Efficient Multi-Lens Bokeh Effect Rendering and Transformation

Tim Seizinger[1*], Marcos V. Conde[1*], Manuel Kolmet[2], Tom E. Bishop[2], Radu Timofte[1]

[1] Computer Vision Lab, CAIDAS & IFI, University of Würzburg, Germany
[2] Glass Imaging Inc., CA

Figure 1. Samples using the Bokeh Effect Transformation Dataset (BETD) [7]. (Up) Rendering Bokeh. (Bot.) Transforming Bokeh in real captures from a Canon 50mm lens *f/1.4* to Sony 50mm lens *f/1.8*. The method respects the foreground and provides real Bokeh aesthetics.

Abstract

Many advancements of mobile cameras aim to reach the visual quality of professional DSLR cameras. Great progress was shown over the last years in optimizing the sharp regions of an image and in creating virtual portrait effects with artificially blurred backgrounds. Bokeh is the aesthetic quality of the blur in out-of-focus areas of an image. This is a popular technique among professional photographers, and for this reason, a new goal in computational photography is to optimize the Bokeh effect itself.

This paper introduces EBokehNet, a efficient state-of-the-art solution for Bokeh effect transformation and rendering. Our method can render Bokeh from an all-in-focus image, or transform the Bokeh of one lens to the effect of another lens without harming the sharp foreground regions in the image. Moreover we can control the shape and strength of the effect by feeding the lens properties i.e. type (Sony or Canon) and aperture, into the neural network as an additional input. Our method is a winning solution at the NTIRE 2023 Lens-to-Lens Bokeh Effect Transformation Challenge, and state-of-the-art at the EBB benchmark.

1. Introduction

Computational photography research and recent advancements of mobile cameras aim to reach the visual quality of full-frame DSLR cameras [8, 15]. One of the most popular effects in photography is Bokeh, which refers to the way the lens renders the out-of-focus blur in a photograph (Fig. 1) [14, 28]. Professional photographers can produce different Bokeh styles by using different lens designs and configurations. This effect is controlled by the optical design of a lens, its aperture setting, the distance to the subject, and the focal length of the lens. However, due to the limited optics of mobile cameras, these cannot produce re-

* Corresponding first authors. marcos.conde@uni-wuerzburg.de
Code & models: https://github.com/mv-lab/AISP

alistic Bokeh naturally. In this case, the effect is added as a post-processing; this is the main focus and application of most algorithms for Bokeh rendering.

Note that Bokeh and depth-of-field (DoF) are two different, yet related techniques in photography. DoF refers to the sharp areas of focus, while Bokeh is the artistic quality of out-of-focus areas. These are related, and often used interchangeably since Bokeh rendering can be seen as transforming wide to shallow depth-of-field images [14].

Classical approaches [3, 25, 36, 39, 46] can render and change Bokeh styles easily by controlling the shape and size of the blur kernel, which is usually an estimated point spread function (PSF). However, these methods often suffer from unpleasant artifacts such as chromatic aberration and depth discontinuities.

Deep learning-based methods [14, 28, 31, 37, 38] represent the *state-of-the-art* for this task, but they have difficulty to simulate real Bokeh styles *e.g*. bokeh balls, and only produce the effect present in the training data. Moreover, these methods cannot adjust different styles, and cannot apply large blur kernels to high-resolution (HR) images, as they are limited by the fixed receptive field of the neural network. This is an important point that limits their potential application in real scenarios.

A common approach to render Bokeh consists in segmenting out the foreground (*e.g*. person, face, or object of interest) in the image, and then processing the background [17, 34, 35, 47] independently. A similar approach is to blur the image based on a depth map [13, 28]. We can also find end-to-end deep learning solutions [14, 17, 33] capable of transforming wide to shallow depth-of-field images automatically, without using depth or segmentation maps.

Despite the active research in this topic, rendering photorealistic Bokeh is still a challenging task. Recently the new **Bokeh Transformation** task was introduced [7]. This task is defined as follows: for a given input image A (all-in-focus, out-of-focus or in-between) with known lens-type and aperture setting, knowing the target lens type and setting, we aim to produce or transform the corresponding effect B while preserving the foreground intact.

In this work, we present an efficient neural network capable of rendering or converting the Bokeh effect of one lens to the effect of another lens without harming the sharp foreground regions in the image. The proposed EBokehNet model achieves *state-of-the-art* results at the Bokeh Effect Transformation Dataset (BETD) [7] and the Everything is Better with Bokeh! (EBB) benchmark [14, 16].

2. Related Work

Classical Bokeh rendering methods require a single image and 3D information. The most practical ones use depth maps [2, 11, 25, 39, 42], while more advanced classical rendering requires the complete 3D scene information [28].

Early works such as Bertalmío *et al*. [3] use a point-spread function (PSF) to simulate realistic bokeh. Yang *et al*. [42] use simple ray tracing to render the effect.

Due to its complexity, it is common to split this task into: depth estimation [13], semantic segmentation [5], and classical rendering [25, 29, 34–36, 46]. This task decomposition also implies decoupling the image into foreground and background, and execute rendering from back to front.

These modular approaches are flexible, however, they might struggle at depth discontinuities. Furthermore their overall performance depends highly on the individual performance of each module *e.g*. the quality of the estimated depth maps, the quality of the background-foreground segmentation, the power of the classical rendering.

During the recent years we can observe a trend towards using deep learning to simulate the rendering process as an end-to-end operation. Early works such as Nalbach *et al*. [27] and Xiao *et al*. [40] train convolutional neural networks (CNN) to produce a bokeh effect from an all-in-focus image and its accurate depth map. Wang *et al*. [37] proposes an automatic rendering system comprised of depth estimation, lens blur, and guided upsampling to generate high-resolution depth-of-field (DoF) images from a single image. Most recently, Peng *et al*. proposes BokehMe [28], a framework that combines neural and classical rendering techniques and achieves *state-of-the-art* results.

Other deep learning-based methods [14, 17, 21, 23, 31, 33, 38] do not require any prior information such as depth maps, which are not easy to capture in real-world scenes. These methods usually follow a encoder-decoder architecture [32], and map the all-in-focus input images into shallow DoF images in an end-to-end manner.

Despite the promising results, these deep learning-based rendering methods lack controllability. The trained neural network can produce only the style of the effect present in the training data, and the blur range is limited by the kernels' receptive field.

We aim to improve mobile photography, therefore it is also important to address the method complexity considering the computational limitations of mobile devices. Ignatov *et al*. [14, 16, 17] studied efficient Bokeh rendering for mobile devices, being able to deploy the models on different target platforms [17]. These challenges use the popular large-scale *Everything is Better with Bokeh!* (EBB!) dataset [14] containing more than 10 thousand images collected in the wild.

By controlling the aperture size of the lens, pairs of images with wide (aperture *f/16*) and shallow (aperture *f/1.8*) depth-of-field were taken, resulting in a normal sharp photo and one exhibiting a strong Bokeh effect.

In this work we propose a novel neural rendering method able to control the effect by feeding the lens properties into the neural network as an additional input.

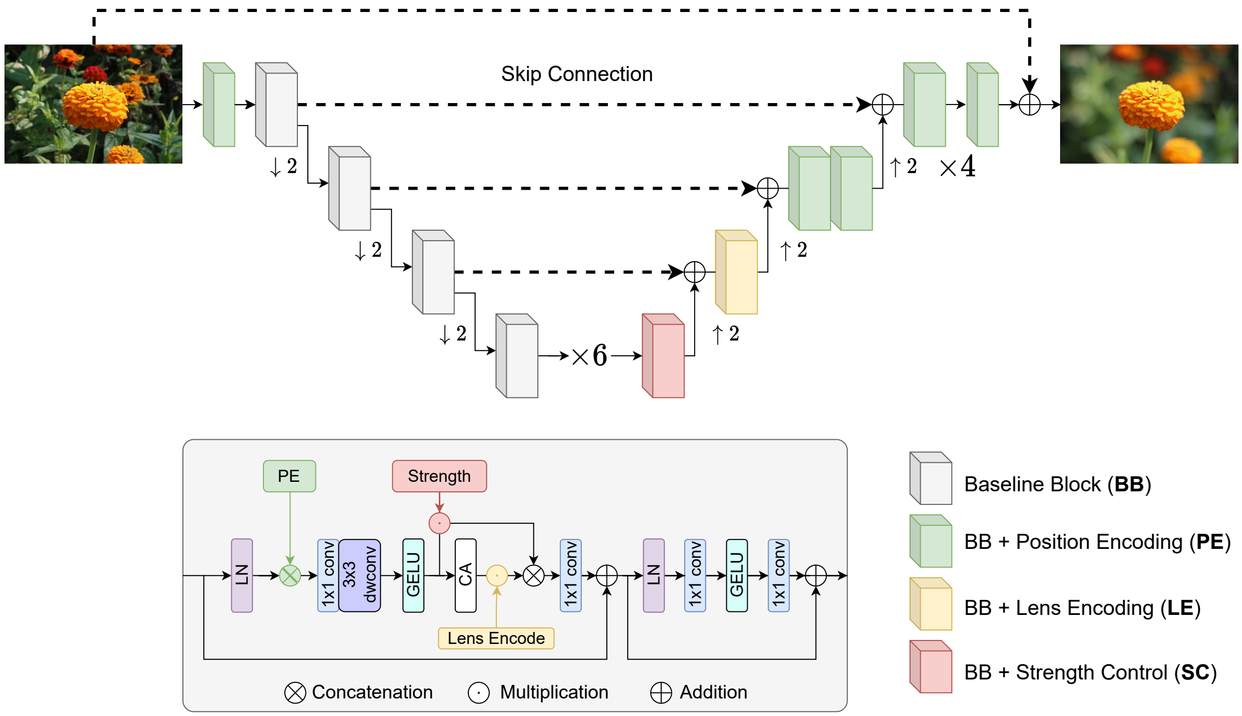

Figure 2. Architecture of the proposed EBokehNet. We use a encoder-decoder structure inspired in NAFNet [4]. We propose a new modified `Baseline Block`. We inject the lens information (type and strength) at different stages -as shown in the colour legend-, by doing this we are able to control the shape and strength of the Bokeh effect by "conditioning" the network's features. Additionally, we employ 2D positional encoding (PE) in some blocks [24] to provide extra spatial context. All the indicated operations are channel-wise.

3. Efficient Bokeh Rendering

We design our network EBokehNet for Bokeh effect rendering and transformation considering the following desired features: (i) the network should allow to control the strength and style of the Bokeh effect. This is fundamental to tackle the novel Bokeh Effect Transformation task. (ii) The model must be efficient in order to be usable, to achieve this we adopt the already efficient NAFNet architecture [4] and simplify it further by reducing the number of encoder-decoder blocks. (iii) The model should be able to render or convert the Bokeh effect of one lens to the effect of another lens without modifying the sharp foreground regions in the image. This ultimately implies a SOTA performance.

Model Design We illustrate the architecture of EBokehNet in Fig. 2. We follow a classical U-Net [32] encoder-decoder structure inspired in NAFNet [4], yet reducing notably the number of blocks.

We propose a new modified `Baseline Block` that incorporates LayerNorm [1], GELU activations [12], pixel-wise convolutions, inverted residual blocks [8], and additional residual connections. Following [4] we use two core elements that ensure efficiency and performance: (i) downsampling using strided convolutions, (ii) upsampling using pixelshuffle [44].

The core modifications allow to integrate the encoded lens information, and thus we can condition the deep features and the overall model behaviour.

We inject the lens information (type and bokeh strength) at different stages of the decoder, by doing this we are able to control the style and strength of the Bokeh effect by "conditioning" the network's features. This idea was successfully applied for flexible compression removal [18, 41].

We calculate the Bokeh *strength factor* as follows:

$$L_{bokeh} = \frac{1}{{F_{tgt}}^2} - \frac{1}{{F_{src}}^2}$$

$$BF = \frac{L_{bokeh} * disparity}{100} * 2$$

$$BF_{norm} = \frac{BF + 1}{2}$$

considering the aperture of the source F_{src} and target F_{tgt} lens, and the disparity. This is injected at the 1st encoding block as shown in Fig. 2. The *disparity* value indicates the relative distance of the foreground to the background, and serves as another indicator of the "amount of blur". This information is provided for each sample in the BETD dataset [7], in the cases were this information is not

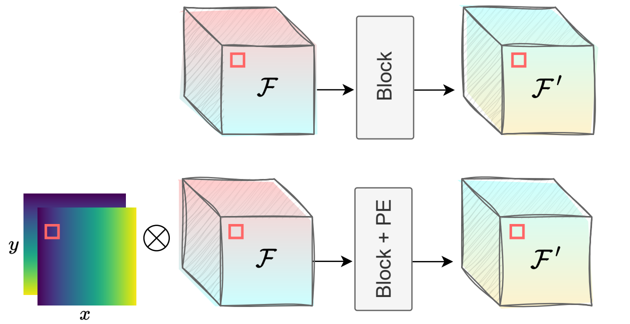

Figure 3. Our integration of positional encoding (PE). We show (up) the classical feature $\mathcal{F}$ processing [4], and (bot.) our block with PE for additional spatial context.

available *e.g.* EBB dataset [14], our model allows to disable such inputs and conditions (Fig. 2).

Additionally, similar to CoordConv [24], we also employ 2D positional encoding (PE) in some blocks, this provides extra spatial context and guide the network in rendering optical vignetting and cats-eye bokeh in the corners. We use a 2-channel coordinates map as illustrated in Fig. 3. Since the real and synthetic Bokeh effect from BETD [7] is associated with **space-varying PSFs**, we found PE a very powerful method to enhance features.

Note that we use five blocks with PE at the final decoding stage using high-resolution features (see Fig. 2), we found this especially important to improve performance notably.

4. Experimental Resuls

We evaluate our models using the novel Bokeh Effect Transformation Dataset (BETD) [7], and the *"Everything is Better with Bokeh!"* (EBB) benchmark [14, 16].

BETD The dataset [7] contains 20000 and 200 image pairs for the training and test sets, respectively. The average resolution of the images is 1584×1056. The training set consists on synthetic images generated using an estimated PSF of professional lenses. The test set contains 100 synthetic images, and 100 real captures. Both the simulated and real images are based on *Sony Alpha 7R II* and *Sony Alpha 7R IV* professional cameras with a Sony 50mm lens set to *f/1.8* and *f/16* apertures and a Canon EF 50mm lens set to *f/1.8* and *f/1.4* apertures. For each real or synthetic pair, we have the corresponding metadata for the source and target images *e.g.* `Sony50mmf1.8BS` $\rightarrow$ `Canon50mmf1.4BS`. We also use the provided disparity value that indicates the "amount" of blur as the "bokeh strength".

EBB used in [14, 16, 17] is a large-scale *EBB!* dataset containing 5K shallow / wide depth-of-field image pairs collected in the wild with the Canon 7D DSLR camera and 50mm *f/1.8* fast lens.

4.1. Bokeh Effect Transformation

We provide the results on the BETD [7] benchmark in Tab. 1 and qualitative samples in Fig. 5 and Fig. 6.

Our method EBokehNet achieves *state-of-the-art* performance while being $50\times$ smaller in comparison to the others [20, 43]. Even our smaller version EBokehNet-s with less blocks and depth, achieves very competitive results. We believe this is because: (i) novel block with positional encoding, (ii) the new design of our baseline block based on NAFNet [4], and (iii) the efficient integration of lens information into the model.

We define our small version EBokehNet-s as the smartphone model. Following [14, 16] we design this shallow variant for mobile devices. This has blocks with 16 channels instead of 32, and less number of blocks in the middle and decoding part. Since the model is extremely compact (1 Million parameters) we can train using full-resolution images, which we found is an advantage.

In Fig. 4 we show the performance comparison of the two model variants when we vary the bokeh strength *e.g.* render strong Bokeh *f/16* $\rightarrow$ *f/1.8*, transform it *f/1.8* $\rightarrow$ *f/1.4*, or recover sharp regions by "removing" Bokeh *f/1.8* $\rightarrow$ *f/16*.

Both models allow high-resolution image processing without patching or tiling strategies. Also note that we do not employ the provided alpha masks or any sort of segmentation, the foreground and background separation is completely implicitly learned.

Results per transformation *Removal.* Is a transformation from shallow to wide DoF *e.g.* *f/16* $\rightarrow$ *f/1.8*. This is the most challenging sub-task since it is similar to deblurring [8], an ill-posed problem. While the perceived increase in background busyness and detail is captured, the details do not necessarily match the ground-truth and the perceived style of the target lens is rarely captured.

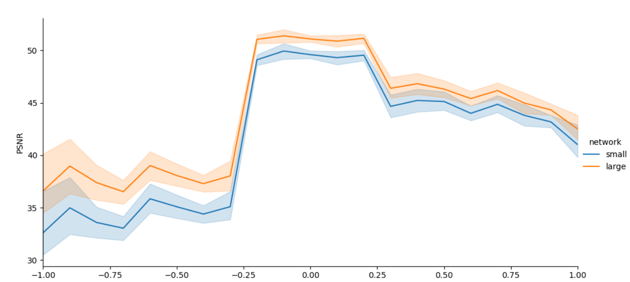

Figure 4. Performance comparison of the small and large variants of EBokehNet on BETD [7] depending on the strength of the bokeh transformation. Negative factor means Bokeh is removed (shallow to wide DoF), around 0 it is transformed *e.g.* `Sony50mmf1.4` $\rightarrow$ `Canon50mmf1.4`, and for positive factors bokeh is rendered by the network.

Method	# Params. (M)	Synthetic + Real			Real		Foreground/Background	
		PSNR ↑	SSIM ↑	LPIPS ↓	SSIM ↑	LPIPS ↓	PSNR_F ↑	SSIM_B ↑
NAFBET [20]	115	35.264	**0.9362**	0.0985	0.8416	0.2186	47.512	0.9553
SBTNet [7]	265	34.572	0.9361	**0.0966**	0.8435	0.2224	47.889	0.9559
CBTNet [7]	182	32.326	0.9333	0.1076	0.8420	0.2230	46.875	0.9500
BokehOrNot [43]	21	32.288	0.9327	0.1130	0.8423	0.2199	48.280	0.9488
SGLMS [7]	7	32.076	0.9324	0.1076	0.8419	0.2161	47.024	0.9484
IR-SDE [26]	78	30.866	0.9297	0.1301	0.8427	0.2387	44.905	0.9418
DoubleGAN [17]	5	27.970	0.9213	0.1542	0.8455	0.2175	41.522	0.9312
Synthetic	-	28.599	0.9128	0.2181	-	-	48.163	0.9132
EBokehNet-s	**1**	34.543	0.9350	0.1039	0.8414	0.2206	47.220	0.9530
EBokehNet	20	**35.521**	**0.9362**	0.0993	0.8412	0.2208	47.577	0.9557

Table 1. NTIRE 2023 Lens-to-Lens Bokeh Effect Transformation (**BETD**) [7] results. The methods are ranked by PSNR/SSIM. The models were tested on unseen real captures and synthetic rendered content. We also provide the rounded number of parameters of each method. Synthetic indicates the metrics for the raw source image. Our method EBokehNet achieves *state-of-the-art* performance while being extremely smaller in comparison to others. Moreover we can appreciate that the method is not harming the sharp foreground regions.

Sony 50mm lens *f1.4* | Canon 50mm lens *f1.4*

Canon 50mm lens *f1.4* | Sony 50mm lens *f1.4*

Figure 5. **Real captures** from the BETD [7]. These images were captured using the same DSLR camera. The proposed *EBokehNet* is able to do a bidirectional conversion between both setups Sony ⟷ Canon. Images courtesy of Glass Imaging, Inc.

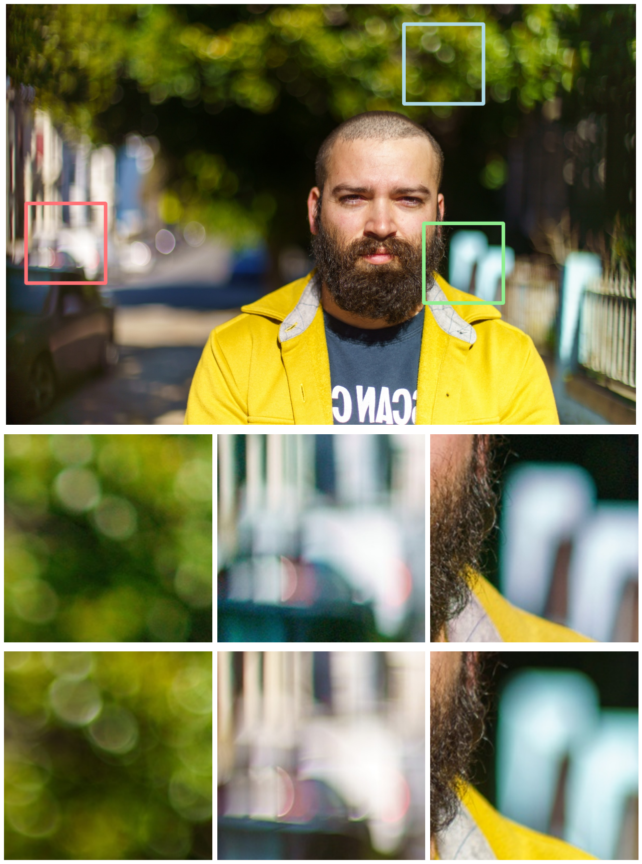

Figure 6. Qualitative samples from BETD [7] of realistic Bokeh transformation using the proposed network. The 1st row of crops corresponds to the setting `Sony50mmf1.4`. The 2nd corresponds to the transformed effect towards the setting `Canon50mmf1.4` with our model. Image courtesy of Glass Imaging.

Transformation. The method achieves great performance, the bokeh style is very well adjusted to the target lens. However, we found that the method struggles to generalize to real images, which were not present in the training data and have different properties.

Rendering. The method can emulate the bokeh style and strength, and the results match the target lens settings. There is a slight degradation in performance when the bokeh strength is too different from the source (see Fig. 4).

Real vs Synthetic Data We found a clear gap generalization gap between the synthetic and real content. Despite the synthetic content was rendered using accurate estimated PSFs, the properties of real images differ notably *e.g.* illumination, distance to the objective (implicit depth). We provide extensive qualitative results in Fig. 8 and Fig. 5.

Method	PSNR ↑	SSIM ↑	LPIPS [45] ↓
EBokehNet	**24.99**	0.852	0.1912
SKN [22]	24.66	0.8521	0.3323
DBSI [9]	23.45	0.8675	0.2463
DMSHN [10]	24.72	0.8793	0.2271
DDDF [30]	24.14	0.8713	0.2482
BGGAN [31]	24.39	0.8645	0.2467
BRADCN [23]	24.83	0.8737	**0.1448**
PyNet [14]	24.93	0.8788	0.2219
BEViT [38]	24.57	0.8880	0.1985

Table 2. Quantitative results on the **EBB** [14, 16] **Val294** testset. Some numbers are borrowed from [23, 38].

4.2. Bokeh Effect Rendering

First, we evaluate our model (pre-trained on the BETD [7] dataset and task) on the EBB dataset [14] in a zero-shot manner. We found that the model only works for a few images, the reason is because the training data in BETD [7] is clearly separable into foreground and background. Also the foreground in BETD is always a person (or a face), while at EBB we find a wide variety of objects.

Therefore, we fine-tune the model on the EBB dataset [14]. We notice that the model quickly learns how to separate foreground and background in real images, and can emulate strong natural Bokeh.

As we show in Tab. 2, our model achieves *state-of-the-art* results. In comparison to the baseline PyNET [14, 16] with 40M parameters, our method is $40\times$ smaller. Moreover, we achieve better results than other complex methods that apply depth estimation or foreground segmentation, our approach is purely end-to-end with one efficient network. We can conclude that our method is easily transferable to real images with a wide variety of focused objects *e.g.* faces, persons, cars, animals, plants, etc. Pre-training on simple synthetic data -with a realistic blur model- improves the SOTA for Bokeh rendering on EBB [14].

4.3. Implementation Details

We train all the models using Adam optimizer [19] with Cosine Annealing learning rate scheduler with 10 epochs of linear warmup using a maximum learning rate of 1e-3 and minimum learning rate of 5e-5. We train the models to convergence, for the small model 220 epochs, and 280 for the large version. We use simple $\mathcal{L}_1$ loss.

Since the small model is quite efficient, we can high resolution crops of 1024x1024 on images for training, meanwhile we use 512×512 crops for the larger version. We apply standard augmentations consisting in flips and rotations. We set the mini-batch size to 14, and run a distributed training over 7 RTX 3090Ti GPUs via DDP. The complete train-

Figure 7. Rendering Bokeh on **real-wold** images from the EBB dataset [14]. We show the rendered Bokeh effect from our EBokehNet in a large variety of scenes in-the-wild. We provide additional qualitative results and comparisons in our project site.

ing time is approximately 48hrs per model. Furthermore, to increase the inference performance on the full-resolution images we employ a Test Time Local Converter [6].

Fine-tuning on EBB. We use the aforementioned experimental setup with the following modifications. We start with a learning rate 5e-4. We keep the same architecture and disable the lens and Bokeh strength encoding, therefore we use only baseline blocks (see Fig. 2). We just need to train 50 epochs to achieve SOTA results. Since the images are not perfectly aligned, we train the model using a combination of fidelity and perceptual losses as follows:

$$\mathcal{L}_{EBB} = 0.5 \times \mathcal{L}_1 + 0.05 \times \mathcal{L}_{SSIM} + 0.1 \times \mathcal{L}_{VGG} \quad (1)$$

5. Conclusions

We introduce EBokehNet, a efficient state-of-the-art solution for Bokeh effect transformation and rendering. Our method can render Bokeh from an all-in-focus image, or transform the Bokeh of one lens to the effect of another lens without harming the sharp foreground regions in the image. Moreover, even being an end-to-end network, we can control the shape and strength of the effect by feeding the lens properties into the neural network as an additional input.

We prove the benefits of our method in the novel Bokeh Effect Transformation Dataset, and the real scenes from EBB, achieving state-of-the-art results in both benchmarks.

As future work we will study closer the gap between synthetic and real captures. We also aim to reduce further the complexity of the network and test it on real mobile devices.

Acknowledgements This work was partially supported by the Humboldt Foundation. Glass Imaging thanks Griffin Kettler and his models for assistance with the real images captured for the test portion of the BETD dataset.

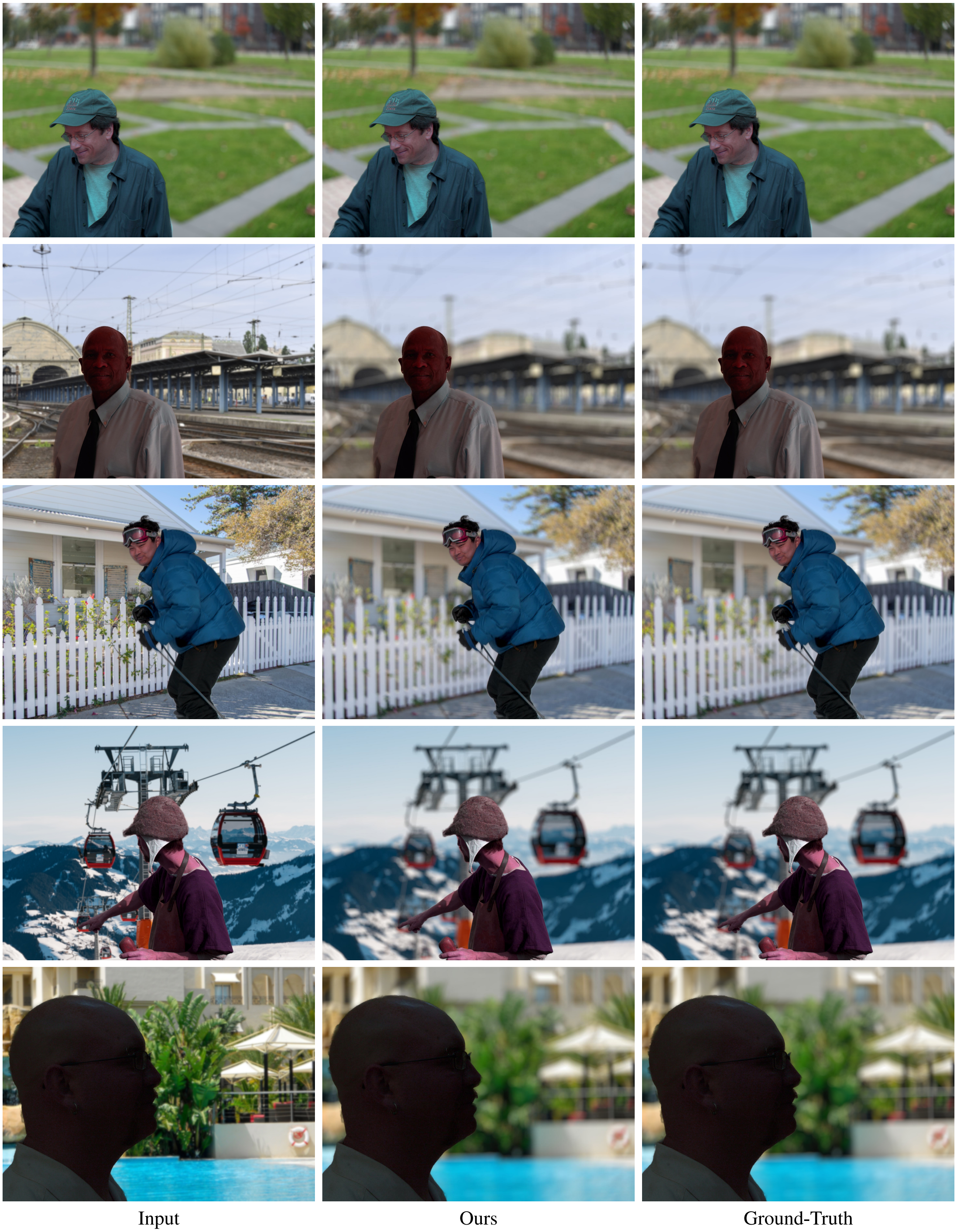

Figure 8. *Qualitative results* on the Bokeh Effect Transformation Dataset (**BETD**) [7] testset.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 3

[2] 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*, pages 4466–4474, 2015. 2

[3] Marcelo Bertalmio, Pere Fort, and Daniel Sanchez-Crespo. Real-time, accurate depth of field using anisotropic diffusion and programmable graphics cards. In *Proceedings. 2nd International Symposium on 3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004.*, pages 767–773. IEEE, 2004. 2

[4] Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. Simple baselines for image restoration. *arXiv preprint arXiv:2204.04676*, 2022. 3, 4

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017. 2

[6] Xiaojie Chu, Liangyu Chen, Chengpeng Chen, and Xin Lu. Improving image restoration by revisiting global information aggregation. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part VII*, pages 53–71. Springer, 2022. 7

[7] Marcos V Conde, Manuel Kolmet, Tim Seizinger, Thomas E. Bishop, Radu Timofte, et al. Lens-to-lens bokeh effect transformation. NTIRE 2023 challenge report. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 1, 2, 3, 4, 5, 6, 8

[8] Marcos V Conde, Florin Vasluianu, Javier Vazquez-Corral, and Radu Timofte. Perceptual image enhancement for smartphone real-time applications. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1848–1858, 2023. 1, 3, 4

[9] Saikat Dutta. Depth-aware blending of smoothed images for bokeh effect generation. *Journal of Visual Communication and Image Representation*, 77:103089, 2021. 6

[10] Saikat Dutta, Sourya Dipta Das, Nisarg A Shah, and Anil Kumar Tiwari. Stacked deep multi-scale hierarchical network for fast bokeh effect rendering from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2398–2407, 2021. 6

[11] Thomas Hach, Johannes Steurer, Arvind Amruth, and Artur Pappenheim. Cinematic bokeh rendering for real scenes. In *Proceedings of the 12th European Conference on Visual Media Production*, pages 1–10, 2015. 2

[12] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 3

[13] Andrey Ignatov, Grigory Malivenko, Radu Timofte, Lukasz Treszczotko, Xin Chang, Piotr Ksiazek, Michal Lopuszynski, Maciej Pioro, Rafal Rudnicki, Maciej Smyl, et al. Efficient single-image depth estimation on mobile devices, mobile ai & aim 2022 challenge: report. In *Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part III*, pages 71–91. Springer, 2023. 2

[14] Andrey Ignatov, Jagruti Patel, and Radu Timofte. Rendering natural camera bokeh effect with deep learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 418–419, 2020. 1, 2, 4, 6, 7

[15] Andrey Ignatov, Radu Timofte, Shuai Liu, Chaoyu Feng, Furui Bai, Xiaotao Wang, Lei Lei, Ziyao Yi, Yan Xiang, Zibin Liu, et al. Learned smartphone isp on mobile gpus with deep learning, mobile ai & aim 2022 challenge: report. In *Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part III*, pages 44–70. Springer, 2023. 1

[16] Andrey Ignatov, Radu Timofte, Ming Qian, Congyu Qiao, Jiamin Lin, Zhenyu Guo, Chenghua Li, Cong Leng, Jian Cheng, Juewen Peng, et al. Aim 2020 challenge on rendering realistic bokeh. In *Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 213–228. Springer, 2020. 2, 4, 6

[17] Andrey Ignatov, Radu Timofte, Jin Zhang, Feng Zhang, Gaocheng Yu, Zhe Ma, Hongbin Wang, Minsu Kwon, Haotian Qian, Wentao Tong, et al. Realistic bokeh effect rendering on mobile gpus, mobile ai & aim 2022 challenge: report. In *Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part III*, pages 153–173. Springer, 2023. 2, 4, 5

[18] Jiaxi Jiang, Kai Zhang, and Radu Timofte. Towards flexible blind jpeg artifacts removal. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4997–5006, 2021. 3

[19] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[20] Xiangyu Kong, Fan Wang, Dafeng Zhang, Jinlong Wu, and Zikun Liu. Nafbet: Bokeh effect transformation with parameter analysis block based on nafnet. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 4, 5

[21] Brian Lee, Fei Lei, Huaijin Chen, and Alexis Baudron. Bokeh-loss gan: multi-stage adversarial training for realistic edge-aware bokeh. In *Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part II*, pages 619–634. Springer, 2023. 2

[22] Xiang Li, Wenhai Wang, Xiaolin Hu, and Jian Yang. Selective kernel networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 510–519, 2019. 6

[23] Lu Liu, Lei Zhou, and Yuhan Dong. Bokeh rendering based on adaptive depth calibration network. *arXiv preprint arXiv:2302.10808*, 2023. 2, 6

[24] Rosanne Liu, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution. *Advances in neural information processing systems*, 31, 2018. 3, 4

[25] Xianrui Luo, Juewen Peng, Ke Xian, Zijin Wu, and Zhiguo Cao. Bokeh rendering from defocus estimation. In *Computer*

Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16, pages 245–261. Springer, 2020. 2

[26] Ziwei Luo, Fredrik K Gustafsson, Zheng Zhao, Jens Sjölund, and Thomas B Schön. Refusion: Enabling large-size realistic image restoration with latent-space diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 5

[27] Oliver Nalbach, Elena Arabadzhiyska, Dushyant Mehta, H-P Seidel, and Tobias Ritschel. Deep shading: convolutional neural networks for screen space shading. In *Computer graphics forum*, volume 36, pages 65–78. Wiley Online Library, 2017. 2

[28] Juewen Peng, Zhiguo Cao, Xianrui Luo, Hao Lu, Ke Xian, and Jianming Zhang. Bokehme: When neural rendering meets classical rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16283–16292, 2022. 1, 2

[29] Juewen Peng, Xianrui Luo, Ke Xian, and Zhiguo Cao. Interactive portrait bokeh rendering system. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 2923–2927. IEEE, 2021. 2

[30] Kuldeep Purohit, Maitreya Suin, Praveen Kandula, and Rajagopalan Ambasamudram. Depth-guided dense dynamic filtering network for bokeh effect rendering. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 3417–3426. IEEE, 2019. 6

[31] Ming Qian, Congyu Qiao, Jiamin Lin, Zhenyu Guo, Chenghua Li, Cong Leng, and Jian Cheng. Bggan: Bokeh-glass generative adversarial network for rendering realistic bokeh. In *Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 229–244. Springer, 2020. 2, 6

[32] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015. 2, 3

[33] Tim Seizinger, Marcos V Conde, Manuel Kolmet, Tom E Bishop, and Radu Timofte. Efficient multi-lens bokeh effect rendering and transformation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 2

[34] Xiaoyong Shen, Aaron Hertzmann, Jiaya Jia, Sylvain Paris, Brian Price, Eli Shechtman, and Ian Sachs. Automatic portrait segmentation for image stylization. In *Computer Graphics Forum*, volume 35, pages 93–102. Wiley Online Library, 2016. 2

[35] Xiaoyong Shen, Xin Tao, Hongyun Gao, Chao Zhou, and Jiaya Jia. Deep automatic portrait matting. In *European conference on computer vision*, pages 92–107. Springer, 2016. 2

[36] 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

[37] 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

[38] Zhifeng Wang and Aiwen Jiang. A dense prediction vit network for single image bokeh rendering. In *Pattern Recognition and Computer Vision: 5th Chinese Conference, PRCV 2022, Shenzhen, China, November 4–7, 2022, 2022, Proceedings, Part IV*, pages 213–222. Springer, 2022. 2, 6

[39] Jiaze Wu, Changwen Zheng, Xiaohui Hu, and Fanjiang Xu. Rendering realistic spectral bokeh due to lens stops and aberrations. *The Visual Computer*, 29:41–52, 2013. 2

[40] Lei Xiao, Anton Kaplanyan, Alexander Fix, Matt Chapman, and Douglas Lanman. Deepfocus: Learned image synthesis for computational display. In *ACM SIGGRAPH 2018 Talks*, pages 1–2. 2018. 2

[41] Ren Yang, Radu Timofte, Xin Li, Qi Zhang, Lin Zhang, Fanglong Liu, Dongliang He, Fu Li, He Zheng, Weihang Yuan, et al. Aim 2022 challenge on super-resolution of compressed image and video: Dataset, methods and results. In *Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part III*, pages 174–202. Springer, 2023. 3

[42] Yang Yang, Haiting Lin, Zhan Yu, Sylvain Paris, and Jingyi Yu. Virtual dslr: High quality dynamic depth-of-field synthesis on mobile platforms. *Electronic Imaging*, 28:1–9, 2016. 2

[43] Zhihao Yang, Wenyi Lian, and Siyuan Lai. Bokehornot: Transforming bokeh effect with image transformer and lens metadata embedding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 4, 5

[44] Eduard Zamfir, Marcos V Conde, and Radu Timofte. Towards real-time 4k image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 3

[45] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 6

[46] Xuaner Zhang, Kevin Matzen, Vivien Nguyen, Dillon Yao, You Zhang, and Ren Ng. Synthetic defocus and look-ahead autofocus for casual videography. *arXiv preprint arXiv:1905.06326*, 2019. 2

[47] Bingke Zhu, Yingying Chen, Jinqiao Wang, Si Liu, Bo Zhang, and Ming Tang. Fast deep matting for portrait animation on mobile phone. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 297–305, 2017. 2