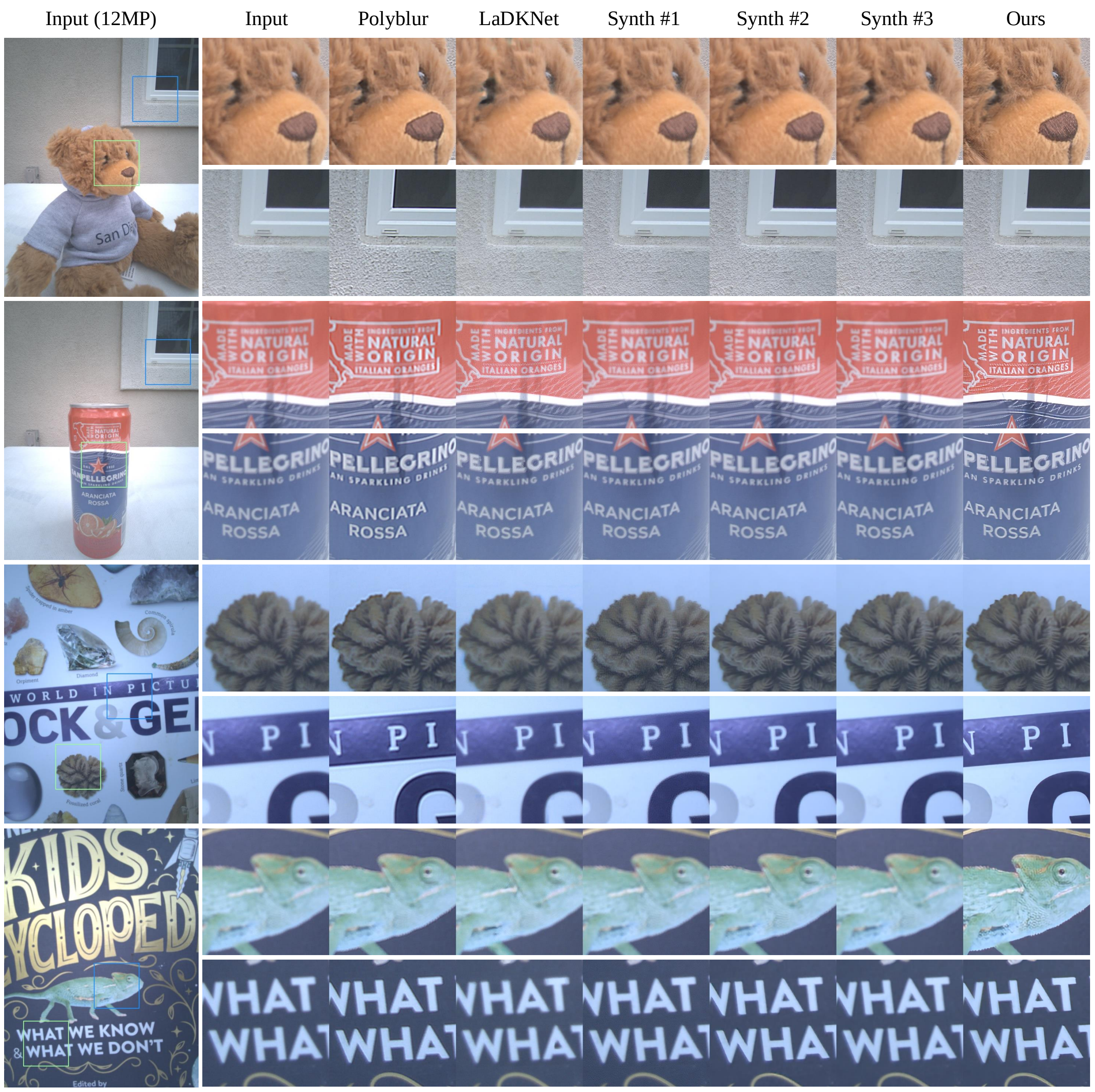


Figure S4. **Comparison of deblurring results on 12-megapixel real-world images.** From left to right: (1) input images captured by our smart glass camera, (2) deblurred using the classical Polyblur [4] algorithm, (3) deblurred with the pretrained LaDKNet [7], and (4) deblurred using NAFNet [3] trained on different synthetic datasets. Synthetic Dataset Details: Synth #1 simulates noise and defocus blur in RGB space; Synth #2 in RAW space without depth variation or auxiliary channels; Synth #3 in RAW space with auxiliary channels but without depth variation; and Ours simulates noise and defocus blur in RAW space with both auxiliary channels and depth variation.

2.2. Importance of incorporating depth-varying simulation

Experimental results presented in the main paper demonstrate that for synthetic datasets without depth-varying optical simulation, the network may become misled by sharp regions in the images, thereby failing to accurately distinguish and restore spatially varying defocus. Fig. S5 provides further examples of this failure on a synthetic validation dataset. In the first example (car), both network models are able to restore the blurry regions effectively, leading to clear image content. However, in the second and third examples (bicycle handle and sunglasses), the model trained with a constant depth map is confused by the presence of sharp background textures, resulting in an inadequate restoration of the intended blurry objects.

These results emphasize the critical importance of incorporating depth-varying simulation into the network training process. By integrating realistic depth transitions, the network becomes better equipped to handle complex scenes and subtle defocus variations inherent in real-world imaging. This leads to enhanced deblurring performance along with improved robustness against misinterpretations caused by misleading scene geometry. In summary, the inclusion of depth-varying simulation not only addresses the limitations observed with constant depth map training but also contributes to more accurate and reliable deblurring. These findings can bridge the gap between synthetic training scenarios and actual imaging conditions.

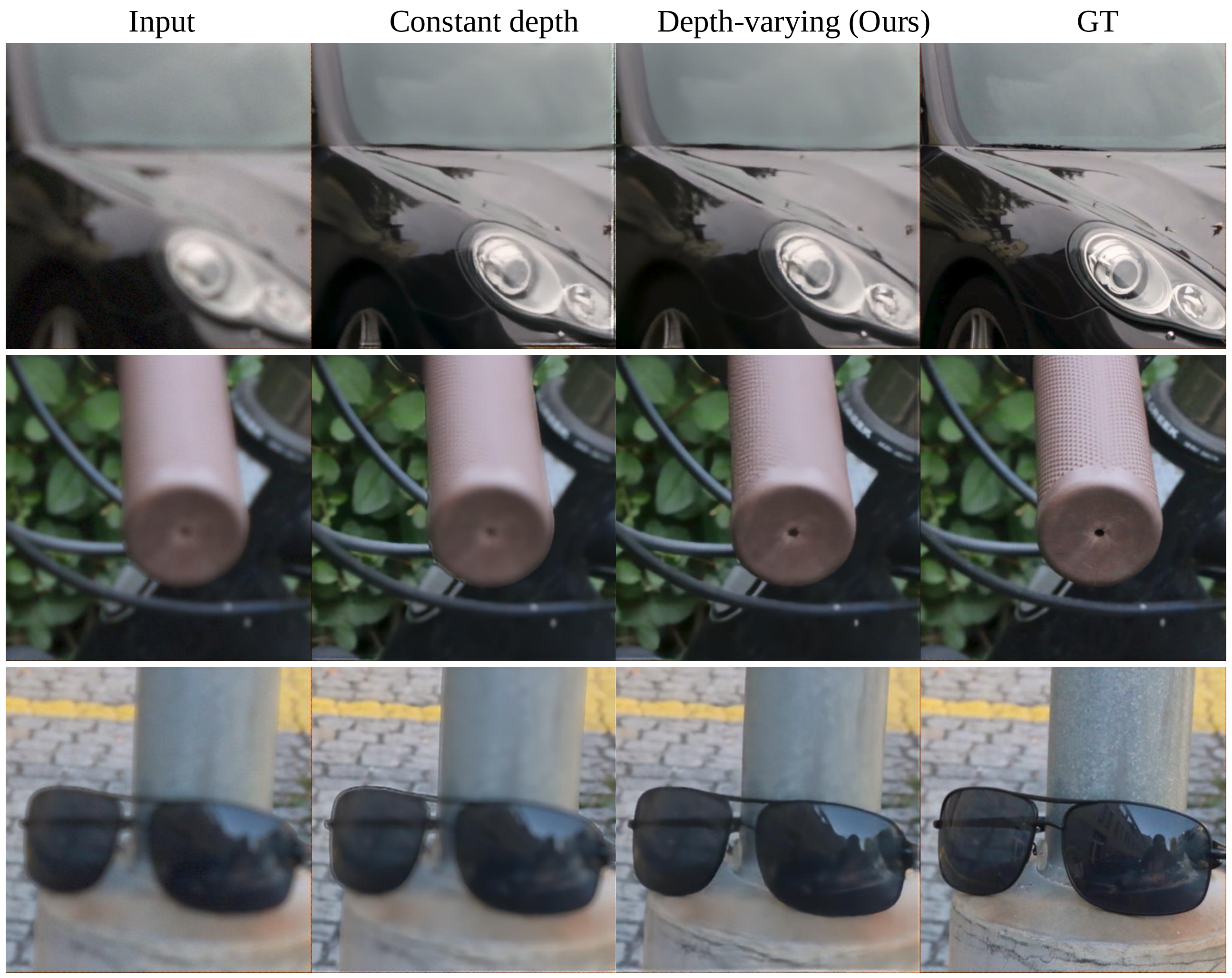


Figure S5. **Deblur results comparison between constant depth map and varying depth map.** In the first example (car), both network models successfully restored the blurry part and produced clear image content. However, for the second and third examples (bicycle handle and sunglasses), the network model trained with constant depth maps gets confused by sharp background textures and fails to restore blurry objects.

2.3. Example of auxiliary channel adjustment

Auxiliary channels play an important role in enhancing the performance of image-processing networks. They not only guide the network in handling various noise levels, spatially varying optical aberrations, and defocus deblurring but also offer flexibility in achieving desired subjective image quality. This flexibility provides image quality engineers with greater control during post-processing, allowing them to fine-tune images according to specific requirements.

Fig. S6 presents two examples of tuning ISO channels in the post-processing stage to achieve the desired image quality. Specifically, if the ISO used during capture is 100, it can be increased to 200 or reduced to 50. For example, in low-light environments, manually increasing the ISO channel value can produce smoother results, which are typically more pleasing. Conversely, for applications requiring high detail retention, manually reducing the ISO channel value can preserve more details.

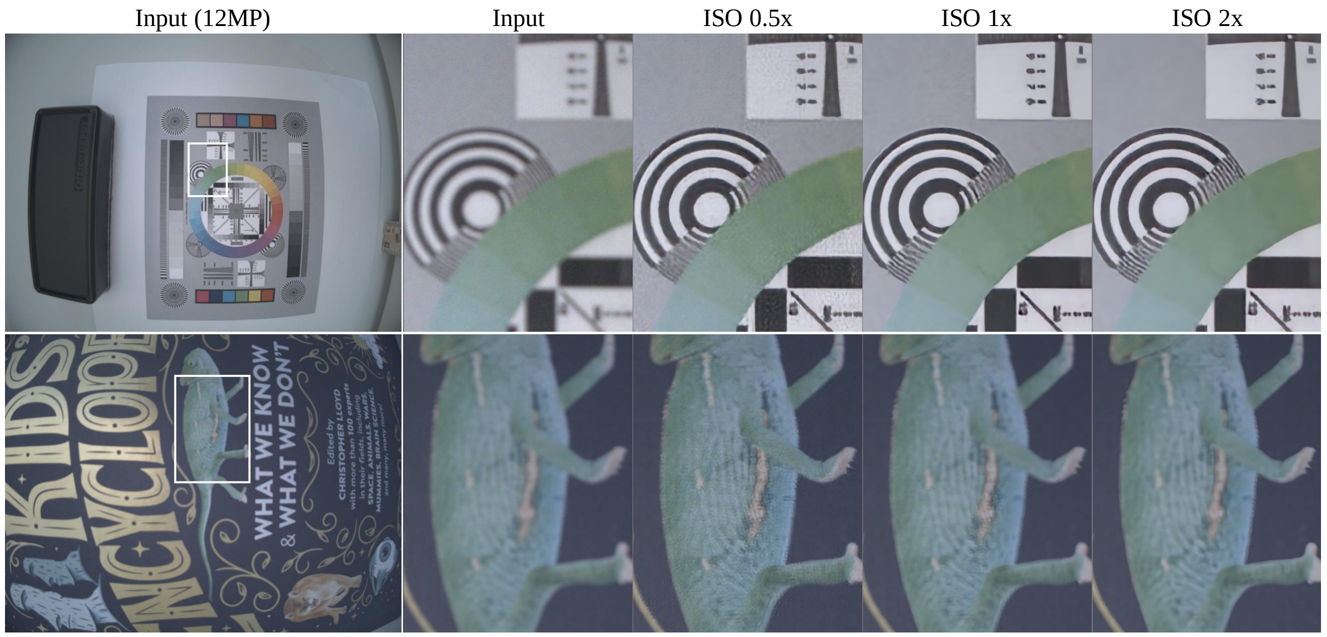


Figure S6. **Example of tuning the ISO channel in post-processing for desired image quality.** The top image demonstrates how manually increasing the ISO channel value can result in a smoother image quality, reducing noise and enhancing overall appearance. The bottom image shows that decreasing the ISO channel value preserves more details, which is beneficial for applications requiring high detail retention.

3. More discussions

3.1. Nonuniform reconstruction

In the results, we observe some nonuniformity in the experimental outcomes (Fig. S7). We hypothesize that this may be attributed to both the capture data where tiny details already get lost, as well as the training strategy, which involves randomly sampling small image patches and feeding them into the network for training.

3.2. Depth map as auxiliary channel

During inference, we do not incorporate the estimated depth map into the input. This decision is based on our observation that the network can implicitly learn to perform depth estimation and defocus detection during the training phase. Consequently, the network demonstrates the capability to handle real-world scenes with varying depths without explicit depth information. While integrating a depth map estimated by DepthAnythingV2 [8] could potentially simplify the defocus deblurring process, it would also introduce additional complexity and computational overhead during inference. This trade-off between performance enhancement and resource consumption led us to prioritize an implicit learning approach.

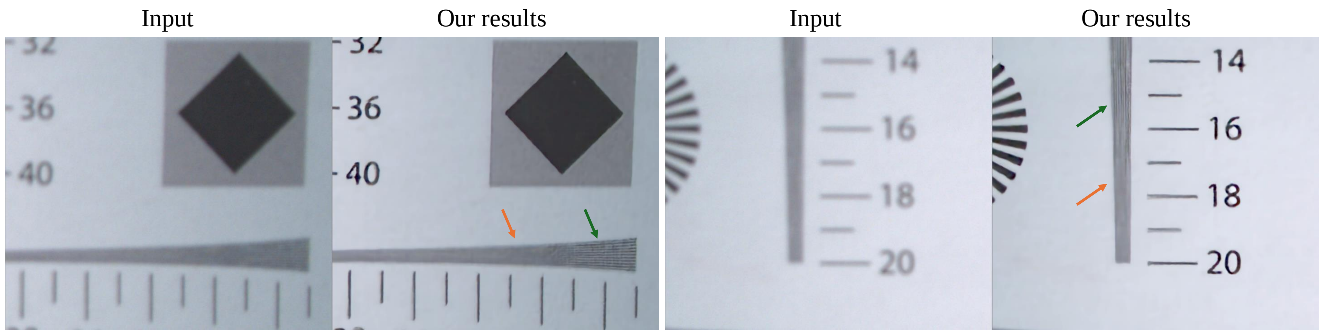


Figure S7. **Nonuniformity in deblurred results.** Our network successfully reconstructs blurry input text images, producing recognizable text. However, some nonuniformity is present in the reconstructions, where some parts are sharp (green arrow) enough but the neighbors still remains blurry (orange arrow).

References

[1] Abdullah Abuolaim and Michael S. Brown. *Defocus Deblurring Using Dual-Pixel Data*, page 111–126. Springer International Publishing, 2020. 4

[2] Chakravarty R. Alla Chaitanya, Anton S. Kaplanyan, Christoph Schied, Marco Salvi, Aaron Lefohn, Derek Nowrouzezahrai, and Timo Aila. Interactive reconstruction of monte carlo image sequences using a recurrent denoising autoencoder. *ACM Transactions on Graphics*, 36(4):1–12, 2017. 2

[3] Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. *Simple Baselines for Image Restoration*, page 17–33. Springer Nature Switzerland, 2022. 5

[4] Mauricio Delbracio, Ignacio Garcia-Dorado, Sungjoon Choi, Damien Kelly, and Peyman Milanfar. Polyblur: Removing mild blur by polynomial reblurring. *IEEE Transactions on Computational Imaging*, 7:837–848, 2021. 4, 5

[5] Shangbang Long, Siyang Qin, Dmitry Panteleev, Alessandro Bissacco, Yasuhisa Fujii, and Michalis Raptis. Towards end-to-end unified scene text detection and layout analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 2

[6] Shangbang Long, Siyang Qin, Dmitry Panteleev, Alessandro Bissacco, Yasuhisa Fujii, and Michalis Raptis. Icdar 2023 competition on hierarchical text detection and recognition. *arXiv preprint arXiv:2305.09750*, 2023. 2

[7] Lingyan Ruan, Mojtaba Bemana, Hans-peter Seidel, Karol Myszkowski, and Bin Chen. Revisiting image deblurring with an efficient convnet. *arXiv preprint arXiv:2302.02234*, 2023. 4, 5

[8] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *arXiv preprint arXiv:2406.09414*, 2024. 2, 7

[9] Zemax, Inc. *Zemax OpticStudio® - Design and Analysis Software for Optical Systems*. Zemax, Inc., Fremont, CA, USA, version 23.0 edition, 2023. 2