Neural Bokeh: Learning Lens Blur for Computational Videography and Out-of-Focus Mixed Reality

David Mandl [1*] Shohei Mori [1] Peter Mohr [1] Yifan Peng [2]
Tobias Langlotz [3] Dieter Schmalstieg [4,1] Denis Kalkofen [5,1†]

[1] Graz University of Technology [2] The University of Hong Kong [3] University of Otago [4] University of Stuttgart [5] Flinders University

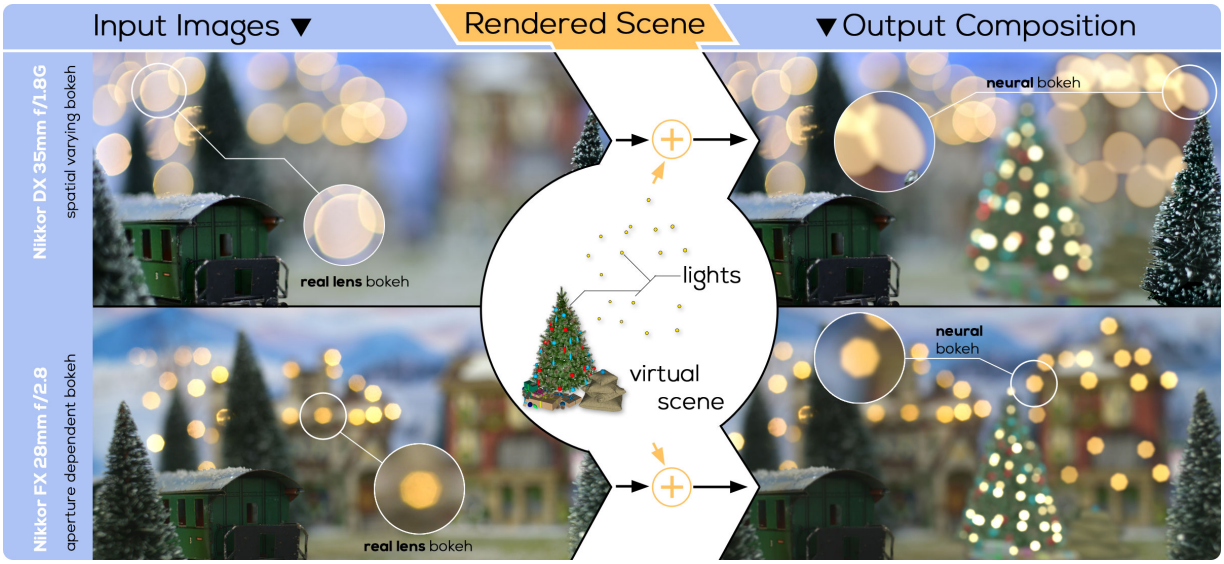

Figure 1: Neural Bokeh enables coherent compositing of synthetic all-in-focus scene elements with photographs that facilitate out-of-focus blur. (top) The photograph contains bokeh resulting from a rounded aperture, which varies with its position in image space. Neural Bokeh can learn the appearance and spatial variation to apply them to virtual out-of-focus scene content (for example, the Christmas tree and the decoration lights on the house). (bottom) The photograph contains bokeh with a heptagon shape, which is generated by applying Neural Bokeh to the elements of the virtual scene to support coherent composition.

Abstract

We present *Neural Bokeh*, a deep learning approach for synthesizing convincing out-of-focus effects with applications in Mixed Reality (MR) image and video compositing. Unlike existing approaches that solely learn the amount of blur for out-of-focus areas, our approach captures the overall characteristic of the bokeh to enable the seamless integration of rendered scene content into real images, ensuring a consistent lens blur over the resulting MR composition. Our method learns spatially varying blur shapes, i.e., bokeh, from a dataset of real images acquired using the physical camera that is used to capture the photograph or video of the MR composition. Accordingly, those learned blur shapes mimic the characteristics of the physical lens. As the run-time and the resulting quality of Neural Bokeh increase with the resolution of input images, we employ low-resolution images for the MR view finding at runtime and high-resolution renderings for compositing with high-resolution photographs or videos in an offline process. We envision a variety of applications, including visual enhancement of image and video compositing containing creative utilization of out-of-focus effects.

*e-mail: mandl@icg.tugraz.at
†e-mail: kalkofen@icg.tugraz.at

1 Introduction

Three-dimensional (3D) compositing is an essential component in visual effect productions, where the visual characteristics of rendered and captured footage must be aligned. Precisely aligning the rendering to the specific character of the camera footage requires experienced visual effects specialists and involves complex software suites, such as Foundry Nuke[1] or Blackmagic Fusion[2]. Such software tools rely on a time-consuming workflow to manipulate rendering and composition.

Photo and video editing for social media, such as Snap Lens-Studio, Adobe Premiere Rush, and Adobe Aero, are designed for casual users seeking simplified image editing capabilities. However, our objective extends beyond providing basic editing tools and aims to support high-quality image compositions using Mixed Reality (MR) at runtime (during view finding). In order to facilitate the creation of high-quality image compositions by casual users, it is imperative to establish a simple yet effective workflow that automatically aligns rendering parameters. Recent research in the field of automatic light and material estimation [28, 29, 36, 52], depth estimation [3, 9], and camera simulation [22, 27] has enabled visually coherent renderings. However, despite the impressive results of these approaches in representing scene elements that are close to the focal plane, precisely and automatically mimicking out-of-focus

[1] https://www.foundry.com/products/nuke-family
[2] https://www.blackmagicdesign.com/products/fusion

lens blur remains challenging.

In virtual environments, attainment of physically plausible defocus blur can be achieved by ray tracing [23]. However, when incorporating actual camera footage and digital content into a 3D composition, it becomes necessary to replicate the out-of-focus characteristics of the physical system. These characteristics commonly vary per lens, across the image sensor, with focus distance, and based on the size and shape of the aperture. Consequently, casual 3D compositing often mixes all-in-focus (AIF) footage with AIF rendering. By combining the two, an AIF composite is formed, allowing for the application of arbitrary out-of-focus effects to virtual and real scene elements. However, relying solely on AIF footage precludes the use of blurry source materials and camera systems, imposing undesired constraints on the creative process.

In this work, we propose Neural Bokeh, a deep learning approach to automatically generate a coherent defocus blur in MR renderings. The method is based on an end-to-end neural network for learning an implicit model of a physical lens from images captured through said lens. It is built upon the insight that the lens model can be inherently characterized through a comprehensive image database, readily available from the physical lens. Given the potentially large number of parameters associated with the model, we opt to directly learn its impact on the image, rather than the parameters, to explicitly synthesize blur using techniques such as ray tracing.

The input to our system comprises an RGB photograph, accompanied by its meta-data, which includes information about the aperture size and focus distance, an object mask to account for occlusions, and an AIF RGB-D rendering of virtual scene elements to be inserted via 3D compositing. The blur shape is learned by concatenating residual blocks with per-channel convolutions, gradually increasing in size. The maximal size of these convolutions is identified based on the range of depth and focus distance. To also learn spatially varying blur shapes, we partition the input image into overlapping tiles. These tiles are processed separately, but are evaluated in conjunction during training to mitigate potential color discrepancies between tiles.

Figure 1 demonstrates the effectiveness of Neural Bokeh on images captured with two different lenses (Figure 1, left). While the bladed aperture of the Nikkor FX 28 mm lens is blurring out-of-focus points to a characteristic heptagon pattern, the vignetting of the Nikkor DX 35 mm lens is producing varying blur shapes across the image. Elliptical blur shapes are more prominent near the border, whereas circular shapes dominate near the center. Due to the fact that our system has been trained with camera images from the physical lens systems, it is capable of applying both effects to an AIF rendering of virtual scene elements (Figure 1 middle), thereby ensuring a coherent blur across the MR composition (Figure 1, right).

Our approach is based on previous work on neural rendering of camera effects in virtual reality (VR) and augmented reality (AR) [27, 49], but with several crucial distinctions. Pure VR does not require compositing and, therefore, does not necessitate a physical camera lens model. In contrast, AR does consider lens models, but often trades the quality of its approximations for real-time performance. Neural Bokeh, on the other hand, introduces the first approach that aims to precisely replicate the out-of-focus blur of a real lens. As runtime performance and the resulting quality of Neural Bokeh are contingent on the resolution of the input image, we provide low-resolution images for online Mixed Reality (MR) view finding and high-resolution images and videos for high-quality compositions computed on demand. Ultimately, our system's ability to automatically render 3D objects with the characteristics of a specific physical lens has the potential to make visual effects production more accessible to casual users. In summary, we make the following contributions:

- We present the design and prototype of a neural network tailored to learning and reproducing spatially varying out-of-focus lens blur with high precision. Our design incorporates several key features, including (I) residual blocks with per-channel convolutions of varying size, (II) an approach to optimizing the number of network layers based on the maximal size of the circle of confusion (CoC), and (III) image tiling with global optimization to facilitate the learning of spatial variations while maintaining global consistency.
- We evaluate the efficacy of our neural rendering scheme by conducting an ablation study and by comparing its results against photographs and synthetic datasets with realistic out-of-focus blur generated using time-consuming ray-tracing techniques.
- We demonstrate the practical benefits of learned lens blur for out-of-focus composition through experiments conducted on several learned physical lenses.

2 Related work

In optical imaging, DoF represents the region between the nearest and furthest elements of a scene that appear to be visually sharp. Given an optical system, the DoF can usually be determined by the focal length, the distance from the subject, and the aperture [1, 4]. In photography and film production, the blur that is produced from scene elements outside the DoF often has a specific character and is, in this context, referred to as bokeh [26]. DoF and bokeh have recently become features that even non-professional users desire for artistic purposes. Expressive freedom is granted by the large variety of blur effects caused by deviations in lens aberrations and aperture shapes. This section reviews state-of-the-art approaches to rendering the bokeh.

2.1 Rendering defocus blur

In the computer graphics community, researchers have established a large body of physics-based rendering algorithms that can synthesize natural-looking lens blur [25]. For example, physically plausible DoF effects can be achieved by sampling through rendering or capturing the aperture of the camera system using pinhole cameras [15, 30]. While rendering and subsequently sampling many images is time-consuming, several approaches address real-time bokeh rendering via post-processing [19, 37, 38]. However, the performance gains of these methods often come at the cost of visible artifacts [8].

High-quality rendering of lens effects can be achieved with distributed ray tracing [7], light field synthesis [16, 43], or focal stack composition [17], which all tend to incur a high computational cost due to dense sampling [10, 34]. Furthermore, artistic manipulation of bokeh rendering requires more domain knowledge of optical aberrations [47] than can be expected from casual users. For a more comprehensive review, we refer the reader to the literature [5].

Several attempts have been made to model aperture effects for bokeh rendering [24] and other deblurring purposes [45]. A common approach is to fit a model to a spatially varying effect observed via a dedicated calibration board [18], typically a 2D dot grid or a chessboard pattern [24], which requires long exposures in a dark room or otherwise in a single defocus image [53]. However, these models only approximate the optical behavior (e.g., assuming image-space blur or planar scenes [45] and symmetric lenses [41]). In general, complex optics and hard-to-model imperfections are neglected [21]. To address this issue, we train our network designed for mimicking spatially varying bokeh shapes from given images or photographs.

2.2 Deep learning for synthesizing defocus blur

With the rise of deep learning for image synthesis [31, 51], learning various DoF effects has been investigated as well. For example, DeepFocus [49] generates convincing defocus blur for synthetic scenes, but is primarily designed to approximate blur as it arises in the human eye at real-time update rates for display in VR headsets. It learns how blur depends on the depth and the circle of confusion

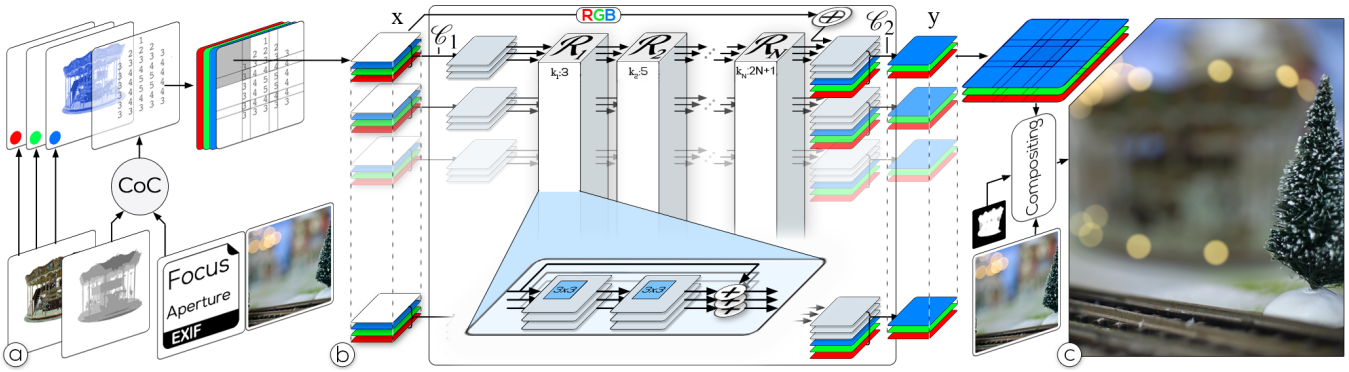

Figure 2: Overview. (a) Input to Neural Bokeh is an AIF image that is split into tiles and separated by R, G, and B channels and the circle of confusion (CoC) map, which encodes the depth per pixel and the desired focus distance. (b) Tiles are processed in the network, using a series of ResNet blocks with increasing kernel size. (c) The network outputs a list of blurred tiles, which are assembled into the output image and composited with a photograph.

(CoC). Similarly, Neural Cameras [27] only learn the parameters of a Gaussian mixture model, which enables fast rendering, but suffers from a coarse approximation. Both approaches aim for interactivity, but are not designed to accurately model the aperture shape and spatially varying properties of artistic bokeh are required for realistic bokeh.

DeepLens [44] trains the spatially varying blur kernels of a depth map and an RGB input image in a pipeline consisting of a network to segment foreground and background, a kernel prediction network and a feature map extraction network. This pipeline delivers low-resolution, shallow-DoF images, which can be upsampled to the desired resolution via a post-processing network. Although powerful, this pipeline is rather complex and does not address the need for an end-to-end solution.

As an alternative to the above approaches for learning lens blur, one can synthesize defocus blur via guided up-sampling and down-sampling for varying convolution layers with different kernel sizes and spatial resolutions [14]. This approach can generate convincing out-of-focus blur, but it is limited by its convolutional approach and lacks the characteristic round or bladed bokeh shapes that are expected from aperture shapes in real photography.

Recent neural rendering techniques seek to reconstruct the radiance of the scene as a function of 3D samples taken along the rays [32, 33, 51]. Specifically, DoF-NeRF [48] aims to reconstruct a CoC-dependent neural field trained per scene via a patch-based sampling strategy. Alternatively, an unsupervised approach [20] with a generative adversarial network can learn pairs of shallow DoF and deep DoF images from noise. While the reported results look convincing, these approaches assume a pinhole camera model, making it challenging to integrate characteristics of real lenses. A work developed concurrently with ours focuses on learning defocus blur to synthetically vary camera aperture and focus distance [2]. This work uses information from a dual-camera system, whereas ours is focusing on a more common monocular camera.

3 Method

The appearance of scene points outside the DoF depends on camera properties and scene geometry, such as focal length of the lens, distance to the object, aperture diameter, and scene depth [4]. However, camera design limitations, optical system imperfections, and physical properties of light often make generating an explicit appearance model difficult [24]. We develop an end-to-end neural renderer which directly mimics the appearance of such out-of-focus scene elements. Our approach is designed as a post-process to traditional virtual scene rendering, so that it can be combined with any rendering pipeline which supports combining real and virtual scene elements.

In the following, we provide details on how our network supports learning the blur shape for spatially varying effects. An overview of the runtime system of Neural Bokeh is shown in Figure 2.

3.1 Learning local bokeh effects

Rendering defocus blur requires spreading the contribution of a single point p in the scene onto several pixels in image space. Since scattering can be more efficiently implemented by gathering [11], we design a convolutional neural network that collects the contributions of the pixels surrounding the projection of p depending on the diameter c of the CoC, which is calculated as:

$$c = \frac{w_i}{w_s} \cdot a \cdot d_s \frac{|d_p - d_f|}{d_p \cdot d_f}, \tag{1}$$

where w_i is the image width in pixels, w_s is the sensor width in mm, a is the aperture diameter, d_s is the distance between the lens and sensor, d_p is the distance to the point p, and d_f is the distance to the object plane of the lens.

We account for chromatic aberrations by following the idea of Cholewiak et al. [6] of processing the color channels of each input pixel separately, together with c, forming a four-channel vector for each pixel as input to the network.

To efficiently learn the appearance, we reduce the number of input parameters and thus fix the aperture size, the width of the camera sensor, and the width of the image size, leading to different networks for different values of a, w_i, and w_s. From a combination of those parameters, we compute the maximal size of the CoC, which serves as a hyper-parameter for adjusting the number of convolutions in the network. In particular, our network $\mathscr{F}$ consists of N sequential residual blocks [12], denoted as $\mathscr{R}_n(\cdot)$, consisting of two per-channel convolutional layers with a kernel size of $k_n = 2n+1$, where $0 < n \leq N$, followed by a ReLU activation function. Additional convolutional layers $\mathscr{C}$ are placed before and after the residual blocks for conversion into a three-channel tensor. Note that $\mathscr{C}$ has a 1×1 kernel and does not contribute to bokeh effects. The color channels are concatenated with the output tensor of $\mathscr{R}_N$ via a skip connection, followed by a tanh activation function,

$$\mathscr{F}(\mathbf{x}) = \mathscr{C}_2(\mathscr{R}_N(\ldots \mathscr{R}_1(\mathscr{C}_1(\mathbf{x}))\ldots) \oplus \mathbf{x}_{RGB}), \tag{2}$$

where $\mathbf{x}$ is a tensor representing the concatenation of R, G, B, and CoC per pixel, and $\oplus$ the channel-wise tensor concatenation. Here, $\mathbf{x}_{RGB}$ is a tensor with only the RGB components of the input vector.

Similarly to Xiao et al. [**?**], we use residual blocks to effectively learn to blur the input while retaining the high-frequency content

located in focus. However, unlike previous approaches, we split the input and introduce a maximum kernel size, k_N, covering the largest CoC in the supported depth range for a single aperture size. The depth range is determined by the minimum and maximum depths in the training dataset. Therefore, the largest kernel size depends on the largest CoC in a given dataset.

3.2 Learning spatial variation

The network must also learn spatially varying blur shapes, because lens vignetting commonly results in a varying blur across the field of view of the camera. Therefore, we tile and aggregate an input image into a tensor $\mathbf{x}$, so that each tile has a different convolutional layer. The tiling suppresses learning of global effects, but instead enforces learning of tiled-area-dependent DoF effects. For that purpose, we first divide the 4-channel input of AIF RGB and CoC images into tiles with $t \times t$ pixels and t_o pixels overlap and then concatenate the resulting N_{tiles} tiles along the channel dimension.

Thus, the aim of our network $\mathscr{F} : \mathbf{x} \rightarrow \mathbf{y}$ is transforming the $t \times t \times 4N_{\text{tiles}}$ input tensor $\mathbf{x}$ into a $t \times t \times 3N_{\text{tiles}}$ output tensor $\mathbf{y}$. The output tensor consists of a set of RGB image tiles with defocus effects. In the final step, the image is reconstructed from tiles by placing them in their original position. Pixels in overlapping areas are blended by averaging the overlapping pixel colors to avoid visible seams between tiles.

3.3 Training

The training data consists of AIF and corresponding defocus images and CoC maps. To generate the data, we capture several defocus images at each viewpoint of an AIF, each image with a different focus distance. We use RAW images from the camera and convert each pixel to floating point. These values are normalized to [-1,1]. The raw pixel values are mapped into a linear space and additionally de-bayered using RawTherapee[3]. For each defocus image, we compute its CoC map using the depth map retrieved from the AIF image. While many approaches can be used to calculate per-pixel depth information [42], we leverage a recent approach for multi-view reconstruction based on AIF images of a single scene [39,40].

For training the network, we pair an AIF image with the CoC map of a corresponding defocus image to form the input $\mathbf{x}$. The error of a single training cycle is calculated by comparing the output of the network $\mathscr{F}(\mathbf{x})$ with the ground truth defocus image $\hat{\mathbf{y}}$ which corresponds to the CoC map in the input $\mathbf{x}$. We optimize the network using mean-square error (MSE) over the output $\mathscr{F}(\mathbf{x})$ and the defocus image $\hat{\mathbf{y}}$, i.e., we seek

$$\arg\min_{W} \text{MSE}(\mathscr{F}(\mathbf{x}), \hat{\mathbf{y}}), \tag{3}$$

where W denotes the network weights of $\mathscr{F}$.

While the network convolutions are independent between tiles, a loss of the entire image is computed and back-propagated to adjust the convolutions of all tiles. This optimization scheme avoids learning bias per tile, which would result in unbalanced colors on different tiles in a single image. We use photographed focal stacks rather than capturing randomly focused images, which lets us compute the AIF images from the corresponding defocused photographs. The training scheme is illustrated in Figure 3.

3.4 Compositing

When inserting synthetic objects into a real scene, we must adequately blend them to create the illusion that both would have come from the same camera. Therefore, we first render the synthetic scene using a pinhole camera model with the same intrinsic parameters as the real camera. To get the CoC map of the synthetic scene, we use the depth buffer and camera parameters to compute the diameter

[3]http://www.rawtherapee.com

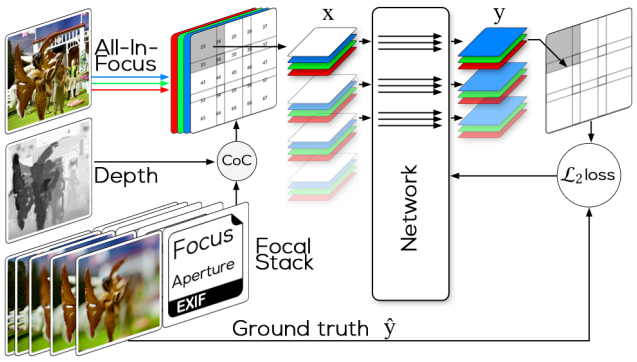

Figure 3: Training scheme. The network learns to simulate a specific bokeh by processing pairs of an all-in-focus (AIF) image and a circle-of-confusion (CoC) map into a defocused output image, which is compared to the ground-truth defocused photograph that corresponds to the CoC-map. AIF images are computed from captured focal stacks, so that the ground-truth defocused photograph and a corresponding AIF input image share the same viewpoint.

of the CoC for every pixel. Next, we use Neural Bokeh to blur the scene according to the learned real camera lens. We render the virtual objects on top of the real-world image to avoid an implausible pixel color leakage of the surroundings after applying Neural Bokeh.

We compose real and virtual scenes using alpha compositing. As a base mask image, we first render the objects of an unlit white material onto a black image. We then employ an erosion and a Gaussian filter to smooth the borders on this mask.

4 Evaluation

We validate our network design choice selected among those introduced by Neural Bokeh and compare our Neural Bokeh with the state-of-the-art approaches to reproduce bokeh effects for AIF images in both real and synthetic scenes. The datasets are available on the project's github repository [4].

4.1 Dataset

Evaluation by comparing photographed defocus blur with their synthesized counterparts is affected by the noise in the captured images. Since Neural Bokeh is based on HDR input the generated pixel colors are also affected by th operator that is used for tone mapping. Therefore, we created four synthetic datasets that simulate distinct bokeh shapes (see the Input and Ground Truth rows in Figure 5 for example renderings) for ablating the system components and for an isolated comparison of generated bokeh shapes. To train and test with images that are still close to real photographs, we render images using the path-tracing capabilities of VRay 6 for 3ds Max[5]. The dataset consists of rendered focal stacks and their corresponding AIF and depth images. The focal stacks are used as ground truth, while the AIF and depth images are used as input to the networks for generating the focal stacks that are compared against the ground truth renderings. In total, we render images from 15 viewpoints at random locations in 15 virtual environments to ensure a broad coverage of scene types. For each viewpoint, we render 10 differently focused images at a size of 1920×1080 pixels, resulting in 2250 images per evaluation dataset. In the following, we use the resulting data to validate our network design (Figure 4).

We also captured a similarly large dataset to learn the real lens systems demonstrated throughout this paper. We captured 143 focal stacks for Nikon Z6 and 138 for Nikon D3300. Each focal stack is

[4]https://immersive-technology-lab.github.io/projects/neuralbokeh
[5]https://www.chaos.com/vray/3ds-max

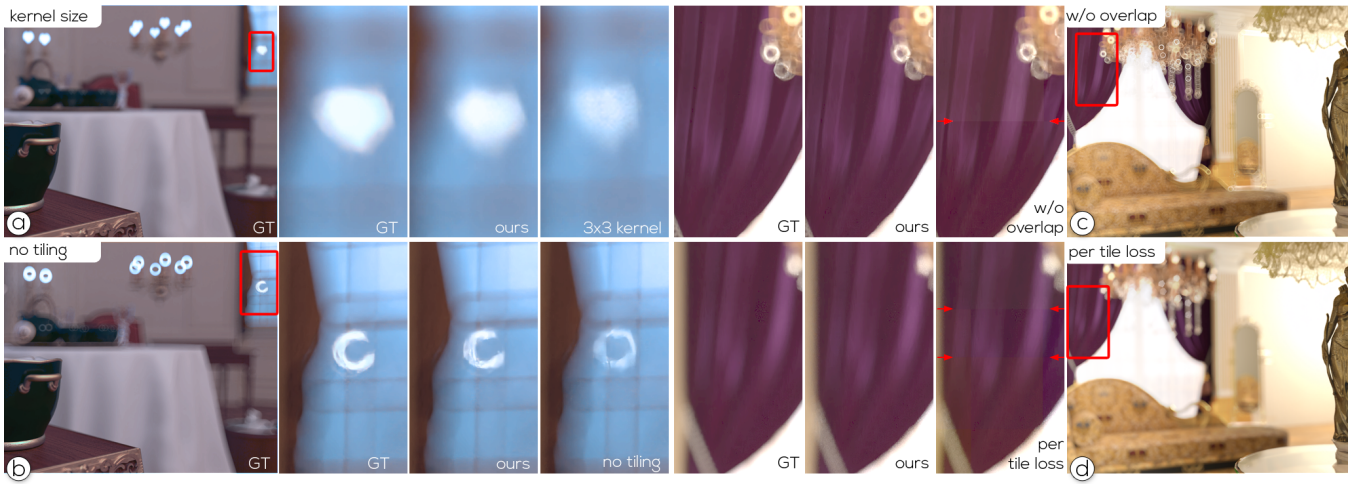

Figure 4: Design Validation. We compare the ground truth (GT) (from left to right) with the result of our network (ours) and the results of our network variants (a) using a 3×3 kernel size, (b) without tiling, (c) with tiling but without overlaps between tiles, (d) computing a loss for each tile separately.

composed of 30 focal slices. We relied on this real dataset to draw the quantitative and qualitative characteristics of our approach in comparison to the related work (Figures 6 and Table 3). For both real and synthetic datasets, we split the data into 80% training and 20% test data.

4.2 Design validation

To show the impact of our design choices, we validate each choice by comparing the results with those achieved with its naïve alternative. In particular, we validate the kernel size and tile sizes to learn spatially varying bokeh shapes.

Convolutions. The ResNet blocks in Neural Bokeh use increasing kernel sizes up to the size of the largest possible CoC. Since a convolution could be represented by multiple smaller convolutions, an alternative design would represent our kernels with a series of smaller fixed-sized convolutions. The alternative can reduce training and inference time due to the reduced parameters per block. However, fewer parameters can lead to inferior performance, and performance compensation with more layers requires more memory and longer inference. Also, our non-linear tanh activation functions differ from conventional linear concatenations of small kernels.

To demonstrate the impact of our design choice, we train our network with the same number of layers, but with 3×3 kernels, and compare it with the results of increasing kernel sizes. One additional layer of ResNet blocks with 3×3 kernels already exceeds the memory consumption of Neural Bokeh.

Figure 4(a) shows renderings achieved in both conditions. From the visual comparison, it can be seen that an increasing kernel size (labeled 'ours' in Figure 4(a)) reproduces the shape of the bokeh well, while the results achieved with the same network using 3×3 kernels suffer from missing contrast and poor shape representation. This observation is supported by the quantitative measurements shown in the first and second rows of Table 1, which present mean values for peak signal-to-noise ratio (PSNR) [13], structured similarity image metric (SSIM) [46] and learned perceptual image patch similarity (LPIPS) [50]. For example, while the network achieves a PSNR of 42.0 dB when using increasing kernel sizes, its variant using a 3×3 kernel size achieves only 36.9 dB. Similar differences can be observed for other measurements.

The images resulting from using constant 3×3 kernel size suffer from a lack of contrast. The difference in the bokeh becomes even more apparent. More network layers of 3×3 convolution kernels may improve the results. However, our tests show that memory consumption and inference time are already very similar between both approaches, with 2920 MB vs. 2867 MB and 2.90 seconds vs. 2.89 seconds for the increasing and 3×3 kernel size, respectively (measured on a laptop computer with an Intel i9-8950HK at 2.90GHz using an externally connected mid-class graphics processor, a GeForce RTX 2080). Table 2 shows that the overall performance increases when using a more recent PC system; however, the difference between a variable and a constant kernel size with more network layers is similar on both rendering systems. The performance values shown in Table 2 were measured using a recent PC system (Intel i7-12700 3.6 GHz, 128 GB Ram, Geforce RTX 4090), for several resolutions and maximal CoC sizes (two max. CoC sizes relate to real lens systems, Nikkor 28mm and Nikkor 35mm, and one to a virtual camera learned for validating the design choices).

Table 1: Performance measures of design variations. Mean values have been computed from 4×15 test images of rendered data shown in Figure 4 and Figure 5.

	PSNR↑	SSIM↑	LPIPS↓
Neural Bokeh (ours)	**42.0**	**0.981**	**0.017**
3×3 kernel size, Fig. 4(a)	36.9	0.971	0.030
No tiling, Fig. 4(b)	35.3	0.964	0.035
Tiling w/o overlap, Fig. 4(c)	40.2	0.980	0.018
Loss per tile, Fig. 4(d)	38.1	0.979	0.022

Image tiling. Our design supports image-space variations by splitting the input image into tiles, which are processed separately. The impact of our design decision is demonstrated in Figure 4(b). The physically correct spatial variation of the circular bokeh appears thick on the left, thin in the middle, and open on the right side of the image. As can be seen in the insets, learning the bokeh for each tile enables reproducing such spatial variations, while learning the appearance over the entire image produces a uniformly shaped bokeh, as demonstrated by the closed, round bokeh in Figure 4(b - no tiling). In addition to the visual improvements, we can also see an improvement in all measurements when learning the bokeh per tile (see 'No tiling' vs. 'Neural Bokeh' in Table 1).

Although tiling improves the results, separation within the network could potentially introduce color shifts over different tiles, and

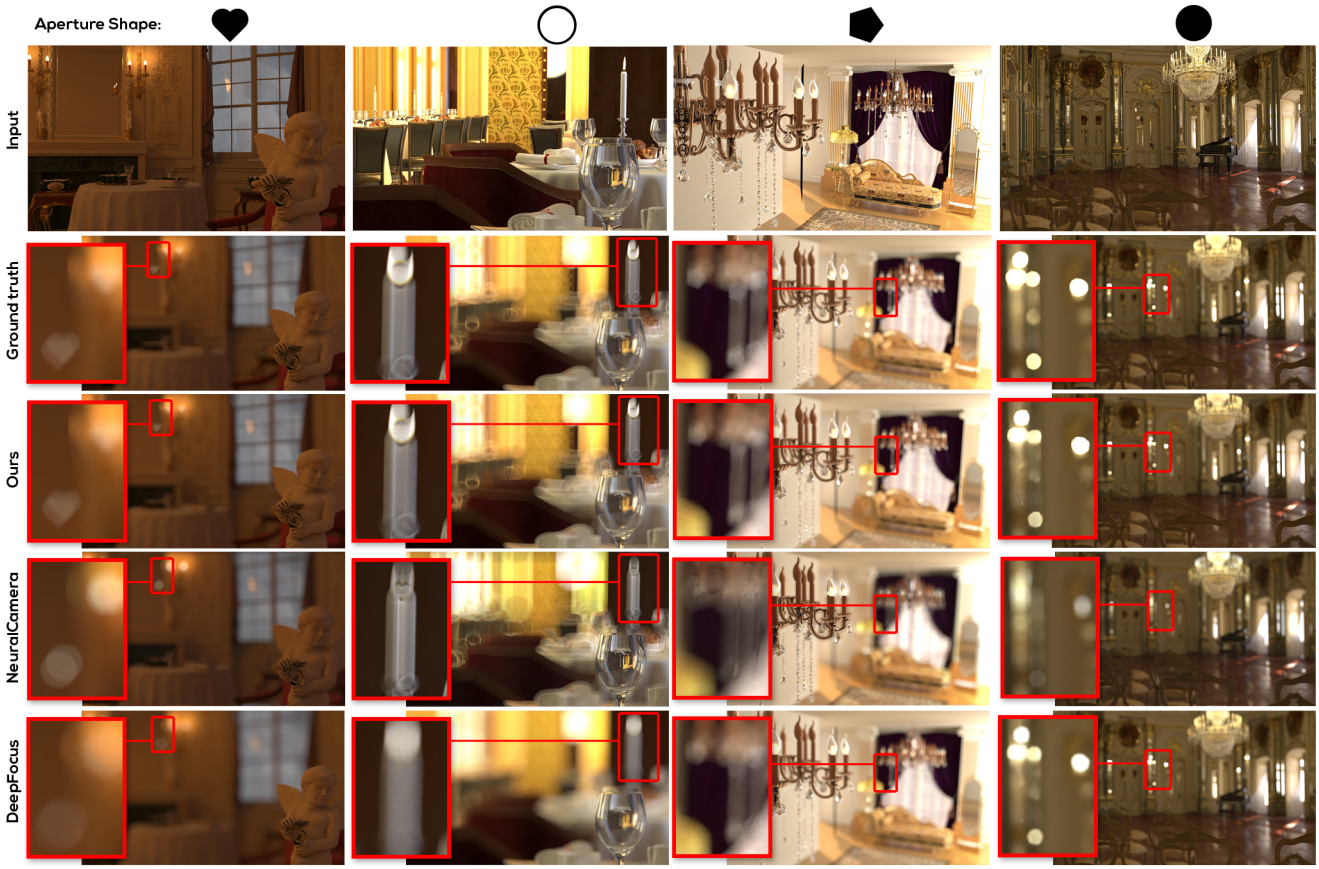

Figure 5: Comparison on renderings. We compare the results of Neural Bokeh with ground truth renderings and photographs, and with results achieved with NeuralCamera [27] and DeepFocus [49].

Table 2: Inference times (in ms) of Neural Bokeh using varying kernel sizes.

Maximal diameter of the CoC	360p	720p	1080p	1440p
Small (50 pixel)	15.69	15.57	32.34	46.31
Medium (65 pixel)	31.25	31.20	31.24	47.21
Large (81 pixel)	31.33	38.35	157.5	251.3

the amount of color shifts depends on the color distribution in the training dataset. To overcome this issue and balance the colors of neighboring tiles, we introduce overlaps between tiles and compute the training loss over the image instead of separately for each tile. To validate our design, we compare our results to those obtained using non-overlapping tiles and per tile loss. The tile size in all tests was set to 200×200 pixels, and the additional overlap was set to 60 pixels on each side.

Figure 4(c - w/o overlap) and Figure 4(d - per tile loss) show visible seams introduced by mismatching colors of neighboring tiles. Computing the loss only per tile seems to have a stronger negative impact. The tiling strategy without overlapping borders achieved a PSNR of 40.2 dB, whereas overlapping tiles with only a local loss per tile could reach only a PSNR of 38.1 dB (see 'Tiling w/o overlap' vs. 'Loss per tile' in Table 1).

4.3 Comparison to related work

To assess the performance of Neural Bokeh, we compare it with current approaches. In particular, we trained the neural networks proposed by Xiao et al. [49] and Mandl et al. [27] with the renderings we generated to validate our design (discussed in the previous section) and with photographs captured with a Nikon Z6 camera using a 35mm lens and a Nikon D3300 camera using a 28mm lens.

Table 3: Quantitative comparison with state-of-the-art approaches.

		Ours	[27]	[49]
Heart bokeh Fig. 5 (1st column)	PSNR↑	**42.1**	26.7	29.4
	SSIM↑	**0.983**	0.912	0.919
	LPIPS↓	**0.015**	0.095	0.095
Circle bokeh Fig. 5 (2nd column)	PSNR↑	**41.3**	25.6	31.6
	SSIM↑	**0.976**	0.856	0.916
	LPIPS↓	**0.020**	0.107	0.103
Bladed bokeh Fig. 5 (3rd column)	PSNR↑	**42.3**	26.4	35.2
	SSIM↑	**0.983**	0.915	0.967
	LPIPS↓	**0.016**	0.091	0.031
Round bokeh Fig. 5 (4th column)	PSNR↑	**42.3**	26.0	35.2
	SSIM↑	**0.982**	0.912	0.972
	LPIPS↓	**0.017**	0.094	0.039
Nikon Z6 w/ Nikkor 35mm Fig. 6 (top)	PSNR↑	**26.2**	21.6	23.6
	SSIM↑	**0.946**	0.896	0.921
	LPIPS↓	**0.057**	0.083	0.068
Nikon D3300 w/ Nikkor 28mm Fig. 6 (bottom)	PSNR↑	**31.2**	28.7	29.7
	SSIM↑	**0.944**	0.920	0.932
	LPIPS↓	**0.133**	0.142	0.155

Evaluation on renderings. As shown in Figure 5 and Table 3, Neural Bokeh is capable of producing high-quality out-of-focus renderings preserving the specific bokeh characteristic. Figure 5

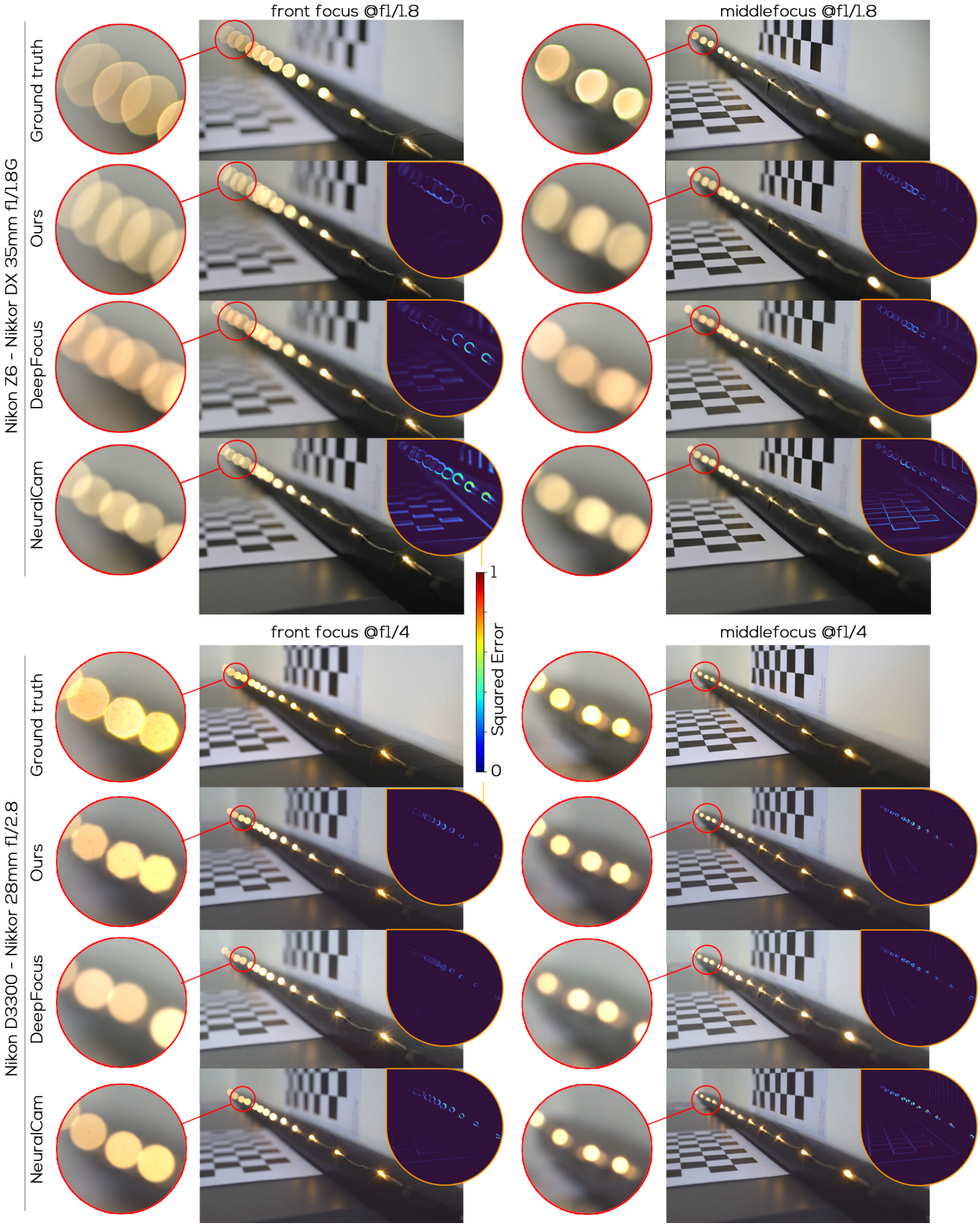

Figure 6: Comparison on photographs. We compare the results of Neural Bokeh with ground-truth photographs, with DeepFocus [49], and with NeuralCamera [27] on two camera systems (Nikon Z6 - Nikkor DX 35mm, Nikon D3300 - Nikkor 28mm) and with two focus settings. In addition, we show a squared error visualization in color to highlight erroneous pixels in each of the test cases. (left) Focus set to the front of the scene, i.e. close to the camera. (right) Focus set to the middle of the scene.

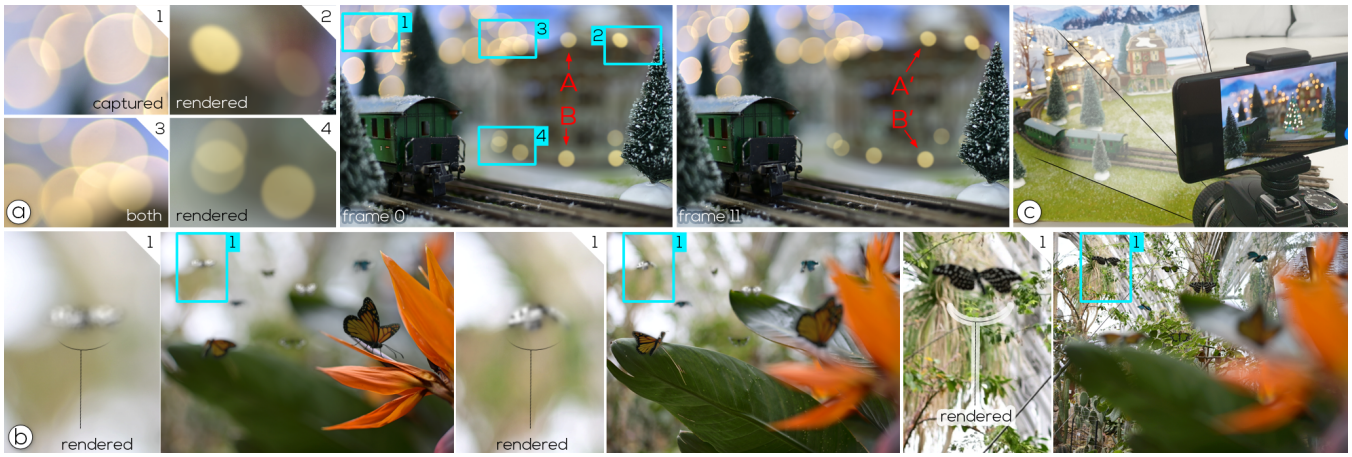

Figure 7: Mixed Reality compositing. (a) Rendering for compositing a 3D animation. The rendered bokeh follows the bokeh present in the photograph and, thus, coherently changes depending on its position in image space. In this example, Neural Bokeh introduces the cat-eye bokeh present in the photograph to the rendering of the virtual scene (the carousel and the decoration lights on the carousel). (b) Rendering for composing a focal stack. The focus distance differs between the image on the left, the image in the middle, and the image on the right. (c) Our Mixed Reality viewfinder enables previews of the composition of captured images with out of focus rendered scene elements. The system streams images from a camera, in this example, a Nikon D3300 with a detachable 28mm lens, to a PC, where it is augmented with renderings of out-of-focus scene elements. The result is sent back to a smartphone for displaying the composition..

shows how closely Neural Bokeh is approximating the ground truth rendering for all four bokeh shapes shown. Existing approaches generate more uniform bokeh shapes, as seen in the differences in bokeh shape reproducibility of the heart, ring, and bladed bokeh. The other approaches can generate results similar to Neural Bokeh on simple bokeh shapes (e.g., the rounded bokeh shown in the rightmost column of Figure 5). Overall, all approaches learned depth-dependent blur from the training dataset, and thus, they generated bokeh at out-of-focus pixels and kept sharp pixels at in-focus pixels. Notably, Neural Bokeh is the most useful when mimicking unusual bokeh shapes.

Evaluation on photographs. To visually investigate the results achieved by Neural Bokeh also on photographs, we have set up a test scene that includes a set of LED lights, which we have placed at several distances in front of the camera. This setup allows us to explore and compare the quality of the bokeh simulation over a certain depth range. Figure 6 shows the setup, the ground-truth camera images of Nikon Z6 with Nikkor DX 35mm and Nikon D3300 with Nikkor 28mm at two different focus distances, and the results achieved by the approaches. The rendering achieved by each approach is composited with an AIF image, which we computed by capturing a focal stack of the test scene.

Table 3 additionally provides PSNR, SSIM, and LPIPS results, which show that Neural Bokeh is capable of producing superior results in comparison to recent state-of-the-art approaches. The squared error visualization in Figure 6 also supports this. Overall, we noticed that the measurements shown in Table 3 are consistent between real and synthetic data but overall higher for synthetic data. We believe the lower values are caused by camera noise and other aberrations, such as highlights on the border, which Neural Bokeh has not picked up. By investigating the error visualization and the results in Table 3 together, we can further see that the 35mm lens produced lower measurements and larger errors compared to the 28mm lens. This demonstrates that the error depends on the bokeh size, i.e., the larger the bokeh, the higher the measured error becomes. This might also explain why the results achieved with the NeuralCamera approach increase for the D3300 dataset compared to the synthetic data, even though the camera introduces noise.

Neural Bokeh is capable of closely mimicking the shape of the bokeh for both cameras and in both focus settings consistently at the various distances of LEDs. Apparently, Neural Bokeh can generate superior results compared to previous work [27, 49]. However, none of the approaches correctly mimics the bright borders of the bokeh. We noticed that these aberrations appear in a high-frequent and small spatial extent. Thus, we will further investigate solutions to better reproduce high-frequent effects.

5 Applications

We developed Neural Bokeh to support casual photographers with tools to produce composites of shallow DoF photographs and out-of-focus rendering effects. It is agnostic regarding the actual content of the composition, enabling several other relevant applications.

Composing videos. Throughout the tests, we found that Neural Bokeh is capable of generating temporally coherent renderings and, therefore, supports processing a time series of images, such as the composition of tracked video footage and 3D animations. Figure 7(a) shows two frames of a 3D animation, which has been composited with a photograph. The animation shows the synthetic content of a rotating carousel with attached decoration lights that rotate with the carousel. The blur characteristic between the two frames shown in Figure 7(a) is temporally consistent. The bokeh of the light B and B' is consistent throughout the animation, while the bokeh of the light A and A' changes from a round shape to an oval shape. This change occurs because A' is located close to the image corner, where the bokeh becomes more distorted. This example shows that Neural Bokeh is capable of learning spatially varying the lens characteristics across the image. See the accompanying video for further results of temporal composites.

Mixed Reality viewfinder. Neural Bokeh is capable of generating high-quality bokeh effects. However, its runtime scales with the image resolution and depth range of the camera (i.e., the maximal size of the CoC, as it impacts the number of ResNet blocks in the network). See Table 2 for details of the runtime performance of Neural Bokeh on a recent PC system. To introduce an MR viewfinder that performs at real-time update rates, we stream the preview of the camera to the PC system and the resulting composition back to a smartphone. The phone is mounted on the camera so that the display of the composition follows any camera movements. Figure 7(c) shows our example implementation using a Nikon camera for pre-

Figure 8: Applying bokeh characteristics to AIF images. Neural Bokeh enables several application cases in addition to post-processing an AIF rendering to align the appearance of virtual objects in out-of-focus areas with a single photograph that shows a shallow DoF. Results of applying the bokeh of a specific learned lens system to an AIF image captured with an arbitrary lens system. The AIF images are part of a public database [14]. The upper row shows the characteristic bladed bokeh of the learned Nikkor 28mm lens, and the cat-eye bokeh which has been learned from images taken with the Nikkor 35mm lens is shown in the lower row. Depth estimation errors (orange highlight) cause wrongly blurred pixels, since the bokeh effect is directly affected by the depth.

view and image capturing and a Samsung Galaxy S9 to display the resulting composition rendered on a nearby PC system.

Streaming the preview of a high-quality camera system to a PC and the result to the display of a smartphone enables us to observe the composition while investigating potential view points. Once an appropriate camera view or camera path has been identified, the photographer can capture high-resolution image data that will subsequently be composited with high-resolution renderings of the virtual scene that is shown on the smartphone.

Composing focal stacks. Support for temporal coherency is also required for compositing photographed focal stacks and videos. A popular use case of focal stacks is refocusing to keep moving scene elements in focus or to guide the user's attention in a video composition towards a certain scene object. Figure 7(b) shows three images of a focal stack composite that have been generated using Neural Bokeh. The virtual scene consists of several butterflies which have been composited to a captured focal stack. Neural Bokeh believably simulates the blur present in the photograph so that the characteristic bokeh appears in the renderings of the virtual butterflies. The smooth bokeh in the image on the left becomes harder in the image in the middle. The image on the right is focused on the background of the scene, revealing the black butterflies with the white spots in the background of the scene. Neural Bokeh enables smooth changes to the focus distance in a video. However, the smoothness of the transition depends on the number of images in the captured focal stack. To increase the number of images in a focal stack, we can generate an AIF image and a depth map from the focal stack and use Neural Bokeh to generate a larger focal stack.

Transferring lens characteristics to AIF images. Neural Bokeh can apply learned out-of-focus blur to AIF input images. A specific use case applies the unique bokeh of rare or expensive lenses to images taken by a camera without such lenses. Applying learned bokeh of rare lenses can significantly reduce the costs of movie or print productions that would otherwise require hard-to-find optical components to obtain a unique look. In fact, given one-time access to capture training data, Neural Bokeh can be used to preserve the bokeh characteristics of historic lenses.

Figure 8 show examples of the learned bokeh used in Figure 1 applied to AIF images. The rendering in the upper row of Figure 8 shows the characteristic bladed bokeh of the Nikor 28mm lens, while the renderings in the lower row shows the cat-eye bokeh that is learned from the data captured with the Nikor 35mm lens. Since Neural Bokeh requires per-pixel depth information, we use Ranftl et al. [35] to derive the depth map from an AIF image before rendering the bokeh. While recent monocular depth estimators are capable of generating high-quality depth maps, depth estimation of small structures is still challenging, which may causes wrong depth values in such areas in an image. Figure 8 highlights such a case (see the organge circle) and demonstrates how errors in depth estimation causes errors in CoC calculation, which lead to a wrong blur in such areas. Therefore, high quality depth estimation is a key factor for a high-quality transfer of lens characteristics to AIF images.

6 Conclusion and Future Work

This work introduces Neural Bokeh, an approach to learning the bokeh appearance of physical lenses. Contrary to existing approaches that focus mainly on learning the amount of blur, our approach produces more realistic bokeh characteristics in out-of-focus renderings, including effects caused by aperture shapes and spatial variations from lens distortion.

We validate our design decisions and demonstrate the superiority of our approach over state-of-the-art methods in datasets from both synthetic and physical cameras. Neural Bokeh has significance for multiple areas in computational photography and videography, in particular when composing AIF content from renderings. This capability can be crucial when simulating large aperture effects in mobile phone photography, where small sensors and small apertures are common. Since Neural Bokeh is temporally coherent, it has large potential in video compositing applications, which require video sequences or 3D content from different sources to be coherently rendered whilst supporting physically correct out-of-focus effects.

Apart from a general optimization for mobile devices, future work may explore the simulation of additional lens characteristics (e.g., from anamorphic optics that introduce characteristic lens flare). In addition, we consider depth estimation from out-of-focus objects an interesting direction for future work. We believe that having a system that is able to simulate a specific lens blur can be beneficial in estimating the depth of points that fall outside the depth of field.

Acknowledgments

This work was supported by the Austrian Science Fund FWF (P30694), Snap Inc., a Marsden Grant MFP-UOO2124, and the Research Grants Council of Hong Kong (ECS 27212822).

REFERENCES

[1] E. Allen and S. Triantaphillidou. *The manual of photography*. CRC Press, 2012.

[2] H. Alzayer, A. Abuolaim, L. C. Chan, Y. Yang, Y. C. Lou, J.-B. Huang, and A. Kar. DC2: Dual-camera defocus control by learning to refocus. In *Proc. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 21488–21497, 2023.

[3] A. Atapour-Abarghouei and T. Breckon. Real-time monocular depth estimation using synthetic data with domain adaptation. In *Proc. Computer Vision and Pattern Recognition*, pp. 1–8. IEEE, June 2018.

[4] S. Bae and F. Durand. Defocus magnification. In *Computer Graphics Forum*, vol. 26, pp. 571–579. Wiley Online Library, 2007.

[5] B. A. Barsky and T. J. Kosloff. Algorithms for rendering depth of field effects in computer graphics. In *WSEAS international conference on Computers*, vol. 2008. World Scientific and Engineering Academy and Society (WSEAS), 2008.

[6] S. A. Cholewiak, G. S. Love, P. P. Srinivasan, R. Ng, and M. S. Banks. Chromablur: Rendering chromatic eye aberration improves accommodation and realism. *ACM Trans. on Graphics (TOG)*, 36, 2017.

[7] R. L. Cook, T. Porter, and L. Carpenter. Distributed ray tracing. In *Proc. SIGGRAPH*, pp. 137–145, 1984.

[8] J. Demers. Depth of field: A survey of techniques. In *GPU Gems: Programming Techniques, Tips, and Tricks for Real-Time Graphics*, chap. 23, pp. 375–390. Addison-Wesley Professional, 2004.

[9] R. Du, E. Turner, M. Dzitsiuk, L. Prasso, I. Duarte, J. Dourgarian, J. Afonso, J. Pascoal, J. Gladstone, N. Cruces, S. Izadi, A. Kowdle, K. Tsotsos, and D. Kim. DepthLab: Real-time 3D Interaction with Depth Maps for Mobile Augmented Reality. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, UIST '20, pp. 829–843. ACM, 2020. doi: 10.1145/3379337.3415881

[10] P. Haeberli and K. Akeley. The accumulation buffer: Hardware support for high-quality rendering. In *Proc. Conf. on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pp. 309–318, 1990. doi: 10.1145/97879.97913

[11] E. Hammon, Jr. Practical post-process depth of field. In *GPU Gems 3*, chap. 28. Addison-Wesley Professional, 2007.

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[13] A. Hore and D. Ziou. Image quality metrics: PSNR vs. SSIM. In *Int. Conf. on Pattern Recognition*, pp. 2366–2369, 2010.

[14] A. Ignatov, J. Patel, and R. Timofte. Rendering natural camera bokeh effect with deep learning. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 418–419, 2020.

[15] A. Isaksen, L. McMillan, and S. J. Gortler. Dynamically reparameterized light fields. In *Proc. Conf. on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pp. 297–306, 2000. doi: 10.1145/344779.344929

[16] A. Isaksen, L. McMillan, and S. J. Gortler. Dynamically reparameterized light fields. In *Proc. Conf. on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pp. 297–306, 2000.

[17] D. E. Jacobs, J. Baek, and M. Levoy. Focal stack compositing for depth of field control. *Stanford Computer Graphics Laboratory Technical Report*, 1(1):2012, 2012.

[18] J. Jang, J. D. Yun, and S. Yang. Modeling non-stationary asymmetric lens blur by normal sinh-arcsinh model. *IEEE Trans. on Image Processing*, 25(5):2184–2195, 2016.

[19] Y. Jeong, S. Y. Baek, Y. Seok, G. B. Lee, and S. Lee. Real-time dynamic bokeh rendering with efficient look-up table sampling. *IEEE Trans. on Visualization and Computer Graphics (TVCG)*, 28(2):1373–1384, 2020.

[20] T. Kaneko. Unsupervised learning of depth and depth-of-field effect from natural images with aperture rendering generative adversarial networks. In *Proc. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 15679–15688, 2021.

[21] E. Kee, S. Paris, S. Chen, and J. Wang. Modeling and removing spatially-varying optical blur. In *Proc. IEEE Int. Conf. on Computational Photography (ICCP)*, pp. 1–8. IEEE, 2011.

[22] G. Klein and D. W. Murray. Simulating low-cost cameras for augmented reality compositing. *IEEE Trans. on Visualization and Computer Graphics (TVCG)*, 16(3):369–380, 2010.

[23] C. Kolb, D. Mitchell, and P. Hanrahan. A realistic camera model for computer graphics. In *Proc. Conf. on Computer Graphics and Interactive Techniques (SIGGRAPH)*, SIGGRAPH '95, pp. 317–324. Association for Computing Machinery, New York, NY, USA, 1995. doi: 10.1145/218380.218463

[24] D. Lanman, R. Raskar, and G. Taubin. Modeling and synthesis of aperture effects in cameras. In *Computational Aesthetics in Graphics, Visualization, and Imaging*. The Eurographics Association, 2008.

[25] S. Lee, E. Eisemann, and H.-P. Seidel. Real-time lens blur effects and focus control. *ACM Trans. on Graphics (TOG)*, 29(4):1–7, 2010.

[26] X. Luo, J. Peng, K. Xian, Z. Wu, and Z. Cao. Bokeh rendering from defocus estimation. In *Proc. European Conf. on Computer Vision (ECCV)*, pp. 245–261. Springer, 2020.

[27] D. Mandl, P. M. Roth, T. Langlotz, C. Ebner, S. Mori, S. Zollmann, P. Mohr, and D. Kalkofen. Neural cameras: Learning camera characteristics for coherent mixed reality rendering. In *Proc. Int. Symp. on Mixed and Augmented Reality (ISMAR)*, pp. 508–516, 2021. doi: 10.1109/ISMAR52148.2021.00068

[28] D. Mandl, K. M. Yi, P. Mohr, P. Roth, P. Fua, V. Lepetit, D. Schmalstieg, and D. Kalkofen. Learning lightprobes for mixed reality illumination. In *Proc. Int. Symp. on Mixed and Augmented Reality (ISMAR)*, pp. 82–89, 2017.

[29] A. Meka, M. Maximov, M. Zollhöfer, A. Chatterjee, H. Seidel, C. Richardt, and C. Theobalt. LIME: live intrinsic material estimation. In *Proc. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 6315–6324, 2018. doi: 10.1109/CVPR.2018.00661

[30] R. Ng. Fourier slice photography. *ACM Trans. on Graphics (TOG)*, 24(3):735–744, July 2005.

[31] S. Niklaus, L. Mai, J. Yang, and F. Liu. 3d ken burns effect from a single image. *ACM Transactions on Graphics (ToG)*, 38(6):1–15, 2019.

[32] J. Peng, Z. Cao, X. Luo, H. Lu, K. Xian, and J. Zhang. Bokehme: When neural rendering meets classical rendering. In *Proc. Computer Vision and Pattern Recognition*, pp. 16283–16292, 2022.

[33] J. Peng, Z. Pan, C. Liu, X. Luo, H. Sun, L. Shen, K. Xian, and Z. Cao. Selective bokeh effect transformation. In *Proc. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 1456–1464, 2023.

[34] M. Potmesil and I. Chakravarty. Synthetic image generation with a lens and aperture camera model. *ACM Trans. on Graphics (TOG)*, 1(2):85–108, 1982. doi: 10.1145/357299.357300

[35] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, and V. Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020.

[36] T. Richter-Trummer, D. Kalkofen, J. Park, and D. Schmalstieg. Instant mixed reality lighting from casual scanning. In *Proc. Int. Symp. on Mixed and Augmented Reality (ISMAR)*, pp. 27–36, 2016.

[37] G. Riguer, N. Tatarchuk, and J. Isidoro. Real-time depth of field simulation. In *ShaderX2: Shader Programming Tips and Tricks with DirectX*, vol. 9, pp. 529–556. Wordware Publishing, Inc., 2004.

[38] T. Scheuermann and N. Tatarchuk. Improved depth of field rendering. In *ShaderX3: Advanced Rendering with DirectX and OpenGL (Shaderx Series)*. Charles River Media, 2004.

[39] J. L. Schönberger and J.-M. Frahm. Structure-from-motion revisited. In *Proc. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[40] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016.

[41] J. D. Simpkins and R. L. Stevenson. Parameterized modeling of spatially varying optical blur. *Journal of Electronic Imaging*, 23(1):013005, 2014.

[42] P. P. Srinivasan, T. Wang, A. Sreelal, R. Ramamoorthi, and R. Ng. Learning to synthesize a 4d RGBD light field from a single image. In *Proc. Int. Conf. on Computer Vision (ICCV)*, pp. 2262–2270, 2017.

[43] V. Vaish, B. Wilburn, N. Joshi, and M. Levoy. Using plane+ parallax for calibrating dense camera arrays. In *Proc. Conf. on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, pp. I–I. IEEE, 2004.

[44] L. Wang, X. Shen, J. Zhang, O. Wang, Z. Lin, C.-Y. Hsieh, S. Kong, and H. Lu. Deeplens: shallow depth of field from a single image. *arXiv*

preprint arXiv:1810.08100, 2018.

[45] R. Wang and D. Tao. Recent progress in image deblurring. *arXiv preprint arXiv:1409.6838*, 2014.

[46] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[47] J. Wu, C. Zheng, X. Hu, Y. Wang, and L. Zhang. Realistic rendering of bokeh effect based on optical aberrations. *The Visual Computer*, 26(6):555–563, 2010.

[48] Z. Wu, X. Li, J. Peng, H. Lu, Z. Cao, and W. Zhong. Dof-nerf: Depth-of-field meets neural radiance fields. In *Proc. ACM International Conference on Multimedia*, pp. 1718–1729, 2022.

[49] L. Xiao, A. Kaplanyan, A. Fix, M. Chapman, and D. Lanman. DeepFocus: Learned image synthesis for computational displays. *ACM Trans. on Graphics (TOG)*, 37(6), 2018. doi: 10.1145/3272127.3275032

[50] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proc. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[51] B. Zheng, Q. Chen, S. Yuan, X. Zhou, H. Zhang, J. Zhang, C. Yan, and G. Slabaugh. Constrained predictive filters for single image bokeh rendering. *IEEE Transactions on Computational Imaging*, 8:346–357, 2022.

[52] H. Zhou, S. Hadap, K. Sunkavalli, and D. W. Jacobs. Deep single portrait image relighting. In *Proc. Int. Conf. on Computer Vision (ICCV)*, 2019.

[53] X. Zhu, S. Cohen, S. Schiller, and P. Milanfar. Estimating spatially varying defocus blur from a single image. *IEEE Trans. on Image Processing*, 22(12):4879–4891, 2013.