

Performing Defocus Deblurring by Modeling its Formation Process

Zhengbo Zhang¹ Lin Geng Foo¹ Hossein Rahmani² Jun Liu^{2*} De Wen Soh¹
¹Singapore University of Technology and Design ²Lancaster University

Abstract

Single image defocus deblurring (SIDD) is a challenging task that aims to recover an all-in-focus image from a defocused one. In this paper, we make the observation that a defocused image can be viewed as a blend of illuminated blobs based on fundamental imaging principles, and the defocus blur in the defocused image is caused by large illuminated blobs intermingling with each other. Thus, from a novel perspective, we perform SIDD by adjusting the shape and opacity of the illuminated blobs that compose the defocused image. With this aim, we adopt a novel 2D Gaussian blob representation for illuminated blobs and a differentiable rasterization method to obtain the parameters of the 2D Gaussian blobs that compose the defocused image. Additionally, we propose a blob deblurrer to adjust the parameters of the 2D Gaussian blobs corresponding to the defocused image, thereby obtaining a sharp image. We also explore incorporating prior depth information via our depth-based regularization loss to regularize the size of Gaussian blobs, further improving the performance of our method. Extensive experiments on five widely-used datasets validate the effectiveness of our proposed method.

1. Introduction

Single image defocus deblurring (SIDD) aims to reconstruct an all-in-focus image from a single defocused and blurry input. SIDD is essential for removing unwanted blur from captured images, which often contain defocus blur resulting from photographing scenes with varying depths. Deblurring images through SIDD can also significantly enhance the performance of downstream machine vision tasks (e.g., object detection [6, 9] and semantic segmentation [34, 53]) for important practical applications. Existing SIDD works can be broadly divided into two approaches: i) Conventional SIDD approaches typically employ a two-step process involving defocus map estimation [8, 11, 16, 18, 28, 60] followed by deconvolution meth-

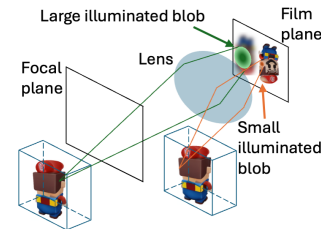


Figure 1. Due to the varying distances between objects and the lens, the sizes of the illuminated blobs formed on the film plane differ, resulting in some objects appearing sharp while others appear blurred (i.e., with larger illuminated blobs).

ods [3, 12, 21] using the estimated defocus maps for defocus deblurring. ii) Recently, end-to-end deep learning-based approaches [2, 24, 36–38, 45, 49] have been introduced for SIDD, where a deep neural network (DNN) is trained to map defocused images to their sharp counterparts. Existing state-of-the-art (SOTA) methods [37–39] are mostly based on deep learning, and typically focus on learning the defocus kernel or its inverse, employing intricately designed networks that utilize the learned defocus kernels or inverse kernels for SIDD. Despite recent progress in SIDD, the task remains challenging, particularly in scenarios involving defocus blur with large blur areas [38, 49], severely blurred regions [36], and irregularly shaped blurs [24, 45]. There are also challenges in terms of the model’s ability to generalize across different datasets [37].

To better address the challenging SIDD task, it is crucial to review how the phenomenon of defocus blur is formed from fundamental imaging principles. As shown in Fig. 1, defocus blur is a common “degradation” in optical systems caused by variations in scene depth during image capture. Within an optical system (e.g., camera), lenses are employed to refract incoming light, directing the light rays to converge at the apex of a conical path. When an object is positioned such that the apex of this cone aligns precisely with the film plane, the light rays form small illuminated blobs on the film plane, rendering the object in *sharp focus*. This specific distance where objects appear sharp and clear is known as the focal plane. Objects positioned away from the

*Corresponding author

focal plane cause the light rays to diverge, forming larger illuminated blobs on the film plane that intermingle with other blobs, thus producing a *blurry appearance* in the captured image [45, 49]. Hence, due to the inherent physical properties of the camera and its lenses, defocused images can be interpreted as being composed from numerous illuminated blobs of varying sizes, where the larger illuminated blobs lead to defocus blur in the resulting images. It is worth noting that the illuminated blobs here are not necessarily perfectly circular; they match the shape of the aperture and can also be oval, polygonal, or other shapes [42, 54].

In this work, based on the insights from defocused photography discussed above, we approach SIDD from a novel perspective, by *modeling the formation process of defocused images*. Then, by leveraging the modeled formation process, we can effectively perform deblurring on the defocused images to obtain sharp images. However, it is very challenging to model the formation process of a defocused image, since it is an inherently ill-posed problem. This difficulty arises because we often lack crucial information, *e.g.*, the camera’s focal length and aperture size, making it difficult to directly apply optical principles to compute the resulting defocus blur. Consequently, a key question naturally emerges: *how do we tackle SIDD according to the fundamental optical imaging principles of photography, despite its challenges and ill-posed nature?*

To address the above question and effectively model the formation process of defocused images, we draw inspiration from 3D Gaussian Splatting (3D GS) [19], a method for 3D scene representation that has gained significant attention in the areas of 3D scene deblurring [22], single-view reconstruction [27, 46, 50, 56, 68], novel view synthesis [4, 5, 10, 33, 67], 3D generation [51], and image compression [63]. Specifically, inspired by Deblurring 3D GS [22] and GaussianImage [63], we propose a novel approach that represents defocused images as stacks of 2D Gaussian blobs, *i.e.*, via 2D Gaussian blob representation [63]. Intuitively, by representing the defocused images with a set of 2D Gaussian blobs, we encourage *each 2D Gaussian blob to model an illuminated blob that contributes to the defocused image*, which allows us to better model the formation process of the defocus blur. Our method involves several key designs: First, to represent the defocused image via the 2D Gaussian blob representation, we design a rasterization method to facilitate optimization in an end-to-end differentiable manner. Next, after modeling the formation of the defocused image via our 2D Gaussian blob representation, we introduce a blob deblurrer that adjusts the shape and opacity of each 2D Gaussian blob to perform the deblurring. The adjustment by the blob deblurrer reduces the blur on defocused images by decreasing the size of the 2D Gaussian blobs (*i.e.*, illuminated blobs) and reducing their intermingling. Finally, to further improve the

quality of the conversion process between the defocused images to 2D Gaussian blobs, we enhance our method with a depth-based regularization loss using pre-trained depth estimators, which leads to further performance gains.

By taking a different approach as compared to previous methods, our proposed method holds a few advantages. To be specific, traditional two-stage SIDD methods typically rely on manually defined kernels to estimate the defocus blur, but the blurring patterns in real-world defocused images can be quite complex and may not conform to these manually defined kernels. Thus, trying to model the complex blurs with manually defined kernels may impede accurate defocus map estimation and successful defocus deblurring [45, 49]. Our method does not employ defocus map estimation and does not rely on any such kernels, thus we avoid facing such errors. On the other hand, current SOTA end-to-end DNN-based SIDD approaches [37–39] often operate under the assumption that the blurring in an image can be modeled by blurring kernels which are applied in a convolutional manner to the original sharp image. Through this modelling, they attempt to learn the inverse kernels (deconvolutional kernels) to effectively deblur the images. However, these works make the assumption that the blurring in the image is the result of a (convolutional) kernel, which might not hold well in real-world scenarios. Therefore, since our method does not rely on such assumptions, we avoid such issues that may affect our performance. Overall, our approach achieves more effective deblurring and demonstrates strong generalizability, as shown by our extensive experiments.

In summary, our contributions are as follows: (i) We tackle SIDD from a *novel perspective*, by viewing defocused images as blended illuminated blobs based on fundamental imaging principles. (ii) We propose a deblurring framework for performing SIDD on defocused images represented by 2D Gaussian blobs, that adjusts the size, shape and opacity of the 2D Gaussian blobs to achieve deblurring. (iii) To further improve performance, we introduce a depth-based regularization loss to regularize the Gaussian blob optimization process, regularizing the sizes of Gaussian blobs based on prior depth information. (iv) We conduct extensive experiments to assess the performance of our proposed framework, where we demonstrate SOTA performance across multiple benchmarks.

2. Related work

Two-stage SIDD. Conventional SIDD typically employs a two-stage approach: estimating a defocus map [3, 8, 11, 16, 18, 23, 28, 35, 48, 55, 60, 64, 65] from the input defocused image, followed by applying the non-blind deblurring methods [12, 21, 25, 28, 31, 41, 58] using the defocus kernels derived from the defocus map. The majority of two-stage approaches for the SIDD focus predominantly on the initial

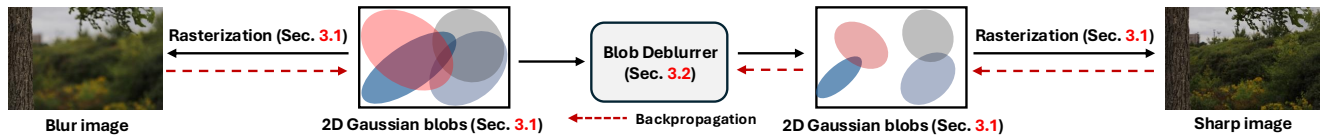


Figure 2. Illustration of our proposed SIDD pipeline. We first represent the defocused image using illuminated blobs (2D Gaussian blobs) through a rasterization method. Then, we adjust the size, shape, and opacity of the 2D Gaussian blobs with a blob deblurrer to minimize the intermingling between the blobs, thereby enhancing the clarity of the resulting image.

stage—defocus map estimation. The subsequent stage typically involves implementing existing non-blind deconvolution techniques [12, 21, 25]. Significant effort [16, 35, 48] has been dedicated to enhancing the accuracy of defocus map estimation, as it substantially influences the performance of deblurring processes. Despite significant efforts by researchers, the two-stage approaches for the SIDD still exhibit limitations. First, blur kernel estimation is typically based on manually defined kernels and may lack precision. Second, even with an accurately estimated blur kernel, deconvolution can introduce ringing artifacts along edges due to the Gibbs phenomenon [57]. Unlike traditional two-stage SIDD methods, which typically rely on manually defined kernels for defocus map estimation and subsequently deblurring defocused images, our method takes a novel perspective to SIDD. We represent defocused images using 2D Gaussian blobs and perform deblurring by adjusting the shape and opacity of these blobs.

End-to-end DNN-based SIDD. Recently, deep end-to-end learning has emerged as a promising approach for SIDD, with researchers increasingly adopting end-to-end DNN to directly restore sharp images from defocused images. The seminal work [1] involves training an encoder-decoder CNN that maps a defocused image to its in-focus counterpart, achieving significant performance gains over traditional two-stage methods. Deep end-to-end learning-based SIDD methods [2, 7, 17, 24, 26, 29, 32, 36–38, 44, 45, 49, 61, 66] generally surpass conventional two-step approaches in both performance and efficiency. However, current SOTA deep end-to-end learning approaches [37–39] for SIDD assume that the defocus blurring in the image can be modeled by a convolutional kernel, an assumption that may not be valid in real-world scenarios. Different from current SOTA deep end-to-end learning SIDD methods [37–39], our approach *does not assume* that the blur in defocused images is caused by specific blur kernels. We approach SIDD from a novel perspective by modeling the formation of defocused images based on fundamental imaging principles, specifically modeling how such defocused images result from the blending of various illuminated blobs. After representing the image with 2D Gaussian blobs, we adjust the size, shape and opacity of the blobs to reduce the intermingling between pixels, thereby yielding sharper images.

3D Gaussian Splatting (3D GS). 3D GS [19] represents

a significant advancement in 3D scene representations, enabling high-quality real-time rendering, and has been applied across various domains, including 3D scene deblurring [22], single-view reconstruction [27, 46, 50, 56, 68], novel view synthesis [4, 5, 10, 33, 67], 3D generation [13, 15, 51], and image compression [63]. Since 3D GS is an efficient representation [19, 63], its applications have led to significant advancements within each respective domain. In this paper, we adopt a novel perspective to model the fundamental principles underlying defocus deblurring. Inspired by [22, 63], we represent the defocused image using blended 2D Gaussian blobs, and adopt a rasterization method to facilitate the optimization. Our proposed approach effectively models the blur of the defocused image from the standpoint of imaging principles, leading to improved SIDD performance.

3. Method

Defocus blur is a degradation phenomenon in optical systems when capturing images of scenes with varying depths. Objects that are out of the focal plane have their light rays diverge, forming large illuminated blobs on the film plane that overlap with others, causing the objects to appear blurry in the captured image. To recover an all-in-focus image with sharp and clear details from a defocused image, our key insight is that we can represent the defocused image in terms of “illuminated blobs”, and then we can adjust the size, shape, and opacity of the illuminated blobs in defocused images to minimize the intermingling between them, thereby enhancing the clarity of the resulting image. The overall pipeline of our method is depicted in Fig. 2.

Specifically, from a new perspective, we propose using 2D Gaussian blobs to represent illuminated blobs in defocused images. To render images from 2D Gaussian blobs, we adopt a differentiable rasterization method (Sec. 3.1). Then, we introduce a blob deblurrer (Sec. 3.2) to adjust the size, shape and opacity of each 2D Gaussian blob, thereby performing the deblurring. Additionally, to improve the fitting of the defocused image to our 2D Gaussian blob representation, we design a depth-based regularization loss (Sec. 3.3), which regularizes the sizes of 2D Gaussian blobs based on their depths, since the size of illuminated blobs is often highly correlated with depth. Finally, we detail the

training and testing processes of our method (Sec. 3.4).

3.1. Fitting to our 2D Gaussian blob representation

To represent the illuminated blobs that comprise defocused images, inspired by [22, 63], we adopt a 2D Gaussian blob representation. Specifically, given an input defocused image, we first fit it with the 2D Gaussian blob representation, where the image is represented with N Gaussian blobs. Intuitively, by representing the defocused image with a set of 2D Gaussian blobs, we encourage each 2D Gaussian blob to model an illuminated blob of the defocused image, thereby allowing us to model the formation process of the defocused image. Next, to obtain the defocused image from the 2D Gaussian blob representation, we develop an efficient *rasterization method* to render the image formed by these 2D Gaussian blobs, which facilitates the optimization process of representing the input image with 2D Gaussian blobs. In this subsection, we first describe our 2D Gaussian blob representation, followed by the rasterization method used for rendering these 2D Gaussian blobs.

2D Gaussian blob representation. Defocused images consist of a stack of illuminated blobs that are blended together, and our goal is to model these illuminated blobs, thereby modeling the formation process of the defocused images. Yet, this can be challenging, since illuminated blobs can take various shapes (*e.g.*, oval, polygonal, etc) based on the shape of the aperture [42, 54], as well as various sizes and colors. To address this challenge, we represent the defocused image with N 2D Gaussian blobs [63], where each Gaussian blob contains a set of attributes (*i.e.*, parameters) representing the location (u), size and shape (Σ), color (c), and opacity (o). These parameters enable our Gaussian blobs to effectively model and approximate circular/polygonal/oval shapes of various sizes, while also accounting for individual color and opacity. Below, we detail each attribute and how they are encoded.

Specifically, the distribution represented by each i -th Gaussian blob $G_i(x)$ can be defined as:

$$G_i(x) = \exp\left(-\frac{1}{2}(x - u_i)^\top \Sigma_i^{-1}(x - u_i)\right), \quad (1)$$

where $u_i \in \mathbb{R}^2$ encodes the *position* of the i -th 2D Gaussian blob, *i.e.*, the coordinates of the central point, $\Sigma_i \in \mathbb{R}^{2 \times 2}$ is the covariance matrix that determines the *shape and size* of the Gaussian blob, while $x \in \mathbb{R}^2$ is the variable encoding the location of the queried point on the image. In particular, the covariance matrix Σ_i encodes the rotation and scaling of the Gaussian blob, and needs to remain positive semi-definite during parameter updates to maintain a feasible shape. Therefore, to naturally enforce this constraint during parameter updates, we follow [63] by further decomposing the covariance matrix Σ_i and optimizing the decomposed form instead. The covariance matrix Σ_i is decom-

posed as follows:

$$\Sigma_i = R_i S_i S_i^\top R_i^\top, \text{ where} \\ R_i = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) \end{bmatrix}, \quad S_i = \begin{bmatrix} s_{1,i} & 0 \\ 0 & s_{2,i} \end{bmatrix}. \quad (2)$$

Here, θ_i represents the rotation angle, and $s_{1,i}$ and $s_{2,i}$ are scaling factors along the directions of each eigenvector. Importantly, with this decomposition, we can freely update the parameters $\theta_i, s_{1,i}, s_{2,i}$ while naturally constraining Σ_i to be positive semi-definite. This is because Σ_i is the result of multiplying matrix $R_i S_i$ with itself, thus ensuring that the resulting Σ_i is positive semi-definite.

Additionally, each i -th 2D Gaussian blob has color parameters $c_i \in \mathbb{R}^3$ which are normalized RGB values, where the color of the Gaussian blob at each point x is given by:

$$c_i(x) = c_i \cdot \exp\left(-\frac{1}{2}(x - u_i)^\top \Sigma_i^{-1}(x - u_i)\right). \quad (3)$$

Note that each blob also has an opacity value $o_i \in \mathbb{R}$, which is only applied during rasterization, as discussed below.

Rasterization. Next, we need a way to convert input defocused images to the 2D Gaussian blob representation discussed above. One promising way is to randomly initialize a set of 2D Gaussian blobs, then follow the 3D GS [19] approach to optimize the Gaussian blob parameters to fit the input defocused image. Yet, this approach requires a rasterization technique to render images from the 2D Gaussian blob representation, which should also ideally be differentiable to facilitate convenient end-to-end optimization. To address these issues, we modify the tile rasterizer [19] to develop a differentiable and efficient rasterization method tailored to the 2D Gaussian blobs. We also note that 3D GS involves 3D to 2D projection with EWA [69] estimation, but, as we can directly represent 2D Gaussian blobs efficiently with no projection required, we can achieve much faster rasterization.

Specifically, following the intuition that each defocused image can be seen as being composed of a stack of overlapping illuminated blobs, our rasterization method also regards the rendered image as a blend of layers, with each layer being a 2D Gaussian blob. Therefore, during rasterization, for each pixel in the rendered image, we compute the color of the pixel by blending the effects from all 2D Gaussian blobs, where the color $C(x)$ at a pixel location x of the rendered image is computed as the cumulative sum of these 2D Gaussian blobs' colors weighted by their opacities:

$$C(x) = \sum_{i=1}^N c_i(x) \cdot o_i, \quad (4)$$

where $c_i(x)$ represents the color of the i -th Gaussian blob at pixel location x (calculated using Eq. (3)), and o_i represents

the opacity of the i -th Gaussian blob. With this formulation, the rasterization process only uses differentiable operations, and is thus *fully differentiable*. Besides, our rasterization method is implemented in CUDA to make parallel calculations of pixel colors, thus making it *efficient*.

2D Gaussian blob initializer. Yet, obtaining a good-quality Gaussian blob representation of the defocused image necessitates multiple rounds of rasterization and optimization, which can be time-consuming and may hinder the efficiency of our method in practical applications. To address this issue, inspired by previous works [50, 51], we design an efficient 2D Gaussian blob initializer which adopts a fully convolutional U-Net architecture [43]. The initializer is crafted to perform the transformation from an input defocused image to a high-quality initialization of its 2D Gaussian representation in a single step.

3.2. Deblurring by adjusting Gaussians' attributes

In the previous subsection, we discuss how to represent defocused images using 2D Gaussian blobs. Here, we explore how to effectively deblur these defocused images represented by 2D Gaussian blobs, by adjusting the size, shape and opacity of the 2D Gaussian blobs with our blob deblurrer.

As discussed in Sec. 1, fundamental imaging principles suggest that defocus blur is caused by large illuminated blobs (see Fig. 1). Hence, to deblur defocused images, we need to adjust the size of these large illuminated blobs appropriately. To achieve this, we draw inspiration from [22] and design a blob deblurrer. As shown in Fig. 2, our proposed blob deblurrer takes the stack of 2D Gaussian blobs and their parameters as input, and modifies their *covariance* and *opacity* parameters. Modifying the covariance parameters reduces the size of the large Gaussian blobs (large illuminated blobs), removing the blurriness caused by the divergence of light rays that come from outside the focal plane (see Fig. 1). At the same time, because the luminous flux on the downscaled illuminated blobs should still remain consistent compared to their previous larger size, their brightness will also need to be adjusted. Hence, to allow for these adjustments, our blob deblurrer also adjusts the opacity of the 2D Gaussian blobs. Below, we explain our blob deblurrer in more detail.

Blob deblurrer. Our blob deblurrer, denoted as f_{bd} , is a Transformer model that takes as input the parameters of *all* N 2D Gaussian blobs, where each i -th Gaussian blob includes the parameters: central points u_i , rotation matrix R_i , scaling matrix S_i , color c_i , and opacity o_i .

Specifically, given the all N Gaussian blobs as input, our deblurrer f_{bd} performs a one-time forward pass to produce four outputs $(\Delta\theta_i, \Delta s_{1,i}, \Delta s_{2,i}, \Delta o_i)$ for each of the N Gaussian blobs. Then, these outputs are used to update the Gaussian parameters, as follows:

$$\Delta\phi_i = \text{clamp}(\Delta\theta_i, \min = 0, \max = \pi), \quad (5)$$

$$R'_i = \begin{bmatrix} \cos(\theta_i + \Delta\phi_i) & -\sin(\theta_i + \Delta\phi_i) \\ \sin(\theta_i + \Delta\phi_i) & \cos(\theta_i + \Delta\phi_i) \end{bmatrix}, \quad (6)$$

$$S'_i = \begin{bmatrix} s'_{1,i} & 0 \\ 0 & s'_{2,i} \end{bmatrix} \quad (7)$$

$$\text{where } \begin{cases} s'_{1,i} = \min(s_{1,i} - \text{ReLU}(\Delta s_{1,i}), 0.1), \\ s'_{2,i} = \min(s_{2,i} - \text{ReLU}(\Delta s_{2,i}), 0.1), \end{cases}$$

$$o'_i = \text{clamp}((o_i + \Delta o_i), \min = 0, \max = 1). \quad (8)$$

Here, the outputs R'_i , S'_i , and o'_i represent the attributes of the modified 2D Gaussian blobs. Since our aim is to perform deblurring by reducing the size of the Gaussian blobs, we ensure that the *scaling factors* $s_{1,i}$ and $s_{2,i}$ are decreased by subtracting the non-negative values $\text{ReLU}(\Delta s_{1,i})$ and $\text{ReLU}(\Delta s_{2,i})$ respectively (see Eq. (7)). For improved training stability, we enforce lower bounds on $s_{1,i} - \text{ReLU}(\Delta s_{1,i})$ and $s_{2,i} - \text{ReLU}(\Delta s_{2,i})$, ensuring they are clamped to values no less than 0.1. We also constrain the value range of the predicted $\Delta\theta_i$ based on the symmetric nature of the elliptical shape (see Eq. (5)), transforming it to $\Delta\phi_i \in [0, \pi]$ before applying $\Delta\phi_i$ to adjust the rotation matrix (see Eq. (6)). Besides, the term $o_i + \Delta o_i$ in Eq. (8) is clamped to the range $[0, 1]$.

3.3. Depth-based regularization loss

As discussed in Secs. 3.1 and 3.2, through our novel perspective, we are able to perform deblurring according to fundamental imaging principles by representing defocused images using 2D Gaussian blobs. However, the conversion of images to 2D Gaussian blobs may not always be so clear cut and can be quite ambiguous, with many different combinations to optimize the 2D Gaussian blobs. For instance, ambiguous areas of the image may be modeled with fewer but bigger blobs, or with more but smaller blobs. In these ambiguous cases, the optimization may depend on the initial placement of the Gaussian blobs, where if an area starts with fewer Gaussian blobs, it may tend to continue to use fewer blobs. The ambiguity and inconsistency of the Gaussian blob sizes in such situations may affect the effectiveness of our blob deblurrer in deblurring the images.

To address this issue, we revisit the formation process of the illuminated blobs and observe that the size of each blob is often influenced by the relative distance between the corresponding object (point light source) and the focal plane, *i.e.*, the *depth difference* between them. Thus, to resolve the ambiguity, we propose to leverage depth priors from a pre-trained depth estimator to regularize the Gaussian blob sizes. Below, we first introduce the relationship between the blob sizes and the depth differences, and then explain how to integrate depth priors into our framework.

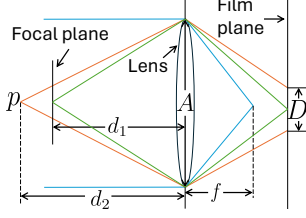


Figure 3. Visualizing the parameters of the imaging process.

It is well known that, the diameter D of the illuminated blob on the film plane caused by a point light source p located outside of the focal plane, is calculated as follows [14]:

$$D = \frac{f \cdot A}{d_2} \cdot \left| \frac{d_2 - d_1}{d_1 - f} \right| = \frac{f \cdot A}{|d_1 - f|} \cdot \left| 1 - \frac{d_1}{d_2} \right|, \quad (9)$$

where f is the lens' focal length, A represents the diameter of the lens' aperture, d_1 and d_2 represent the distances from the focal plane and the point light source p to the lens, respectively. See Fig. 3 for a visualization of these parameters. Since the values of A , f , and d_1 are constant for each defocused image, Eq. (9) can be rewritten as:

$$D \propto \left| 1 - \frac{d_1}{d_2} \right|. \quad (10)$$

As shown in Eq. (10), a linear relationship can be observed between the diameter D of the illuminated blob and a function of the relative depths $|1 - \frac{d_1}{d_2}|$. Next, guided by this relationship, we incorporate depth priors into our proposed framework to regularize the size of the Gaussian blobs.

It is worth noting that our 2D Gaussian blob is represented as an ellipse, not a circle. Therefore, we denote its diameter D using the area-equivalent diameter, which can be interpreted as the diameter of a circle with the same area as the ellipse. Thus,

$$D = 2\sqrt{D_{major} \cdot D_{minor}}, \quad (11)$$

where D_{major} and D_{minor} denote the lengths of the major and minor axes, respectively. In our method, the lengths of the major and minor axes, D_{major} and D_{minor} , are determined by the scaling factors s_1 and s_2 in the scaling matrix S (see Eq. (2)). Hence, following Eq. (10), we can establish the relationship between scaling matrix parameters (s_1 and s_2) and depth (d_1 and d_2) as:

$$\sqrt{s_1 s_2} = K \left| 1 - \frac{d_1}{d_2} \right|. \quad (12)$$

Then, reorganizing this expression leads us to the following equation:

$$\frac{1}{K^2} s_1 s_2 d_2^2 + 2d_1 d_2 - d_1^2 = d_2^2. \quad (13)$$

We note that in Eq. (13), the right side (d_2^2) can be directly computed with the help of a depth estimator. Consequently, after we obtain the depth prior d_2 , they can act as guidance signals for the optimization of the left hand side, which consists of learnable parameters (*i.e.*, s_1, s_2, d_1, K) to be optimized. Thus, the regularization loss L_{reg} can be defined as:

$$L_{reg} = (d_2^2 - [\frac{1}{K^2} s_1 s_2 d_2^2 + 2d_1 d_2 - d_1^2])^2 \quad (14)$$

By applying L_{reg} to the Gaussians and using the depth values (d_2) at their central points as guidance, we can regularize their sizes (*i.e.*, the scaling matrix parameters s_1 and s_2) based on the depth information. We also note that, while s_1 and s_2 are parameters specific to individual Gaussian blobs, d_1 and K are image-level parameters that remain fixed for a single image, meaning they are shared across all Gaussian blobs. As such, we treat d_1 and K as learnable image-level parameters to be updated during training. Overall, by applying this depth-based regularization loss during the Gaussian blob optimization phase, we can regularize the size of Gaussian blobs according to their corresponding depths, which can improve the consistency of their sizes and facilitates better performance by the blob deblurrer.

3.4. Training and testing procedures

Training. Given the input defocused image I_d , we first compute its depth map via a pre-trained depth estimator [40], and then generate the predicted deblurred image I_p through the following steps: **(i)** We input the defocused image I_d along with its depth map into the initializer (described at the end of Sec. 3.1) to obtain an initial 2D Gaussian blob representation Φ (Sec. 3.1). **(ii)** We optimize the image-level parameters (d_1, K) and Φ to bring the rendering I_r of Φ – which is rendered using rasterization (Eq. (4)) – closer to the input defocused image I_d . For the optimization, we apply the loss L , where

$$L = \text{MSE}(I_r, I_d) + \lambda_r \cdot \text{SSIM}(I_r, I_d) + \lambda_{reg} L_{reg}, \quad (15)$$

and λ_r, λ_{reg} are hyperparameters. **(iii)** The obtained 2D Gaussian blob representation Φ , its depth information d_2 , and the optimized image-level parameters (d_1, K) are passed through the blob deblurrer (Sec. 3.2) in a one-time forward pass to perform deblurring and obtain the adjusted 2D Gaussian blob parameters Φ_a (Eqs. (6) to (8)). **(iv)** We render the image through rasterization (Eq. (4)) from Φ_a to get the deblurred image I_p as our output.

Testing. During testing, we are initially given an input defocused image I_d . Then, we follow the steps **(i, ii, iii, iv)** discussed above to obtain the deblurred image I_p .

Table 1. Quantitative comparisons on DPDD and LFDOF datasets. **Best** and **Second Best** results are highlighted.

Model	DPDD			LFDOF		
	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
DPDNet [1]	24.348	0.747	0.277	-	-	-
AIFNet [44]	24.213	0.742	0.309	29.677	0.884	0.202
IFANet [24]	25.366	0.789	0.217	29.787	0.872	0.156
KPAC [49]	25.221	0.774	0.226	28.942	0.857	0.174
GKMNet [36]	25.468	0.789	0.219	29.081	0.867	0.171
MDP [2]	25.347	0.763	0.268	28.069	0.834	0.185
DRBNet [45]	25.485	0.792	0.254	30.253	0.883	0.147
MPRNet [30]	25.730	0.792	0.232	-	-	-
Restormer [59]	25.980	0.811	0.178	-	-	-
INIKNet [38]	26.055	0.803	0.185	30.293	0.886	0.132
NRKNet [37]	26.109	0.810	0.210	30.481	0.884	0.147
GGKMNet [39]	26.272	0.810	0.215	30.552	0.886	0.154
Ours (w/o depth)	26.431	0.827	0.171	30.660	0.889	0.129
Ours	26.651	0.835	0.168	30.885	0.892	0.123

4. Experiments

4.1. Experiment setup

Evaluation datasets and metrics. Following previous works [1, 38, 44], we train our models on two widely-used datasets, DPDD [1] and LFDOF [44]. To evaluate the generalization performance of our method, we also test our method on three additional datasets, RTF [11], RealDOF [24], and CUHK [47]. Since CUHK dataset [47] lacks ground truths, it is only used for qualitative evaluation.

Evaluation metrics. Following previous works [38], we employ three metrics for quantitative evaluation: i) Peak Signal to Noise Ratio (PSNR); ii) Structural SIMilarity index (SSIM); iii) Learned Perceptual Image Patch Similarity (LPIPS) [62] assesses perceptual similarity between images using deep learning features, leveraging pre-trained neural networks to compare image patches, with lower values indicating higher perceptual similarity.

Baselines. We select 12 representative NN-based SIDD methods for comparison, including DPDNet [1], AIFNet [44], IFANet [24], KPAC [49], GKMNet [36], MDP [2], DRBNet [45], Restormer [59], INIKNet [38], NRKNet [37], MPRNet [30], and GGKMNet [39]. We report experiment results for these baselines by referring to their published papers if they are available, otherwise we report the results using their released pretrained models and code.

Implementation details. After obtaining the predicted 2D Gaussian blob parameters by the initializer, we optimize the defocused image’s Gaussian parameters with the Adam optimizer [20] for 600 iterations. For all our experiments, we set λ_r and λ_{reg} to 0.3 and 0.2, respectively. We initialize d_1 as the average depth value from the predicted depth map corresponding to the defocused image, and set the initial value of K to 0.05. Our initializer is an U-Net [43]. Note that the number of Gaussians (N) for each image varies according to the initializer. Following previous SIDD meth-

Table 2. Quantitative comparisons on RealDOF and RTF datasets, with models trained on DPDD and LFDOF. **Best** and **Second Best** results are highlighted.

	Model	RealDOF			RTF		
		PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
Trained on DPDD	DPDNet [1]	22.870	0.670	0.425	23.608	0.591	0.296
	AIFNet [44]	23.093	0.680	0.413	24.041	0.758	0.289
	MDP [2]	23.500	0.681	0.444	24.012	0.738	0.312
	KPAC [49]	23.975	0.762	0.338	24.618	0.777	0.236
	IFANet [24]	24.712	0.748	0.306	24.924	0.801	0.227
	GKMNet [36]	24.254	0.732	0.392	24.970	0.789	0.261
	DRBNet [45]	24.884	0.751	0.376	24.463	0.773	0.311
	MPRNet [30]	24.541	0.736	0.339	24.588	0.788	0.304
	Restormer [59]	25.091	0.762	0.285	24.212	0.821	0.224
	NRKNet [37]	25.148	0.768	0.338	25.931	0.829	0.215
	INIKNet [38]	25.231	0.765	0.287	25.450	0.834	0.215
	GGKMNet [39]	25.355	0.770	0.320	26.012	0.846	0.210
	Ours (w/o depth)	25.567	0.772	0.281	26.114	0.855	0.207
Ours	25.793	0.778	0.278	26.280	0.864	0.204	
Trained on LFDOF	IFANet [24]	22.504	0.669	0.483	26.437	0.838	0.238
	KPAC [49]	22.550	0.671	0.457	25.959	0.803	0.230
	AIFNet [44]	22.623	0.667	0.461	27.552	0.882	0.176
	GKMNet [36]	23.609	0.721	0.408	26.985	0.856	0.246
	MDP [2]	22.726	0.680	0.453	25.580	0.809	0.228
	DRBNet [45]	22.910	0.691	0.437	26.717	0.853	0.200
	NRKNet [37]	23.528	0.716	0.431	28.047	0.889	0.145
	INIKNet [38]	23.810	0.724	0.356	27.401	0.885	0.179
	GGKMNet [39]	24.108	0.735	0.385	28.308	0.905	0.140
	Ours (w/o depth)	24.219	0.750	0.362	28.451	0.909	0.136
	Ours	24.425	0.762	0.355	28.585	0.913	0.132

ods [37, 38], we augment training images by applying random flipping, rotation, and cropping. Tile size for the rasterization is set to 16, which is also used as the block width in CUDA. Our blob deblurrer is a Transformer [52] that is composed of 3 blocks, each containing a feedforward network and a self-attention layer. We train the deblurrer for 20K iterations using the AdamW optimizer [20], starting with an initial learning rate of 5×10^{-2} , which is gradually reduced to 2×10^{-3} . In our work, we employ DPT [40] as the depth estimator, which is a vision transformer that performs full-resolution predictions at different stages of the model, providing fine-grained and globally coherent depth predictions. Notably, we observe that DPT often produces decent depth estimation results even in defocused regions (see Fig. 5 in the main paper and Fig. 1 in the supplementary material).

4.2. Quantitative results

DPDD and LFDOF datasets. In Tab. 1, we present the quantitative results on the DPDD and LFDOF datasets. Besides our full method (Ours), we also run experiments without using the depth information (*i.e.*, the depth-based regularization loss in Sec. 3.3), and report the results as (Ours w/o depth). Even without using depth information, our method already consistently surpasses existing baselines across all metrics on both DPDD and LFDOF datasets. Moreover, when the depth-based regularization

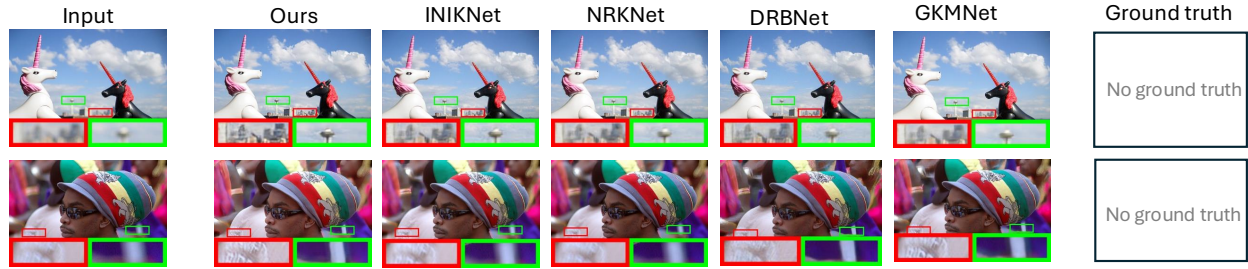


Figure 4. Qualitative comparison using models trained on the DPDD dataset. The visualization results are from the CUHK dataset. Note that CUHK does not provide ground truths.

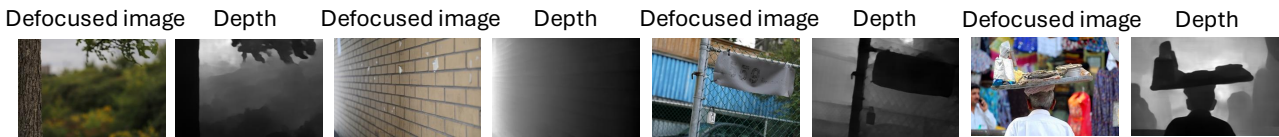


Figure 5. Defocused images from DPDD and CUHK datasets, with their corresponding estimated depth maps.

loss is added, our performance improves even further across all metrics, showing its efficacy. These results demonstrate the efficacy of our novel approach to SIDD, hinting at its huge potential.

RealDOF and RTF datasets. The models trained on DPDD and LFDof datasets are then evaluated on RealDOF and RTF datasets; results are reported in Tab. 2. We observe that, even without depth information, our method already attains SOTA performance across almost all metrics on both RealDOF and RTF. This demonstrates the strong generalization capabilities of our approach. Moreover, using the depth-based regularization loss (Sec. 3.3) leads to even better performance across all metrics. These results show the superior generalizability of our method compared to existing baselines.

4.3. Qualitative results

Qualitative comparison on CUHK dataset. In Fig. 4, we compare the visualization results of our method with the SOTA SIDD methods [36–39, 45] on CUHK dataset. In the second column we present the outputs of our method, and we observe that our method consistently produces clearer and sharper images compared to existing methods, achieving superior visual quality. For instance, compared to existing methods (columns 3-6), our method better recovers image structures and details, restores clearer text, and produces fewer artifacts.

Visualization of the predicted depth map. We provide visualizations to demonstrate the quality of the depth maps generated by the pretrained depth estimation model DPT in Fig. 5. Note that, we perform this evaluation qualitatively since all datasets used in our experiments do not offer ground-truth depth information. From these visualizations,

it can be observed that the depth maps produced by DPT often exhibit good quality, which can provide useful information for our depth-based regularization loss.

4.4. More Experiments

Analysis on adjustment of Gaussian blob size. To verify the efficacy of our method, we further investigate the impact of adjustments to the Gaussian blob size. As the scaling factors s_1 and s_2 in the Gaussian blob representation (Eq. (2)) determine the blob sizes, we first compute the average adjustments in the mean values of s_1 and s_2 before and after applying our method on the DPDD dataset. Specifically, on average across all test images in the DPDD dataset, the values of s_1 and s_2 decrease on average by 21.1% and 18.9% respectively after applying our method for deblurring.

5. Conclusion

In this paper, from a novel perspective, we perform Single image defocus deblurring (SIDD) by adjusting the illuminated blobs that comprise the defocused image. To this end, we adopt a novel 2D Gaussian blob representation and a differentiable rasterization method to obtain the corresponding 2D Gaussian blob parameters of the defocused image. Besides, we also introduce a blob deblurrer to perform defocus deblurring by modifying the size, shape and opacity of the 2D Gaussian blobs in our work. Moreover, we explore a depth-based regularization loss to further enhance the performance of our method. With the proposed framework, our method achieves good results on five widely-used datasets.

References

- [1] Abdullah Abuolaim and Michael S Brown. Defocus deblurring using dual-pixel data. In *Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part X 16*, pages 111–126. Springer, 2020. 3, 7
- [2] Abdullah Abuolaim, Mahmoud Afifi, and Michael S Brown. Improving single-image defocus deblurring: How dual-pixel images help through multi-task learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1231–1239, 2022. 1, 3, 7
- [3] Saeed Anwar, Zeeshan Hayder, and Fatih Porikli. Deblur and deep depth from single defocus image. *Machine vision and applications*, 32(1):34, 2021. 1, 2
- [4] Yuanhao Cai, Yixun Liang, Jiahao Wang, Angtian Wang, Yulun Zhang, Xiaokang Yang, Zongwei Zhou, and Alan Yuille. Radiative gaussian splatting for efficient x-ray novel view synthesis. In *European Conference on Computer Vision*, pages 283–299. Springer, 2024. 2, 3
- [5] Yuanhao Cai, Zihao Xiao, Yixun Liang, Minghan Qin, Yulun Zhang, Xiaokang Yang, Yaoyao Liu, and Alan L Yuille. Hdr-gs: Efficient high dynamic range novel view synthesis at 1000x speed via gaussian splatting. *Advances in Neural Information Processing Systems*, 37:68453–68471, 2024. 2, 3
- [6] Jiale Cao, Hisham Cholakkal, Rao Muhammad Anwer, Fahad Shahbaz Khan, Yanwei Pang, and Ling Shao. D2det: Towards high quality object detection and instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11485–11494, 2020. 1
- [7] Zhe Cao, Lixin Xu, Jin Zhang, Biwen Yang, Kaizheng Chen, and Ruiheng Zhang. DBDB: de-bimodal defocus blur in joint infrared-visible imaging. *Visual Intelligence*, 3(1):7, 2025. 3
- [8] Sunghyun Cho and Seungyong Lee. Convergence analysis of map based blur kernel estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4808–4816, 2017. 1, 2
- [9] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. *Advances in neural information processing systems*, 29, 2016. 1
- [10] Yuanxing Duan, Fangyin Wei, Qiyu Dai, Yuhang He, Wenzheng Chen, and Baoquan Chen. 4d-rotor gaussian splatting: towards efficient novel view synthesis for dynamic scenes. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 2, 3
- [11] Laurent D’Andrès, Jordi Salvador, Axel Kochale, and Sabine Süsstrunk. Non-parametric blur map regression for depth of field extension. *IEEE Transactions on Image Processing*, 25(4):1660–1673, 2016. 1, 2, 7
- [12] DA Fish, AM Brinicombe, ER Pike, and JG Walker. Blind deconvolution by means of the richardson–lucy algorithm. *JOSA A*, 12(1):58–65, 1995. 1, 2, 3
- [13] Lin Geng Foo, Yixuan He, Ajmal Saeed Mian, Hossein Rahmani, Jun Liu, and Christian Theobalt. Avatar concept slider: Controllable editing of concepts in 3d human avatars. *arXiv preprint arXiv:2408.13995*, 2025. 3
- [14] Fernando J Galetto and Guang Deng. Single image deep defocus estimation and its applications. *arXiv preprint arXiv:2107.14443*, 2021. 6
- [15] Jia Gong, Shenyu Ji, Lin Geng Foo, Kang Chen, Hossein Rahmani, and Jun Liu. Laga: Layered 3d avatar generation and customization via gaussian splatting. *arXiv preprint arXiv:2405.12663*, 2024. 3
- [16] Ali Karaali and Claudio Rosito Jung. Edge-based defocus blur estimation with adaptive scale selection. *IEEE Transactions on Image Processing*, 27(3):1126–1137, 2017. 1, 2, 3
- [17] Ali Karaali and Cláudio Rosito Jung. Svbr-net: A non-blind spatially varying defocus blur removal network. In *2022 IEEE International Conference on Image Processing (ICIP)*, pages 566–570. IEEE, 2022. 3
- [18] Ali Karaali, Naomi Harte, and Claudio R Jung. Deep multi-scale feature learning for defocus blur estimation. *IEEE Transactions on Image Processing*, 31:1097–1106, 2022. 1, 2
- [19] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):1–14, 2023. 2, 3, 4
- [20] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 7
- [21] Dilip Krishnan and Rob Fergus. Fast image deconvolution using hyper-laplacian priors. *Advances in neural information processing systems*, 22, 2009. 1, 2, 3
- [22] Byeonghyeon Lee, Howoong Lee, Xiangyu Sun, Usman Ali, and Eunbyung Park. Deblurring 3d gaussian splatting. In *European Conference on Computer Vision*, pages 127–143. Springer, 2024. 2, 3, 4, 5
- [23] Junyong Lee, Sungkil Lee, Sunghyun Cho, and Seungyong Lee. Deep defocus map estimation using domain adaptation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12222–12230, 2019. 2
- [24] Junyong Lee, Hyeongseok Son, Jaesung Rim, Sunghyun Cho, and Seungyong Lee. Iterative filter adaptive network for single image defocus deblurring. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2034–2042, 2021. 1, 3, 7
- [25] Anat Levin, Rob Fergus, Frédo Durand, and William T Freeman. Image and depth from a conventional camera with a coded aperture. *ACM transactions on graphics (TOG)*, 26(3):70–es, 2007. 2, 3
- [26] Yu Li, Dongwei Ren, Xinya Shu, and Wangmeng Zuo. Learning single image defocus deblurring with misaligned training pairs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 1495–1503, 2023. 3
- [27] Tianqi Liu, Guangcong Wang, Shoukang Hu, Liao Shen, Xinyi Ye, Yuhang Zang, Zhiguo Cao, Wei Li, and Ziwei Liu. Mvsgaussian: Fast generalizable gaussian splatting reconstruction from multi-view stereo. In *European Conference on Computer Vision*, pages 37–53. Springer, 2024. 2, 3

- [28] Yu-Qi Liu, Xin Du, Hui-Liang Shen, and Shu-Jie Chen. Estimating generalized gaussian blur kernels for out-of-focus image deblurring. *IEEE Transactions on circuits and systems for video technology*, 31(3):829–843, 2020. 1, 2
- [29] Haoyu Ma, Shaojun Liu, Qingmin Liao, Juncheng Zhang, and Jing-Hao Xue. Defocus image deblurring network with defocus map estimation as auxiliary task. *IEEE Transactions on Image Processing*, 31:216–226, 2021. 3
- [30] Armin Mehri, Parichehr B Ardakani, and Angel D Sappa. Mprnet: Multi-path residual network for lightweight image super resolution. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2704–2713, 2021. 7
- [31] Yuesong Nan and Hui Ji. Deep learning for handling kernel/model uncertainty in image deconvolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2388–2397, 2020. 2
- [32] Saqib Nazir, Lorenzo Vaquero, Manuel Mucientes, Víctor M Brea, and Daniela Coltuc. Depth estimation and image restoration by deep learning from defocused images. *IEEE Transactions on Computational Imaging*, 2023. 3
- [33] Simon Niedermayr, Josef Stumpffegger, and Rüdiger Westermann. Compressed 3d gaussian splatting for accelerated novel view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10349–10358, 2024. 2, 3
- [34] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1520–1528, 2015. 1
- [35] Jinsun Park, Yu-Wing Tai, Donghyeon Cho, and In So Kweon. A unified approach of multi-scale deep and hand-crafted features for defocus estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1736–1745, 2017. 2, 3
- [36] Yuhui Quan, Zicong Wu, and Hui Ji. Gaussian kernel mixture network for single image defocus deblurring. *Advances in Neural Information Processing Systems*, 34:20812–20824, 2021. 1, 3, 7, 8
- [37] Yuhui Quan, Zicong Wu, and Hui Ji. Neumann network with recursive kernels for single image defocus deblurring. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5754–5763, 2023. 1, 2, 3, 7
- [38] Yuhui Quan, Xin Yao, and Hui Ji. Single image defocus deblurring via implicit neural inverse kernels. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12600–12610, 2023. 1, 3, 7
- [39] Yuhui Quan, Zicong Wu, Ruotao Xu, and Hui Ji. Deep single image defocus deblurring via gaussian kernel mixture learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024. 1, 2, 3, 7, 8
- [40] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12179–12188, 2021. 6, 7
- [41] Wenqi Ren, Jiawei Zhang, Lin Ma, Jinshan Pan, Xiaochun Cao, Wangmeng Zuo, Wei Liu, and Ming-Hsuan Yang. Deep non-blind deconvolution via generalized low-rank approximation. *Advances in neural information processing systems*, 31, 2018. 2
- [42] Ken Rockwell. Practical design considerations for modern photographic optics. In *Current Developments in Lens Design and Optical Engineering X*, pages 64–70. SPIE, 2009. 2, 4
- [43] 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. 5, 7
- [44] Lingyan Ruan, Bin Chen, Jizhou Li, and Miu-Ling Lam. Aifnet: All-in-focus image restoration network using a light field-based dataset. *IEEE Transactions on Computational Imaging*, 7:675–688, 2021. 3, 7
- [45] Lingyan Ruan, Bin Chen, Jizhou Li, and Miuling Lam. Learning to deblur using light field generated and real defocus images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16304–16313, 2022. 1, 2, 3, 7, 8
- [46] Qihong Shen, Zike Wu, Xuanyu Yi, Pan Zhou, Hanwang Zhang, Shuicheng Yan, and Xinchao Wang. Gamba: Marry gaussian splatting with mamba for single-view 3d reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025. 2, 3
- [47] Jianping Shi, Li Xu, and Jiaya Jia. Discriminative blur detection features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2965–2972, 2014. 7
- [48] Jianping Shi, Li Xu, and Jiaya Jia. Just noticeable defocus blur detection and estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 657–665, 2015. 2, 3
- [49] Hyeonseok Son, Junyong Lee, Sunghyun Cho, and Seungyong Lee. Single image defocus deblurring using kernel-sharing parallel atrous convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2642–2650, 2021. 1, 2, 3, 7
- [50] Stanislaw Szymanowicz, Christian Rupprecht, and Andrea Vedaldi. Splatter image: Ultra-fast single-view 3d reconstruction. *arXiv preprint arXiv:2312.13150*, 2023. 2, 3, 5
- [51] Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. *arXiv preprint arXiv:2402.05054*, 2024. 2, 3, 5
- [52] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017. 7
- [53] Li Wang, Dong Li, Yousong Zhu, Lu Tian, and Yi Shan. Dual super-resolution learning for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3774–3783, 2020. 1
- [54] 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, 4
- [55] Guodong Xu, Yuhui Quan, and Hui Ji. Estimating defocus blur via rank of local patches. In *Proceedings of the IEEE*

- international conference on computer vision*, pages 5371–5379, 2017. 2
- [56] Xiaodong Yang, Weixing Xie, Sen Peng, Yihang Fu, Wentao Fan, Baorong Yang, and Xiao Dong. 4d gaussian splatting for high-fidelity dynamic reconstruction of single-view scenes. *Neurocomputing*, page 130262, 2025. 2, 3
- [57] Lu Yuan, Jian Sun, Long Quan, and Heung-Yeung Shum. Image deblurring with blurred/noisy image pairs. In *ACM SIGGRAPH 2007 papers*, pages 1–es. 2007. 3
- [58] Lu Yuan, Jian Sun, Long Quan, and Heung-Yeung Shum. Progressive inter-scale and intra-scale non-blind image deconvolution. *Acm Transactions on Graphics (TOG)*, 27(3): 1–10, 2008. 2
- [59] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Restormer: Efficient transformer for high-resolution image restoration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5728–5739, 2022. 7
- [60] Anmei Zhang and Jian Sun. Joint depth and defocus estimation from a single image using physical consistency. *IEEE Transactions on Image Processing*, 30:3419–3433, 2021. 1, 2
- [61] Jie Zhang and Wanming Zhai. Blind attention geometric restraint neural network for single image dynamic/defocus deblurring. *IEEE Transactions on Neural Networks and Learning Systems*, 2022. 3
- [62] 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. 7
- [63] Xinjie Zhang, Xingtong Ge, Tongda Xu, Dailan He, Yan Wang, Hongwei Qin, Guo Lu, Jing Geng, and Jun Zhang. Gaussianimage: 1000 fps image representation and compression by 2d gaussian splatting. *arXiv preprint arXiv:2403.08551*, 2024. 2, 3, 4
- [64] Wenda Zhao, Fan Zhao, Dong Wang, and Huchuan Lu. Defocus blur detection via multi-stream bottom-top-bottom fully convolutional network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3080–3088, 2018. 2
- [65] Wenda Zhao, Fan Zhao, Dong Wang, and Huchuan Lu. Defocus blur detection via multi-stream bottom-top-bottom network. *IEEE transactions on pattern analysis and machine intelligence*, 42(8):1884–1897, 2019. 2
- [66] Wenda Zhao, Fei Wei, You He, and Huchuan Lu. United defocus blur detection and deblurring via adversarial promoting learning. In *European Conference on Computer Vision*, pages 569–586. Springer, 2022. 3
- [67] Shunyuan Zheng, Boyao Zhou, Ruizhi Shao, Boning Liu, Shengping Zhang, Liqiang Nie, and Yebin Liu. Gps-gaussian: Generalizable pixel-wise 3d gaussian splatting for real-time human novel view synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19680–19690, 2024. 2, 3
- [68] Zi-Xin Zou, Zhipeng Yu, Yuan-Chen Guo, Yangguang Li, Ding Liang, Yan-Pei Cao, and Song-Hai Zhang. Triplane meets gaussian splatting: Fast and generalizable single-view 3d reconstruction with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10324–10335, 2024. 2, 3
- [69] Matthias Zwicker, Hanspeter Pfister, Jeroen Van Baar, and Markus Gross. Ewa splatting. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):223–238, 2002. 4