arXiv:2107.14443v1 [eess.IV] 30 Jul 2021

SINGLE IMAGE DEEP DEFOCUS ESTIMATION AND ITS APPLICATIONS

A PREPRINT

Fernando J. Galetto*
Department of Engineering
La Trobe University
Bundoora, VIC 3086, Australia
f.galetto@latrobe.edu.au

Guang Deng
Department of Engineering
La Trobe University
Bundoora, VIC 3086, Australia
d.deng@latrobe.edu.au

August 2, 2021

ABSTRACT

The depth information is useful in many image processing applications. However, since taking a picture is a process of projection of a 3D scene onto a 2D imaging sensor, the depth information is embedded in the image. Extracting the depth information from the image is a challenging task. A guiding principle is that the level of blurriness due to defocus is related to the distance between the object and the focal plane. Based on this principle and the widely used assumption that Gaussian blur is a good model for defocus blur, we formulate the problem of estimating the spatially varying defocus blurriness as a Gaussian blur classification problem. We solved the problem by training a deep neural network to classify image patches into one of the 20 levels of blurriness. We have created a dataset of more than 500000 image patches of size 32x32 which are used to train and test several well-known network models. We find that MobileNetV2 is suitable for this application due to its low memory requirement and high accuracy. The trained model is used to determine the patch blurriness which is then refined by applying an iterative weighted guided filter. The result is a defocus map that carries the information of the degree of blurriness for each pixel. We compare the proposed method with state-of-the-art techniques and we demonstrate its successful applications in adaptive image enhancement, defocus magnification, and multi-focus image fusion.

***K*eywords** Defocus blur estimation, CNN, adaptive image enhancement, shallow depth of field, multi focus fusion

1 Introduction

An image captured by a camera is the projection of a 3-D scene onto a 2-D plane. When an object in a scene is outside the focal plane, it is blurred due to defocus. A larger distance from the focal plane leads to a higher level of defocus resulting in more blurriness of the object [1]. The defocus blur is one of the simplest and efficient depth cues used in photography. It allows the viewer to have a rich appreciation of the 3D space [2]. In computer vision, the defocus blur is used in a wide range of applications such as deblurring [3–5], blur magnification [6], image quality assessment [7, 8], image sharpening [9, 10] and depth estimation [11–14].

Defocus blur estimation has been actively studied in the literature. Some estimation methods used multiple images or special hardware [15–17] while others methods rely on a single image. In this paper, we focus on single image defocus blur estimation methods. A common approach is to model the defocus blur as a linear filter with the impulse response of a Gaussian function. The defocus estimation problem becomes finding the standard deviation σ of the Gaussian kernel for each pixel in the image. The gradient information is usually employed to estimate σ. Tai et al. [18] uses the relationship between the gradient and the local contrast to estimate an initial defocus map which is refined using a Markov Random Field. Others such as [19–21] generate a coarse map using a ratio of gradient magnitudes at the edges that are later refined using a matting algorithm [22, 23] or guided filter [24]. These methods produce impressive

*Corresponding author.

results for some images, however, they can not differentiate between close edges [20]. A statistical method [3] was used to estimate the filter kernel based on solving a SURE minimization problem. This method does not rely on the edge information.

The defocus map can also be obtained by analyzing the image in the frequency domain [25] and [26] which determines the likelihood of a group of pixels being blurred with a certain kernel. Shi et al. [13, 27] proposed two methods to estimate the just noticeable blur level present in natural images by using dictionaries. However, these methods can not handle high degrees of spatially varying defocus blur.

In the past few years, we have witnessed more and more successful applications of deep neural networks (DNN) such as image classification [28, 29], object detection [30], semantic segmentation [31], image restoration [32, 33] and image super-resolution [34] to name a few. The use of DNNs for defocus blur estimation or detection has also been studied. In [9], a CNN is used to estimate the optimum values for the grid wrapping algorithm (GWIS) [35] for image sharpening. The CNN is trained using blurred patches to estimate the optimum parameter based on blurriness to achieve the best sharpening result. In [36–40] a CNN architecture is proposed for end to end defocus map estimation, these networks were trained using either natural or synthetic images labelled at a pixel level to segment focused regions on the image, producing outstanding results. Li et al. [41] on the other hand, proposed a CNN-based method to estimate spatially varying defocus blur on a single image, using synthetic data to train the network and domain transfer to bridge the gap between real and synthetic images.

Inspired by these works, the motivation of this study is to explore the application of deep neural networks in solving the challenging problem of estimating the defocus map from a single image. The main contribution of this work is that, unlike previous works [36–40], we treat the blurriness estimation as a self-supervised multi-class classification problem which is solved by training a CNN to classify a patch of the input image into one of the 20 levels of blurriness. The output of the CNN is a patch-based estimation of blurriness. To obtain an estimate of a pixel-based blurriness, we use an iterative weighted guided filter to perform the refinement which generates the defocus map.

Another main contribution of this work is the development of three algorithms based on the defocus map which carries the blurriness information for each pixel. Since the blurriness is related to the distance between an object and the focal plane, the blur map provides useful information about the depth which is used in the following applications:

- **Adaptive image enhancement:** The defocus map is used to control the level of enhancement in the image to reduce halos and artifacts due to over enhancement of high contrast regions and the enhancement of heavily blur regions.
- **Defocus magnification or shallow depth of field:** The defocus map is non-linearly transformed to estimate the weights for the combination of the sharpened and blurred images to create a shallow depth of field effect.
- **Multi focus image fusion:** The defocus map is used to generate a decision map to blend partially in-focus images to form an all-in-focus image.

The organization of this paper is summarized as follows. In section 2 we briefly review the modelling of defocus blur as a Gaussian blur process. We then formulate the problem of defocus estimation as a classification of spatially varying Gaussian blur. In section 3 we present the details of the proposed method including dataset creation, model testing and selection, the training process to classify the blurriness of a patch, a sliding window algorithm for transferring patch-based blurriness information to the pixel-based format, and a refinement process that produces final pixel-based defocus map. In section IV, we present a comparison of the proposed method with 7 other methods for defocus estimation. In section 5 we present 3 algorithms to demonstrate the usefulness of the defocus map on 3 typical computational photography applications: adaptive image enhancement, generating effect of shallow depth of field, and fusion of images which are focused on different parts of the scene. We present experimental results and comparisons for qualitative and quantitative analysis. We summarize the main ideas and contributions of this work in Section 6. The source code and dataset is publicly available at **https://github.com/fergaletto/deep_defocus_classification**.

2 Modelling of defocus and problem formulation

In this section, we briefly review the optical model that describes the out of focus and show how it can be approximated to a Gaussian filter [42–45]. Although in a practical optical system, the lens usually has more than one moving lens element to correct and fix the focal plane, Fig. 1 allows us to understand the two main reasons for the presence of defocus blur in the observed image: (1) limited depth of field and (2) lens aberrations [43]. To model the defocus by a Gaussian filter, we limit our discussion on the limited depth of field property. As shown in Fig. 1, the point p is in perfect focus because it is placed right in the focal plane. The projection of p on the sensor plane is a single point.

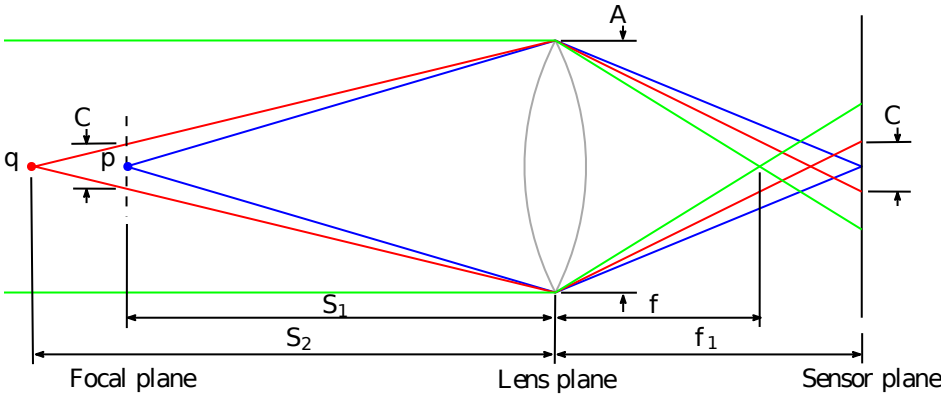


Figure 1: Optical model for defocus blur.

According to the thin lens law, the relationship between the distances is given by:

$$\frac{1}{S_1} + \frac{1}{f_1} = \frac{1}{f} \tag{1}$$

where f is the focal length and f_1 is the distance between the sensor plane and the lens plane. We can see that since the point q is placed in a different plane than the focal plane at a distance S_2 from the lens plane, the projection of point q in the sensor plane is not a single point but a region (or a blurred spot) called circle of confusion. However, the observed blurred spot in the image is not always a circle. The diameter of the circle of confusion for the point q can be calculated as:

$$d = A\frac{f|S_2 - S_1|}{S_2|S_1 - f|} \tag{2}$$

where A is the diameter of the aperture. The diameter d is directly proportional to A and increases when the difference $|S_2 - S_1|$ increases. From a system point of view, we can approximate the lens as a linear shift-invariant system. The point source can be regarded as the impulse input signal and the resulting blurred spot due to defocus blur is thus the impulse response. The shape of the impulse response (the blurred spot) can be modelled as a 2D Gaussian

$$k(x, y; \sigma) \propto \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \tag{3}$$

where σ is the scale parameter proportional to d. Because a scene usually has many objects placed at different distances from the focal plane, these objects will appear to have different degrees of blurriness. Using the linear system model, we can thus model the observed image as

$$J(x, y) = k(x, y; \sigma_{x,y}) * I(x, y) + Z(x, y) \tag{4}$$

where $I(x, y)$ is the desired image where all objects are in focus, $k(x, y; \sigma_{x,y})$ is a spatially varying Gaussian kernel, "*" represents the convolution operations, and $Z(x, y)$ models the possible sensor noise [42–45]. The spatial varying-parameter $\sigma_{x,y}$ is theoretically determined by the distance between the object at location (x, y) and the focal plane.

From the above brief discussion of the Gaussian model for the defocus blur, We can see that (1) the blurriness is caused by the distance of an object from the focal plane, (2) the defocus blur can be modelled as applying a spatial varying Gaussian filter to an image, and (3) the degree of blurriness can implicitly measure by the scale parameter σ of the Gaussian smoothing kernel. In other words, if we know the parameter σ for an image patch, we can infer its blurriness and reveal its distance to the focal plane. We can also infer the relative depth information from a single image. Thus, the problem of inferring depth information or the spatially varying blurriness from a single image reduces to determining the parameter σ of the Gaussian smoothing kernel. In this paper, we proposed a CNN-based approach to first classify an image patch into one of the 20 classes. Each class represents a Gaussian smoothing model of a particular parameter σ. The patch-based blurriness map is refined to obtain a pixel-based blurriness map.

3 Defocus map estimation via CNN based blurriness classification

The proposed method for defocus map estimation has two main building blocks. The first block is a Convolutional Neural Network (CNN) based classifier which takes a patch of 32×32 pixels as input and outputs the blur level of the

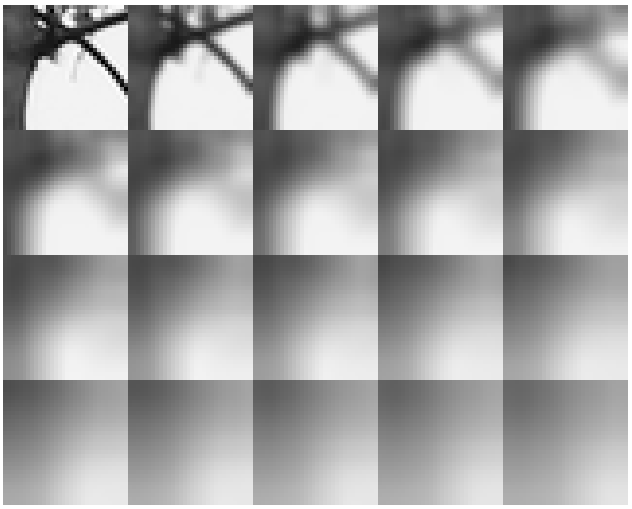


Figure 2: Sample of a training patch with different levels of blurriness.

patch. The classifier is applied to pixels in an image using the sliding window algorithm. The result is an image of patch blurriness. The second block refines the image obtained from the first block to produce an image of pixel blurriness. An additional threshold operation can be applied to produce a binary map for applications that require it.

In this section, we first present the details of the dataset creation, the training of a CNN, and how to use the CNN to estimate the blurriness at a patch level. We then describe the refinement method for producing the pixel-level blur map.

3.1 Blurriness classification

As mentioned earlier, out-of-focus blur is usually modelled as the result of applying a Gaussian low pass filter on an image. The problem of blur estimation is reduced to finding the kernel of the filter [21] which is parameterized by the standard deviation σ of the Gaussian function. In this work, we follow this model. We take a different approach by casting the estimation problem as a classification problem that can be efficiently solved by a CNN.

More specifically, we aim to train a CNN which can classify 20 different levels of blurriness. To train the CNN, we need to prepare the dataset. The dataset consists of image patches of different levels of blurriness. A blurred patch is produced by filtering a sharp/unblurred image patch by a Gaussian filter with a specific σ. Different levels of blurriness are generated by using different values of σ. For an image patch, we generate 19 blurred patches of different blur levels resulting in 20 classes (including the unblurred patch). The 19 levels of blurriness are generated by Gaussian filters of $\sigma = 1, 2, ..., 19$. The unblurred patch can be regarded as being filtered by the filter with $\sigma = 0$ whose impulse response is a unit sample sequence. An example of patches of 20 levels of blurriness can be seen in Figure 2, from top to bottom and left to right the standard deviation σ ranges from 0 to 19. We use the value of σ as a label for the patch.

Table 1: Dataset subsets and samples

Set	**Samples**
Training	392240
Validation	98040
Testing	49920

The size of the dataset depends on the number of unblurred/sharp patches. To find as many as possible sharp patches to generate the dataset for training the CNN, we use the DIV2K [46] database which has 1000 high-resolution images of diverse contents. To find sharp image patches from the database, we convert each image in the database to gray-scale and divide them into patches of 32×32 pixels, denoted I_p. We first filter each patch I_p by a 3×3 Gaussian kernel to produce a slightly smoothed patch B_p. The Gaussian filter is used to remove noise in the patch which could lead to a wrong measurement of the sharpness. We use the variance of the Laplacian of a patch as a measure of the sharpness for the pth patch, which is calculated as

$$\phi_p = \frac{1}{N} \sum_q (\nabla B_p(q) - \overline{\nabla B_p})^2 \tag{5}$$

where ∇ is the Laplacian operator, $\overline{\nabla B_p}$ is the mean value of $\nabla B_p(q)$ over the all pixels in the patch, q is the pixel index, and N is the number of pixels in the patch. Patches satisfying $\phi_p > 1000$ have a high level of sharpness and are regarded as in sharp focus and are assigned the blur level 0. Using the above procedure for patch selection and generation, we create a gray-scale image dataset with more than 5×10^5 patches of 20 classes. The dataset is split into 3 sets for training, validation, and testing. Table 1 details the number of patches of each set.

Using this dataset, we train a neural network to classify the blurriness of a 32×32 gray-scale image patch into one of the 20 classes. The development framework is TensorFlow Keras. Because our goal is to train a classifier for blurriness estimation, we do not attempt to develop new neural network models. Instead, we test some well-known models such as MobileNetV2, NASNet and DenseNet and ShufflenetV2 [29, 47–49] to find a model which has good accuracy and is of small size. We have trained these models and presented results in Table 2. We can see that the MobileNetV2 [29] has a relatively higher accuracy for the test data (0.95) and is computationally more efficient with low complexity and low memory requirements (28MB). ShufflenetV2 [49] presented the worst performance among the models, even after we tuned the hyper-parameters. The sub-optimal performance can be attributed to the small patch size for which the ShufflenetV2 is not designed to work with. It is expected that using a larger patch size will lead to an improvement in the accuracy. However, this is at the increase of the computational complexity in the refinement process of the proposed algorithm because of excessive overlaps of pixels in neighbouring patches.

Table 2: Accuracy and loss results on training, validation, and test set after 100 epochs for different models.

Model	Train loss	Train accuracy	val loss	val accuracy	Test loss	Test accuracy	Size
MobileNetV2 [29]	0.0123	0.9956	0.1205	0.9699	0.1610	0.9580	28MB
EfficientNet0 [50]	0.0075	0.9975	0.0944	0.9793	0.0794	0.9798	49MB
NASNetMobile [47]	0.0073	0.9976	0.2717	0.9585	0.2243	0.9582	55MB
DenseNet121 [48]	0.0124	0.9957	0.6755	0.8996	0.5359	0.9050	86MB
ShufflenetV2 [49]	0.1140	0.9563	1.9462	0.6818	1.8099	0.6902	33MB

The trained network has an input array of size (32×32) and output is an integer ranging from 0 to 19 indicating the level of blurriness. We train MobileNetV2 using the following parameters. The batch size is 128. The sparse categorical cross-entropy function is used to calculate the loss. Adam optimization method is used with a learning rate of $\alpha = 0.001$, an exponential decay rates for the 1st and 2nd moment estimates are $\beta_1 = 0.9$ and $\beta_2 = 0.999$ and a stability constant $\hat{\epsilon} = 10^{-8}$ [51]. After 100 epochs, classification accuracy is 95.80% on the test set which has 49920 samples. The accuracy and loss for the validation and training set after each training epoch is shown in Fig. 3.

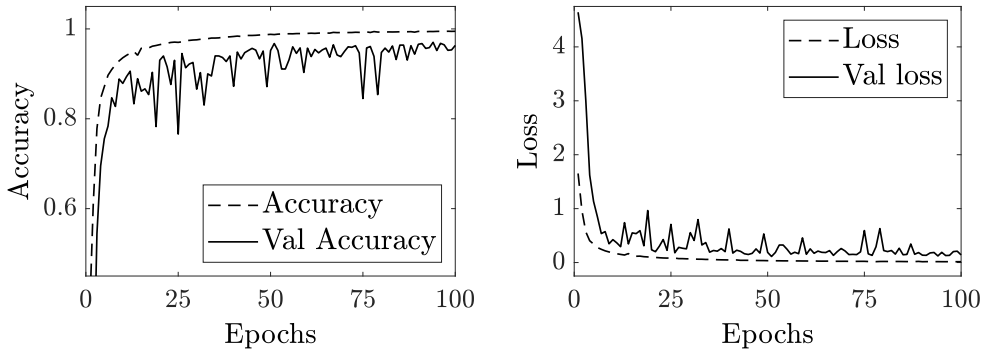


Figure 3: Accuracy and loss over epochs using MobileNetV2 on training and validation set.

3.2 Blur map estimation

The trained neural network predicts the blur level of a patch of 32×32 pixels. This section explains how to obtain the blur map of the whole image at a pixel level using this patch estimation. The blur-map (denoted M) is an image of the same size as the input image (denoted I). A pixel at location q within the pth patch is denoted $I_p(q)$. The image I is divided into overlapping patches of (32×32). The extraction of patches is implemented by using a sliding window method. The amount of overlap depends on the step-size of moving the widow from the current location to the next one. Results presented in this paper are produced with a step size of 8 pixels unless specified. Each patch, e.g., the pth patch is fed to the trained network. The network output, denoted O_p, is assigned to every pixel in the patch. Because of

patch overlapping, one pixel belongs to multiple patches. Specifically, let Ω_q be the set of patches where the pixel $I(q)$ belongs to. The blurriness of the pixel at location q is defined as the average of classification results due to all patches which contain the pixel $I(q)$:

$$M(q) = \frac{1}{|\Omega_q|} \sum_{p \in \Omega_q} O_p \tag{6}$$

where $|\Omega_q|$ is the number of patches which have the pixel $I(q)$. The rationale for performing the average is this. A pixel $I(q)$ belongs to $|\Omega_q|$ patches. Each patch produces a prediction of the blurriness of this pixel. A simple way to aggregate these predictions is by taking the average. In addition, since the classification result O_p is in the interval $[0, 19]$, the blur-map $M(q)$ is also in the same interval.

3.3 Refinement of the defocus map

The defocus map obtained using a patch and sliding window approach needs to be refined to get rid of the undesirable blocking effect and to make the edges follow those in the original image. A well-known tool for the refinement is the matting Laplacian [22] which produces the alpha matting matrix. However, this method is computationally very expensive. Therefore, in this paper, we propose to use a weighted guided filter (WGIF) [52] based algorithm to refine the defocus map. The WGIF has stronger edge-preserving properties and produces fewer halo effects than the original GIF [24]. A new feature in the proposed method is the use of a smoothed version of the original image (through edge-aware filtering) as a guide to producing a sharp defocus map without transferring the texture information.

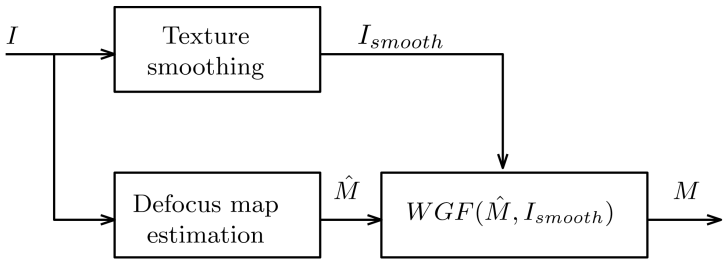
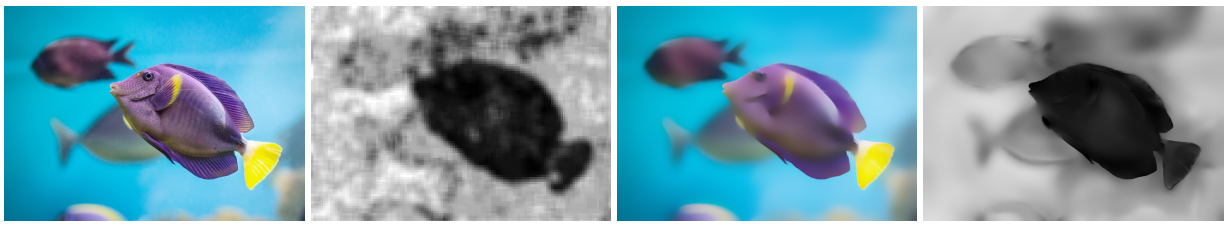


Figure 4: Defocus map refinement process block diagram

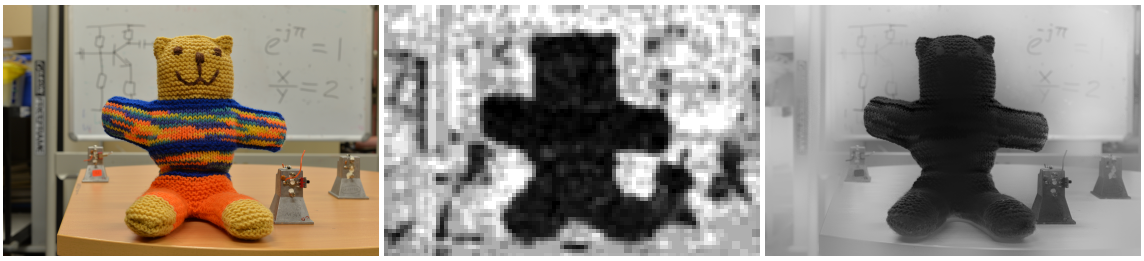
Figure 4 shows the block diagram of the proposed refinement process. First, we perform edge-aware filtering of the original image to smooth out texture information. The filtered image, which retains information of sharp edges is used as a guidance image in the WGIF algorithm to refine the defocus map. We aim to keep only silhouettes of elements in the image which should present an equal level of blurriness preserving large scale edges. It is well known that the WGIF algorithm transfers information from the guidance image to the filtered image. Therefore, texture smoothing is necessary to produce the guidance image such that texture information from the original image will not be transferred to the refined defocus map. Otherwise, the transferred texture information will lead to wrong predictions in pixel blurriness. To produce the guidance image I_{smooth} we use an iterative weighted guided filter [52] which is a variation of the well known guided filter (GIF) [24]. It incorporates an edge-aware weighting factor that reduces/eliminates the halo artifacts while keeping the O(N) complexity of the GIF and still avoids the "reversal gradient" artifacts that the bilateral filter [53] may produce.



(a) Input (b) Blur map (c) Guidance (d) Refined

Figure 5: Defocus blur map refinement. a) Input image, b) Blur map, c) Guidance image, d) Refined blur map.

Figure 5 shows an example of the refinement algorithm, the blur map is shown in Fig. 5b was estimated using the proposed CNN approach with a step size of 8 pixels. It was then refined by using the proposed algorithm with a patch radius $r = 16$, regularization parameter $\epsilon = 0.005$, and the number of iterations $N_{iter} = 7$. The guidance image is shown in Fig 5c and the refinement result is shown in Fig 5d. Only the fish's silhouette has been transferred to the refined defocus map. The darker area represents the area in sharp focus, while the brighter area represents the area of more out of focus. We can see that the refined defocus map accurately represents objects in focus such as the fish in the foreground and blurred objects such as fishes in the background.



(a) Input image (b) Blur map (c) Refined blur map

(d) Input image (e) Blur map (f) Refined blur map

Figure 6: Defocus blur map refinement, a) Original image, b) Blur map estimation, c) Refined blur map.

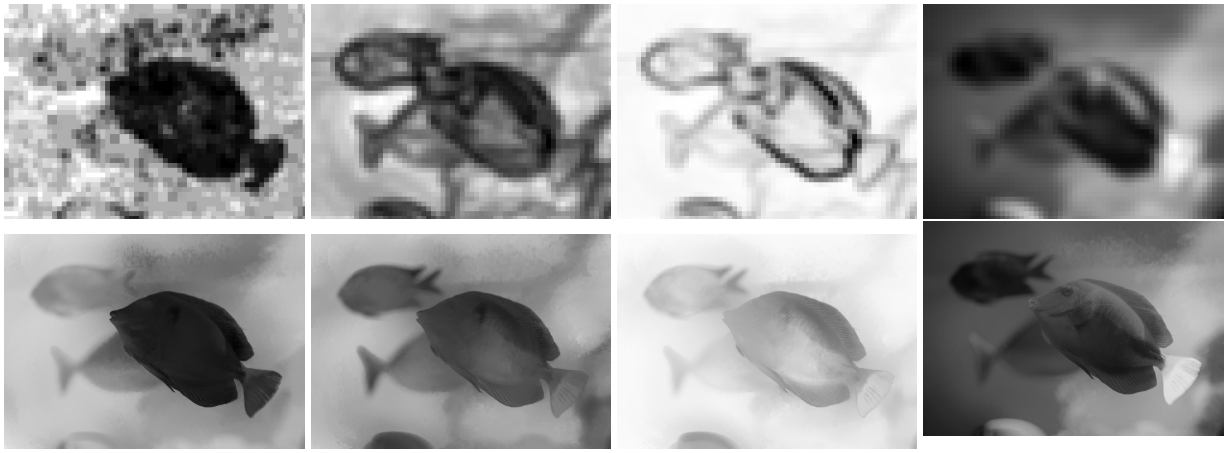
Fig. 6 presents two more examples which are produced by using a step size of 16 pixels to estimate the blur maps. The refinement method clearly removes the blocking effect and wrong predictions in the blur map. Edge consistency is achieved by successfully transferring the edge information from the original image to the result.

4 Comparison of defocus estimation algorithms

In this section, we compare the proposed method with other methods in two categories: classical methods such as entropy, standard deviation and variance of the laplacian of the patch, and modern methods including three handcrafted features "Lbp-based segmentation of defocus blur" from Yi et al. [54], "Defocus map estimation from a single image" from Zhuo et al. [20] and "Fast defocus map estimation" (FDM) from Chen et al. [55] and a CNN based method from Lee et al. "Deep Defocus Map Estimation Using Domain Adaptation" (DMNet) [41].

4.1 Comparison with classical methods

We studied three methods: entropy, standard deviation, and variance of Laplacian [56]. The ideas of these three methods are similar. The blurriness of a pixel is measured by a quantity such as the entropy or standard deviation of a patch of pixels centred at that pixel. For the variance of Laplacian, the image is filtered by using the Laplacian operator. The blurriness of a pixel is measured by the variance of a patch of pixels of the filtered image centred at that pixel. For these three methods, we use a patch size of 16×16 to calculate the entropy, standard deviation and variance of the Laplacian. Simulation results are shown in Fig. 7. The proposed CNN-based method classifies the level of defocus properly for both the edges of objects and the featureless patches. On the other hand, the entropy and standard deviation of the patch can only classify adequately the edges and classify all featureless patches as out of focus. Because both methods are trying to capture edge detection rather than blurriness estimation, they failed to capture information that the fish in the foreground is in focus, see e.g., the certain parts of the body of the fish which are in bright area indicating the area is wrongly classified as out of focus by these two methods. We made a similar observation when we compared the results of the CNN-based method with the variance of the Laplacian. For example, the tail of the fish in the foreground is in bright grayscale indicating it is wrongly classified as out of focus by this method.



(a) Proposed method (b) Entropy (c) Standard deviation (d) Variance of the Laplacian [56]

Figure 7: Defocus blur map and refined map using different methods. The blur map obtained using different methods is displayed on the top row and the refined version using the algorithm introduced in section 3.3 is shown in the bottom row. (a) Our method. (b) Entropy. (c) Standard deviation. (d) Variance of the Laplacian.

4.2 Comparison with modern methods

The bird image shown in Fig. 8 presents a different level of defocus blur. Yi's algorithm was implemented using a threshold $T_{LBP} = 0.025$, for each level we use square local regions of size $s_1 = 7, s_2 = 11, s_3 = 15$ and finally we set the weight $\beta = 0.25$ for the multi-scale inference step. Yi's algorithm struggles to segment the bird which is completely in focus. Fig. 8c shows the result of Zhuo's algorithm, to run this algorithm we set an edge threshold $e = 0.01$ and a standard deviation $\sigma = 1$ for the blur detection step, the interpolation was done using $\lambda = 0.001$ and a maximum blur level $M = 10$. Zhuo's method successfully detects that the bird is in focus but a few details from the background are transferred to the defocus map after interpolation. FDM uses Zhuo's method to estimate the blur level on edges but it is interpolated to the rest of the image using SLIC algorithm increasing the speed significantly but in this particular image the focused region is not well defined. For FDM we use 200 super-pixels and a standard deviation $\sigma = 0.75$. DMNet has a very good performance capturing both background and foreground blurriness levels accordingly. The defocus blur map obtained with the proposed algorithm is shown in Fig. 8f, a step size of 32 pixels was used so no overlapping patches were processed with the CNN. Our method can represent the different levels of blurriness in the image. Our refinement method reduces the number of details transferred from the original image to the defocus map, this can be appreciated more clearly in the background.

Another comparison example is shown in Fig. 9, the caption of the figure contains all the parameter settings for each method to reproduce the experiment but they are omitted here for simplicity. We can see that Yi's algorithm performs better on this image compared with Fig.8 capturing and segmenting focused pixels in the foreground. Zhuo and FDM have difficulty removing details in the foreground especially in the toy's eyes region. DMENet produces a sharp and accurate defocus map. Our method assigns the lowest blur level to the foreground allowing us to differentiate the focused region from the rest of the image.

5 Applications

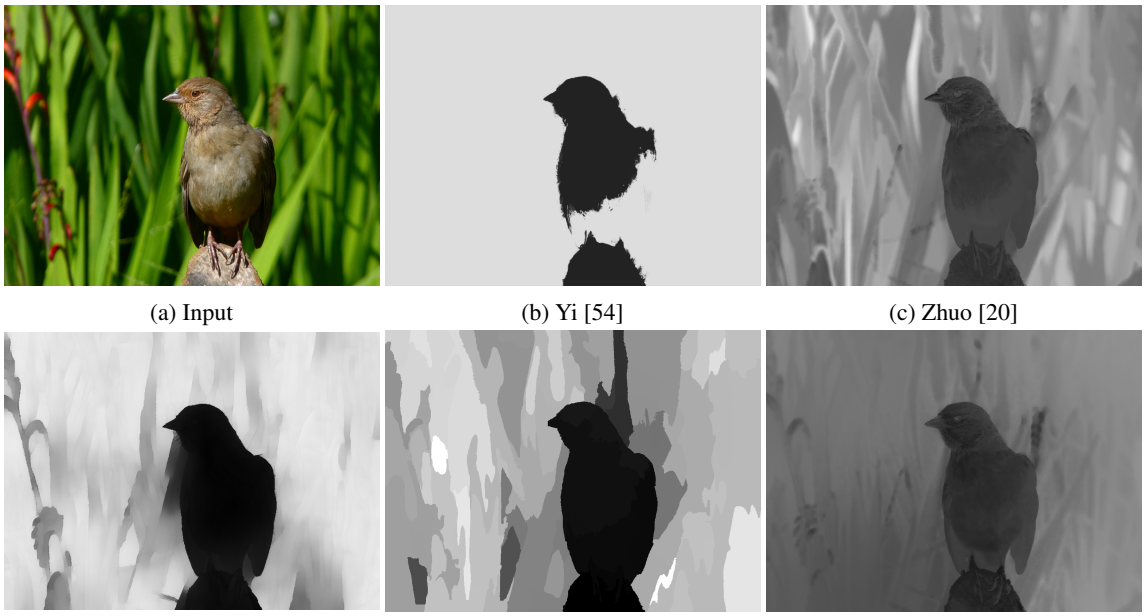
In this section, we demonstrate applications of the proposed defocus map estimation in adaptive image enhancement, creating shallow depth of field, and multi-focus fusion.

5.1 Adaptive enhancement

Unsharp masking (UM) [57] is frequently used for image enhancement. The Unsharp Masking (UM) algorithm can be described as

$$J(q) = I(q) + \lambda Z(q) \tag{7}$$

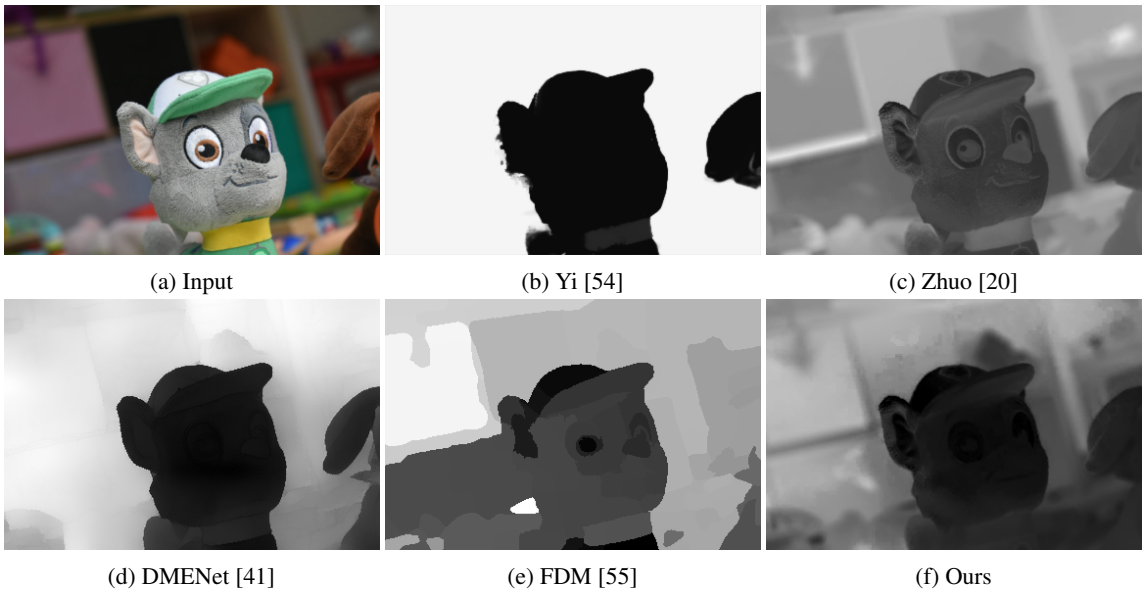
Where λ is used to control the enhancement level, $Z(q) = I(q) - B(q)$, and B is a low pass filtered version of I. For a fixed λ, UM performs a higher degree of enhancement in higher contrast or dynamic regions of the image. To raise the



(a) Input (b) Yi [54] (c) Zhuo [20]

(d) DMNet [41] (e) FDM [55] (f) Ours

Figure 8: Blur map comparisons, a) Input image, b) Yi [54] ($\beta = 0.25, s_1 = 7, s_2 = 11, s_3 = 15, T_{LBP} = 0.025$), c) Zhuo [20] ($e = 0.01, \sigma = 1, \lambda = 0.001, M = 10$), d) DMENet [41], e) FDM [55] ($\sigma = 0.75$) , f) Proposed method.



(a) Input (b) Yi [54] (c) Zhuo [20]

(d) DMENet [41] (e) FDM [55] (f) Ours

Figure 9: Blur map comparisons, a) Input image, b) Yi [54] ($\beta = 0.5, s_1 = 11, s_2 = 15, s_3 = 21, T_{LBP} = 0.005$), c) Zhuo [20], d) DMENet [41], e) FDM [55] ($\sigma = 0.75$), f) Proposed method.

sharpness level of a region with low contrast, the parameter λ has to be set to a relatively higher value, resulting in over enhancement in regions of high contrast. As such, an undesirable non-natural looking result may be produced. Besides, natural images may present different blur and contrast levels in different areas. Having a constant scale-factor λ for the whole image could also lead to the sharpening of regions that are intentionally produced with a high level of blur, e.g., a highly smoothed background to produce a shallow depth of field. It is not desirable to sharpen those areas. Different methods have been proposed to address those problems, including using non-linear or edge-aware filters [58–60] and a pixel adaptive λ that produces a spatially varying enhancement [10, 61].

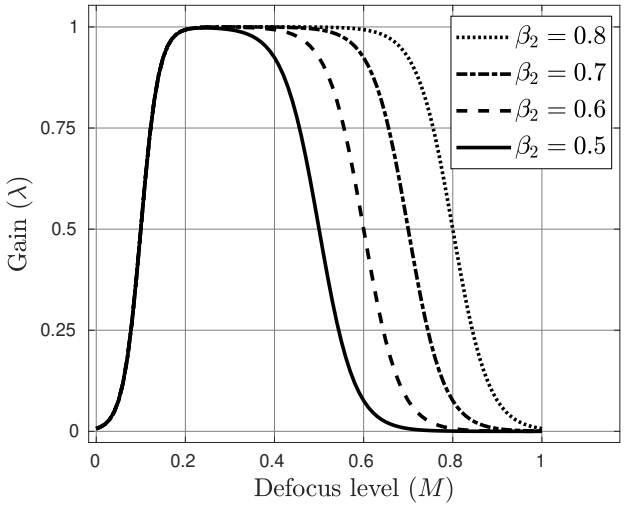


Figure 10: Non linear transformation of the defocus level. For easy visualization, the defocus level is normalized and the parameter settings are $\alpha_1 = 50$, $\beta_1 = 0.1$ and $\alpha_2 = 25$.

We follow the same idea of using a pixel adaptive λ and propose an adaptive unsharp masking algorithm such that the parameter λ for each pixel is a nonlinear function of the refined defocus blur map. The aim of the nonlinear function is to avoid the enhancement of sharp regions as well as in heavily blurred regions. We use the following nonlinear function:

$$\lambda(q) = \lambda_{max} \times \lambda_1(q) \times \lambda_2(q) \tag{8}$$

where λ_{max} is the maximum gain level desired, λ_1 and λ_2 are two sigmoid functions:

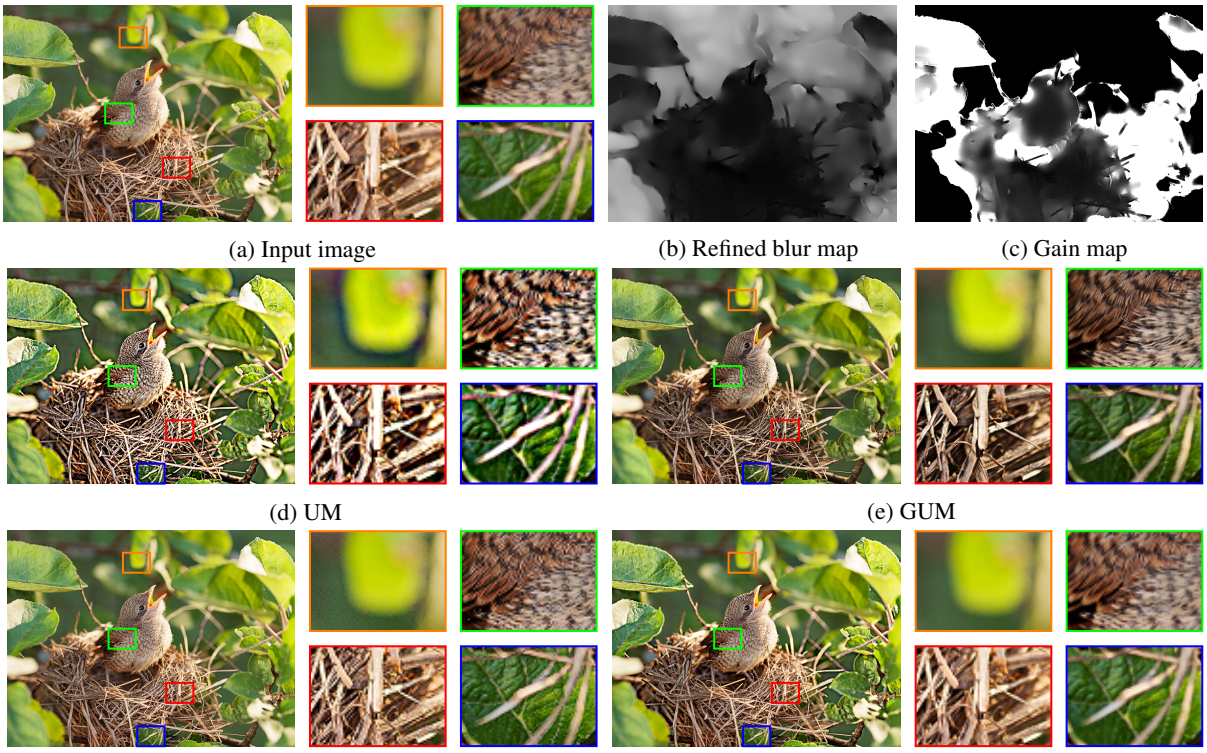
$$\lambda_1(q) = \frac{1}{1 + e^{-\alpha_1(M(q)-\beta_1)}} \tag{9}$$

$$\lambda_2(q) = 1 - \frac{1}{1 + e^{-\alpha_2(M(q)-\beta_2)}} \tag{10}$$

The four parameters α_1, β_1, α_2 and β_2 are determined by the user to have control over the sharpening gain. An example of controlling the gain is shown in Fig. 10. In this example, we set $\alpha_1 = 50$, $\beta_1 = 0.1$ and $\alpha_2 = 25$ are kept fixed while β_2 varies. The scale of the blur map is linearly normalized from the interval $[0, 19]$ to $[0, 1]$ for better visualization and λ_{max} is set to one. β_1 and β_2 indicate the points where the gain is $0.5 \times \lambda_{max}$ and α_1 and α_2 set the growth speed of the curve.

In Fig. 11, we present an example of obtaining the gain map through a non-linear transformation of the defocus map. Fig. 11a shows the input image, enhancing this image is a real challenge because it presents different levels of blurriness, the bird in the foreground is relatively sharp in comparison with the leaves and the background, the background is heavily defocused. Fig. 11b and Fig. 11c are the defocus blur map and the gain map respectively. The blur map was obtained by using a step size of 16 pixels. The gain map was calculated using the nonlinear transformation setting $\alpha_1 = 30, \beta_1 = 0.15, \alpha_2 = 92, \beta_2 = 0.4$ and $\lambda_{max} = 2$. We aim to adaptively enhance the image preventing over-sharpening focused regions and avoiding sharpening blur sections. Fig. 11 also shows the comparison of our method with the classical unsharp masking (UM) and two newer methods contrast adaptive sharpening (CAS)[2] and Generalized Unsharp Masking (GUM) [62]. We can see that applying UM with a fixed global gain $\lambda = 2$ produces an unpleasant sharpening in the background (orange box) and over-sharpening of areas in focus such as the indicated by the red and blur boxes. CAS algorithm was implemented using a gain $\lambda = -0.125$, as can be seen in Fig. 11f this algorithm performs very well with a gain on regions with medium and high defocus blur but still, it can not deal with the blur areas properly producing a noisy result. GUM algorithm (Fig. 11e) handles both background and foreground properly with a gain $\lambda = 7$ and a contrast factor $\kappa = 5 \times 10^{-4}$ but it produces strong halo artifacts when the gain

[2]https://www.amd.com/en/technologies/radeon-software-fidelityfx



(a) Input image (b) Refined blur map (c) Gain map

(d) UM (e) GUM

(f) CAS (g) Proposed method

Figure 11: Adaptive image enhancement comparison, a) Original image, b) Refined blur map (step size of 16 pixels), c) Gain map ($\alpha_1 = 30, \beta_1 = 0.15, \alpha_2 = 92, \beta_2 = 0.4$ and $\lambda_{max} = 2$), d) UM algorithm $\lambda = 2$, e) GUM ($\kappa = 5 \times 10^{-4}, \lambda = 7$), f) CAS ($\lambda = -0.125$), g) Proposed method ($\alpha_1 = 30, \beta_1 = 0.15, \alpha_2 = 92, \beta_2 = 0.4$ and $\lambda_{max} = 2$).

increases. The result of applying the proposed method is shown in Fig. 11g. Our method does not sharpen the heavily blurred background (refer to orange box on Fig. 11g) and prevents over-sharpening of highly focused pixels by limiting the sharpening gain (refer to green, red and blue boxes on Fig. 11g).

5.2 Defocuss magnification

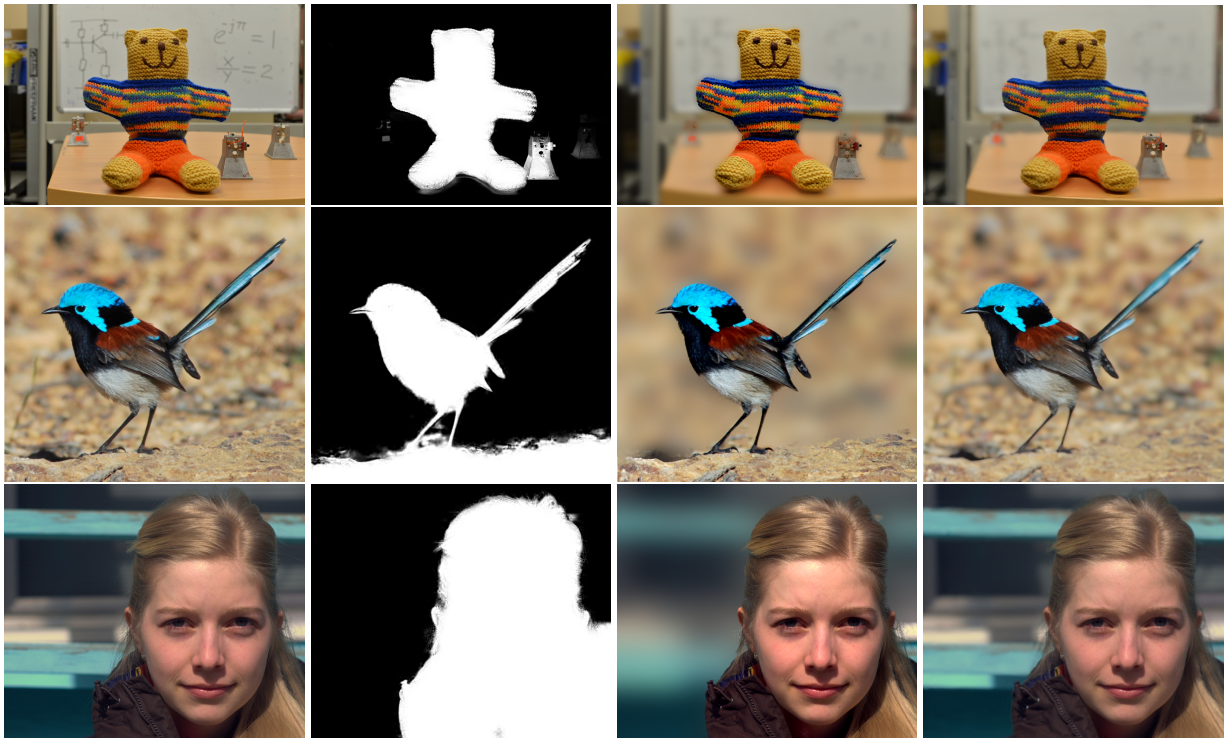
In photography, the shallow depth-of-field (DoF) technique is used to make the main object stand out from the background by producing an image in which in-focus objects are with great details and contrast, while out-of-focus objects are greatly smoothed. The DoF is controlled by a combination of three factors: the f-number, the focal length, and the distance of the camera from in-focus objects. Due to practical limitations such as a lens with a small diameter, it is not always possible to obtain the desired shallow DoF (SDoF) effect when the photo is taken. Enhancement of an image by producing/increasing the SDoF effect is thus a practical image processing problem.

Computer vision algorithms based on blurriness were recently developed to achieve the shallow depth of field effect. For example, an end-to-end CNN [64] selectively increases the level of blurriness in the input image while keeps the edges of the in-focus object intact. The depth information from the RGB-D sensor has also been used to train a CNN to create the SDoF effect from an image that has everything in focus [63]. Also, an object segmentation based algorithm [65] differentiates objects from the background which is further blurred.

A key idea in creating a SDoF effect from a single image is to perform a pixel adaptive weighted combination of a smoothed version denoted $B(p)$ and a sharpened version of the input image denoted $S(p)$,

$$R(q) = W(q)S(q) + (1 - W(q))B(q) \tag{11}$$

We take this approach in this paper by using the defocus map to estimate the weights $W(q)$. The smoothed image B is produced by using one of the edge-aware smoothing filters such as guided filter [24] or one of its weighted versions [66, 67]. The guided filter has two parameters: the radius of the neighbourhood r and the smoothing parameter ϵ. A bigger value of r results in larger structures in the image being smoothed out, while a bigger value of ϵ leads



(a) Input image (b) Weight map (c) Proposed (d) DeepLens [63]

Figure 12: SDoF effect example. a) Original image, b) Weight map, c) SDoF result, d) DeepLens result.

to the overall greater degree of smoothness. To better control the smoothing result, we iterate the guided filter in a self-guidance mode, i.e., the current output is the result of the guided filtering process using the result from the previous iteration as the input and the guidance image. In our experiments presented in Fig. 12, we use $r = 33$, $\epsilon = 128$, and 5 iterations. The sharpened image S is produced by the classical unsharp masking algorithm with a fixed $\lambda = 0.25$ in our experiments.

The weight map W is determined by the nonlinear mapping:

$$W(q) = 1 - \exp\left(-\left|\frac{M(q) - 10}{\sigma}\right|^{\gamma}\right) \tag{12}$$

The justification for the nonlinear transformation and the calculation of σ and γ can be explained as follows. When a pixel is within an in-focus area which corresponds to $M(q) \leq c_0$ (e.g., $c_0 = 1$ is user defined constant) the sharpened image should have greater weight such that the in-focus area is not blurred in the output image. Referring to equation (11), this requires that $W(q) = W_0 \rightarrow 1$. Similarly, when a pixel is within an out-of-focus area which corresponds to $M(q) \geq c_1$ (e.g., $c_1 = 7$ is also a user defined constant), the blurred image should have a greater weight such that in the output image the out-of-focus area is further blurred to create the SDoF effect. This requires $W(q) = W_1 \rightarrow 0$. With these considerations, the two parameters σ and γ can be determined. For example, we can set $W_0 = 0.999$ (for $M(q) = c_0$) and $W_1 = 0.001$ (for $M(q) = c_1$) and calculate the two parameters based on equation (12). In our experiments, the values of c_0 and c_1 were set depending on the image and the defocus levels present application.

Experimental results are shown in Fig. 12 which show that the proposed algorithm can effectively produce the SDoF effect. In the original images shown in Fig. 12a the foreground is in focus while the background is slightly out of focus. After applying the non-linear transformation the weight map splits the image in two well-defined regions. Pixels in focus are with $W(q) \approx 1$ while pixels out of focus are with $W(q) \approx 0$ as shown in Fig. 12b. The resulting SDoF image is shown in Fig. 12c it is evident that the main subject remains sharp while the defocused background is further smoothed. The last column of this figure shows the results from using the DeepLens algorithm. We can see that the proposed algorithm is able to smooth the background to a greater degree.

5.3 Multi-focus image fusion

Most of the lenses have an inherent reduced depth of field, so capturing an image where all the objects in the scene are properly focused can be challenging sometimes. Multi-focus image fusion is a technique that combines two or more partially focused images to produce an all in focus result. A common approach for multi-focus image fusion is through a weighted combination of partially in focus images where the weights define which pixels in the original image are in focus and which ones aren't. Multi focus image fusion is a widely studied topic and several algorithms have been proposed. A recent literature review [68] divided the fusion algorithms into 4 main categories: transform domain, spatial domain, combined transform and spatial domains, and deep learning methods.

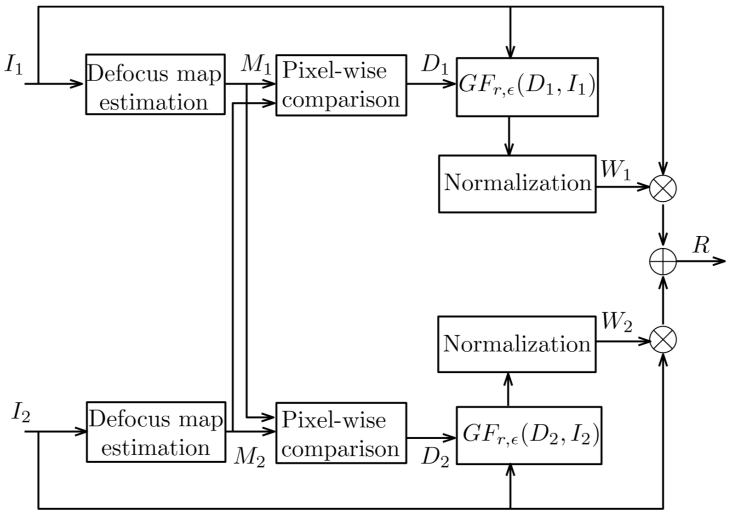


Figure 13: Multi focus image fusion block diagram.

In this paper, we propose a method to fusion 2 or more images using the defocus map to recognize in-focus and out-of-focus regions on each input image. We generate a pixel-based weight map for each input image which allows us to blend them through a linear combination of the weighted images on a pixel level aiming to obtain an all in-focus image. Fig.13 presents the block diagram for the proposed algorithm. The diagram only considers two input images for simplicity but the algorithm can handle multiple input images. The fusion of a set of N images denoted $\{I_n\}_{n=1:N}$ can be described as:

$$R(q) = \sum_{n=1}^{N} W_n(q) I_n(q) \tag{13}$$

where the weights $W_n(q)$ are normalized such that $\sum_{n=1}^{N} W_n(q) = 1$.

To determine the weight for each image I_n, we first use the proposed CNN model to obtain blur map which is refined to produce M_n. We then generate the decision map as follows

$$D_n(q) = \begin{cases} 0, & \text{if} \quad M_n(q) \neq min(M_1(q), ..., M_n(q)) \\ 1, & \text{if} \quad M_n(q) = min(M_1(q), ..., M_n(q)) \end{cases} \tag{14}$$

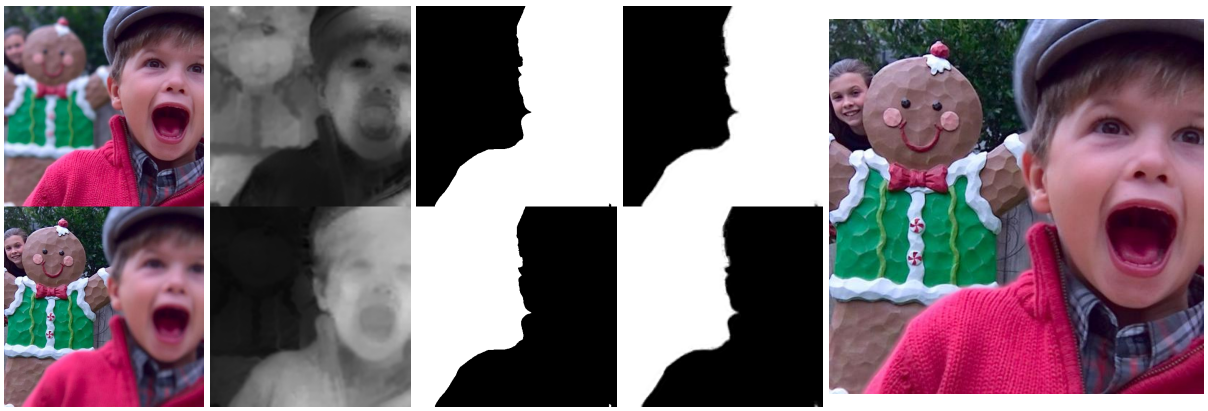
Thus the decision map $D_n(q)$ has a value of 1 only if the pixel of $I_n(q)$ presents the lowest level of blurriness among all N input images. This map allows us to keep from each input image only those pixels that are most in focus.

To avoid halo artifacts around edges we use a guided filter [24] to refine the edges of the decision map,

$$\hat{W}_n = GF_{r,\epsilon}(D_n, I_n) \tag{15}$$

where we use $GF_{r,\epsilon}$ to denote the guided filtering process with D_n being the image to be filtered and I_n being the guidance image. Since we applied the guided filter to each decision map, we must normalize each map to obtain the weights $W_n(q)$ as follows,

$$W_n(q) = \frac{\hat{W}_n(q) + \delta}{\sum_{n=1}^{N} (\hat{W}_n(q) + \delta)} \tag{16}$$



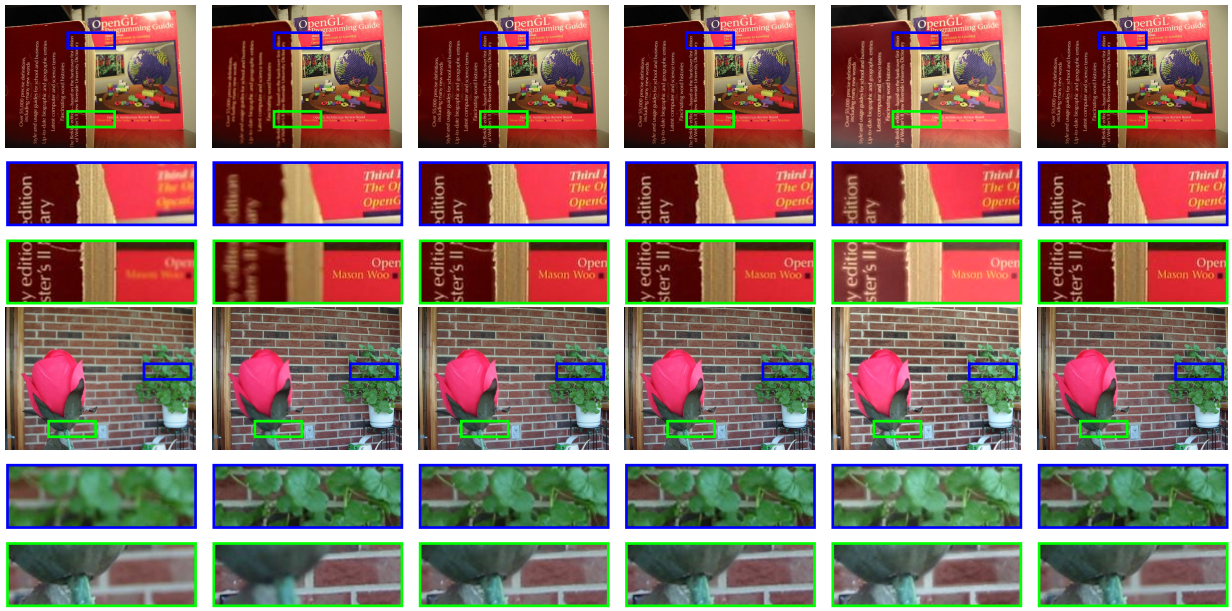
(a) Inputs (b) Defocus maps (c) Decision maps (d) Weights (e) Fusion result

Figure 14: Multi focus image fusion. a) Input images, b) Defocus maps, c) Decision maps, d) Normalized weights ($r = 7, \epsilon = 10^{-3}$), e) Fusion result.

δ represents a small positive regularization term to avoid the division by zero problem and to guaranty that there will be a contribution of each image to the result even when all the weights are zero.

To demonstrate the work-flow of the proposed method, we present the fusion process of two partially in-focus images in Fig.14. Input one has the foreground in focus while the background is out of focus. On the other hand, input two has the background in focus and the foreground out of focus. The defocus maps are presented in Fig. 14b. The resulting decision maps and the normalized weight maps are shown in Fig.14c and 14d, respectively. The fused result is shown in Fig.14e which clearly shows that the resulting image has both foreground and background in focus.

Next, we present a comparison of the proposed method with three state-of-the-art methods: "Guided filter-based multi-focus image fusion through focus region detection" (GFDF) [69], "Image fusion with guided filtering" (GFF) [70] and "Multi-exposure and multi-focus image fusion in gradient domain" (IFGD) [71]. In Fig. 15 we present two fusion results, we applied these algorithms to two pairs of images from the Lytro dataset [72].



(a) Input A (b) Input B (c) GFF [70] (d) GFDF [69] (e) IFGD [71] (f) Proposed method

Figure 15: Multi focus image fusion. a) Input image A, b) Input image B, c) GFF ($r_1 = 45, \epsilon_1 = 0.3, r_2 = 7, \epsilon_2 = 10^{-6}$), d) GFDF($r = 5, \epsilon = 0.3, w = 7$), e) IFGD , f) Ours ($step_{size} = 4, r = 7, \epsilon = 10^{-3}$).

The first example is shown on the first row on Fig. 15, images on column a and b are the inputs to the fusion algorithm. Input A has the foreground in focus and input B has the background in focus. Both foreground and background contain text. The magnified boxes show that our algorithm fuses both images successfully. The resulting image, shown in column f contains sharp text in both foreground and background. The result is comparable with GFF [70] and GFDF [69] in terms of sharpness and edge definition. Our method has better contrast than IFGD [71] producing a more pleasant fusion effect and sharper text.

The second row on Fig.15 shows an example using two input images with a small defocus blur difference (columns a and b), we can see in column f that our algorithm can still fuse the images and obtain an all in focus result. The proposed method produces a similar result as GFF and GFDF but outperforms IFGD.

Quantitative measurement is a complicated task due to the absence of ground truth. We use a set of standard quality metrics that have been used to evaluate the performance of the fusion methods: QG [73] is a gradient-based method that measures the edge information transferred from the input images to the result, QP [74] uses image phase congruence to measure the quality of the fused image, QY [75] employs the structural similarity metric (SSIM) [76] between the inputs and fused image using local windows to evaluate at which level the fusion result contains all the complementary information from the inputs, QCB [77] considers contrast and saliency to evaluate the fusion result, and finally QFMI [78] uses the mutual information to calculate the amount of information transferred to the fused image. Table 3 shows the result of the objective assessment. The proposed method has similar scores as GFDF [69] and is clearly better than GFF [70] and IFGD [71] in these 3 images.

Table 3: Multi focus image fusion quality measurement. A bigger value indicates a better quality.

Image [72]	**Method**	**QG** [73]	**QP** [74]	**QY** [75]	**QCB** [77]	**QFMI** [78]
Cookie	**GFF**	0.68865	0.85206	0.96305	0.73283	0.50355
	GFDF	0.71176	0.86355	0.98848	0.79449	0.5963
	GDIF	0.63084	0.76759	0.85522	0.56627	0.40906
	Ours	0.71277	0.85668	0.98892	0.79378	0.59566
Book	**GFF**	0.61332	0.87643	0.90782	0.74186	0.51189
	GFDF	0.67992	0.8935	0.98554	0.80577	0.63095
	GDIF	0.51231	0.76422	0.70457	0.53092	0.40471
	Ours	0.67762	0.89035	0.98691	0.80535	0.63139
Flower	**GFF**	0.65165	0.79573	0.96029	0.79185	0.53371
	GFDF	0.68329	0.79318	0.98302	0.83116	0.61103
	GDIF	0.6425	0.76802	0.90415	0.6994	0.44966
	Ours	0.68101	0.79275	0.98172	0.82869	0.60643

6 Conclusion

In this paper, we address a challenge problem of estimating the degree of defocus blur from a single image. This work is based on the assumption that the defocus blur can be modelled as spatially varying Gaussian blur. As such the variance of the Gaussian kernel can be used as a proxy for the degree of blurriness. We train a CNN to classify the blurriness of an image patch as one of the 20 classes. Each class label is the variance of a Gaussian kernel. We create a dataset by (1) choosing sharply focused patches of size (32×32) from a high resolution image dataset, and (2) filtering those patches by using Gaussian filters of different standard deviations σ ranging from 1 to 19 to produce 20 classes of blurriness (the original patch is regarded as being filtered by a kernel with $\sigma = 0$). The compiled dataset is used to train a deep CNN model called MobileNetV2 to classify patches onto 20 different levels of blurriness. Using a sliding window algorithm and a weighted guided filter-based refinement, a pixel level defocus map is obtained.

We have tested and validated the proposed method by comparing it with classical and state-of-the-art methods. We have also demonstrated applications of the blur map estimation in adaptive image enhancement, producing shallow of depth of field effects, and multi-focus image fusion. Results have shown that the proposed algorithms lead to competitive performance and are thus effective tools in single image blurriness estimation and a range of applications.

References

[1] A. P. Pentland, "A new sense for depth of field," *IEEE Trans. Pattern Anal. Mach. Intell.*, no. 4, pp. 523–531, 1987.

[2] G. Mather, "Image blur as a pictorial depth cue," *Proceedings of the Royal Society of London. Series B: Biological Sciences*, vol. 263, no. 1367, pp. 169–172, 1996.

[3] F. Xue and T. Blu, "A novel sure-based criterion for parametric psf estimation," *IEEE Trans. Image Process.*, vol. 24, no. 2, pp. 595–607, 2014.

[4] S. H. Chan and T. Q. Nguyen, "Single image spatially variant out-of-focus blur removal," in *Proc. International Conference on Image Processing*. IEEE, 2011, pp. 677–680.

[5] H. Cheong, E. Chae, E. Lee, G. Jo, and J. Paik, "Fast image restoration for spatially varying defocus blur of imaging sensor," *Sensors*, vol. 15, no. 1, pp. 880–898, 2015.

[6] S. Bae and F. Durand, "Defocus magnification," in *Computer graphics forum*, vol. 26, no. 3. Wiley Online Library, 2007, pp. 571–579.

[7] I. van Zyl Marais and W. H. Steyn, "Robust defocus blur identification in the context of blind image quality assessment," *Signal Processing: Image Communication*, vol. 22, no. 10, pp. 833–844, 2007.

[8] F. E. Al-Obaidi, "Image quality assessment for defocused blur images," *American Journal of Signal Processing*, vol. 5, no. 3, pp. 51–55, 2015.

[9] A. Nasonov, A. Krylov, and D. Lyukov, "Image sharpening with blur map estimation using convolutional neural network," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*, vol. 42, no. 2/W12, pp. 161–166, 2019.

[10] W. Ye and K.-K. Ma, "Blurriness-guided unsharp masking," *IEEE Trans. Image Process.*, vol. 27, no. 9, pp. 4465–4477, 2018.

[11] D. Ziou and F. Deschenes, "Depth from defocus estimation in spatial domain," *Computer vision and image understanding*, vol. 81, no. 2, pp. 143–165, 2001.

[12] J. Lin, X. Ji, W. Xu, and Q. Dai, "Absolute depth estimation from a single defocused image," *IEEE Trans. Image Process.*, vol. 22, no. 11, pp. 4545–4550, 2013.

[13] J. Shi, X. Tao, L. Xu, and J. Jia, "Break ames room illusion: depth from general single images," *ACM Trans. Graph.*, vol. 34, no. 6, pp. 1–11, 2015.

[14] H. Tang, S. Cohen, B. Price, S. Schiller, and K. N. Kutulakos, "Depth from defocus in the wild," in *Proc. IEEE Conf. CVPR*, 2017, pp. 2740–2748.

[15] A. Levin, R. Fergus, F. Durand, and W. T. Freeman, "Image and depth from a conventional camera with a coded aperture," *ACM Trans. Graph.*, vol. 26, no. 3, pp. 70–es, 2007.

[16] D. T. Vu, B. Chidester, H. Yang, M. N. Do, and J. Lu, "Efficient hybrid tree-based stereo matching with applications to postcapture image refocusing," *IEEE Trans. Image Process.*, vol. 23, no. 8, pp. 3428–3442, 2014.

[17] C. Zhou, S. Lin, and S. K. Nayar, "Coded aperture pairs for depth from defocus and defocus deblurring," *International Journal of Computer Vision*, vol. 93, no. 1, pp. 53–72, 2011.

[18] Y.-W. Tai and M. S. Brown, "Single image defocus map estimation using local contrast prior," in *Proc. Int. Conf. Image Process., ICIP*. IEEE, 2009, pp. 1797–1800.

[19] X. Zhang, R. Wang, X. Jiang, W. Wang, and W. Gao, "Spatially variant defocus blur map estimation and deblurring from a single image," *Journal of Visual Communication and Image Representation*, vol. 35, pp. 257–264, 2016.

[20] S. Zhuo and T. Sim, "Defocus map estimation from a single image," *Pattern Recognition*, vol. 44, no. 9, pp. 1852–1858, 2011.

[21] H. Hu and G. De Haan, "Low cost robust blur estimator," in *Proc. International Conference on Image Processing*. IEEE, 2006, pp. 617–620.

[22] A. Levin, D. Lischinski, and Y. Weiss, "A closed-form solution to natural image matting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 2, pp. 228–242, 2007.

[23] Q. Chen, D. Li, and C. K. Tang, "KNN matting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 9, pp. 2175–2188, 2013.

[24] K. He, J. Sun, and X. Tang, "Guided image filtering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 6, pp. 1397–1409, 2012.

[25] A. Chakrabarti, T. Zickler, and W. T. Freeman, "Analyzing spatially-varying blur," in *Proc. Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 2512–2519.

[26] X. Zhu, S. Cohen, S. Schiller, and P. Milanfar, "Estimating spatially varying defocus blur from a single image," *IEEE Trans. Image Process.*, vol. 22, no. 12, pp. 4879–4891, 2013.

[27] J. Shi, L. Xu, and J. Jia, "Just noticeable defocus blur detection and estimation," in *Proc. IEEE Conf. CVPR*, 2015, pp. 657–665.

[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Advances in neural information processing systems*, 2012, pp. 1097–1105.

[29] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proc. IEEE Conf. CVPR*, 2018, pp. 4510–4520.

[30] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, 2016.

[31] F. Lateef and Y. Ruichek, "Survey on semantic segmentation using deep learning techniques," *Neurocomputing*, vol. 338, pp. 321–348, 2019.

[32] Z. Jin, M. Z. Iqbal, D. Bobkov, W. Zou, X. Li, and E. Steinbach, "A flexible deep cnn framework for image restoration," *IEEE Trans. Multimedia*, vol. 22, no. 4, pp. 1055–1068, 2019.

[33] S. Anwar and C. Li, "Diving deeper into underwater image enhancement: A survey," *Signal Processing: Image Communication*, vol. 89, p. 115978, 2020.

[34] W. Yang, X. Zhang, Y. Tian, W. Wang, J.-H. Xue, and Q. Liao, "Deep learning for single image super-resolution: A brief review," *IEEE Trans. Multimedia*, vol. 21, no. 12, pp. 3106–3121, 2019.

[35] A. Nasonova and A. Krylov, "Deblurred images post-processing by poisson warping," *IEEE Signal Processing Letters*, vol. 22, no. 4, pp. 417–420, 2014.

[36] K. Ma, H. Fu, T. Liu, Z. Wang, and D. Tao, "Deep Blur Mapping: Exploiting High-Level Semantics by Deep Neural Networks," *IEEE Trans. Image Process.*, vol. 27, no. 10, pp. 5155–5166, 2018.

[37] W. Zhao, F. Zhao, D. Wang, and H. Lu, "Defocus blur detection via multi-stream bottom-top-bottom network," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 8, pp. 1884–1897, 2019.

[38] Z. Jiang, X. Xu, C. Zhang, and C. Zhu, "Multianet: a multi-attention network for defocus blur detection," in *Proc. International Workshop on Multimedia Signal Processing (MMSP)*. IEEE, 2020, pp. 1–6.

[39] C. Tang, X. Liu, S. An, and P. Wang, "Br^{2}net: Defocus blur detection via a bidirectional channel attention residual refining network," *IEEE Trans. Multimedia*, vol. 23, pp. 624–635, 2021.

[40] Y. Zhai, J. Wang, J. Deng, G. Yue, W. Zhang, and C. Tang, "Global context guided hierarchically residual feature refinement network for defocus blur detection," *Signal Processing*, p. 107996, 2021.

[41] J. Lee, S. Lee, S. Cho, and S. Lee, "Deep defocus map estimation using domain adaptation," in *Proc. IEEE Conf. CVPR*, 2019, pp. 12 222–12 230.

[42] T. Sakamoto, "Model for spherical aberration in a single radial gradient-rod lens," *Applied Optics*, vol. 23, no. 11, pp. 1707–1710, 1984.

[43] Y.-W. Tai and M. S. Brown, "Single image defocus map estimation using local contrast prior," in *Proc. Int. Conf. Image Process., ICIP*. IEEE, 2009, pp. 1797–1800.

[44] P. Favaro and S. Soatto, "A geometric approach to shape from defocus," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 3, pp. 406–417, 2005.

[45] A. Kubota and K. Aizawa, "Reconstructing arbitrarily focused images from two differently focused images using linear filters," *IEEE Trans. Image Process.*, vol. 14, no. 11, pp. 1848–1859, 2005.

[46] E. Agustsson and R. Timofte, "Ntire 2017 challenge on single image super-resolution: Dataset and study," in *Proc. IEEE Conf. CVPR*, 2017, pp. 126–135.

[47] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proc. IEEE Conf. CVPR*, 2018, pp. 8697–8710.

[48] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. CVPR*, 2017, pp. 4700–4708.

[49] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "Shufflenet v2: Practical guidelines for efficient cnn architecture design," in *Proc. European conference on computer vision (ECCV)*, 2018, pp. 116–131.

[50] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *arXiv preprint arXiv:1905.11946*, 2019.

[51] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[52] Z. Li, J. Zheng, Z. Zhu, W. Yao, and S. Wu, "Weighted guided image filtering," *IEEE Trans. Image Process.*, vol. 24, no. 1, pp. 120–129, 2014.

[53] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proc. International Conference on Computer Vision*. IEEE, 1998, pp. 839–846.

[54] X. Yi and M. Eramian, "Lbp-based segmentation of defocus blur," *IEEE Trans. Image Process.*, vol. 25, no. 4, pp. 1626–1638, 2016.

[55] D.-J. Chen, H.-T. Chen, and L.-W. Chang, "Fast defocus map estimation," in *Proc. Int. Conf. Image Process., ICIP*. IEEE, 2016, pp. 3962–3966.

[56] J. L. Pech-Pacheco, G. Cristobal, J. Chamorro-Martinez, and J. Fernandez-Valdivia, "Diatom autofocusing in brightfield microscopy: a comparative study," in *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, vol. 3, 2000, pp. 314–317 vol.3.

[57] G. Ramponi, "A cubic unsharp masking technique for contrast enhancement," *Signal Processing*, vol. 67, no. 2, pp. 211–222, 1998.

[58] S. K. Mitra, H. Li, I.-S. Lin, and T.-H. Yu, "A new class of nonlinear filters for image enhancement," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, 1991, pp. 2525–2526.

[59] G. Ramponi, "A cubic unsharp masking technique for contrast enhancement," *Signal Processing*, vol. 67, no. 2, pp. 211–222, 1998.

[60] G. Ramponi, N. K. Strobel, S. K. Mitra, and T.-H. Yu, "Nonlinear unsharp masking methods for image contrast enhancement," *Journal of Electronic Imaging*, vol. 5, no. 3, pp. 353–367, 1996.

[61] A. Polesel, G. Ramponi, and V. J. Mathews, "Image enhancement via adaptive unsharp masking," *IEEE Trans. Image Process.*, vol. 9, no. 3, pp. 505–510, 2000.

[62] G. Deng, "A generalized unsharp masking algorithm," *IEEE Trans. Image Process.*, vol. 20, no. 5, pp. 1249–1261, 2011.

[63] 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.

[64] P. Sakurikar, I. Mehta, and P. Narayanan, "Defocus magnification using conditional adversarial networks," in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2019, pp. 1337–1346.

[65] N. Wadhwa, R. Garg, D. E. Jacobs, B. E. Feldman, N. Kanazawa, R. Carroll, Y. Movshovitz-Attias, J. T. Barron, Y. Pritch, and M. Levoy, "Synthetic depth-of-field with a single-camera mobile phone," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 1–13, 2018.

[66] Z. Li, J. Zheng, Z. Zhu, W. Yao, and S. Wu, "Weighted guided image filtering," *IEEE Trans. Image Process.*, vol. 24, no. 1, pp. 120–129, 2014.

[67] Z. Sun, B. Han, J. Li, J. Zhang, and X. Gao, "Weighted guided image filtering with steering kernel," *IEEE Trans. Image Process.*, vol. 29, pp. 500–508, 2020. [Online]. Available: https://doi.org/10.1109/tip.2019.2928631

[68] Y. Liu, L. Wang, J. Cheng, C. Li, and X. Chen, "Multi-focus image fusion: A Survey of the state of the art," *Inform. Fusion*, vol. 64, pp. 71–91, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1566253520303109

[69] X. Qiu, M. Li, L. Zhang, and X. Yuan, "Guided filter-based multi-focus image fusion through focus region detection," *Signal Processing: Image Communication*, pp. 35–46, 2019.

[70] S. Li, X. Kang, and J. Hu, "Image fusion with guided filtering," *IEEE Trans. Image Process.*, vol. 22, no. 7, pp. 2864–2875, 2013.

[71] S. Paul, I. S. Sevcenco, and P. Agathoklis, "Multi-exposure and multi-focus image fusion in gradient domain," *Journal of Circuits, Systems and Computers*, vol. 25, no. 10, pp. 1–18, 2016.

[72] M. Nejati, S. Samavi, and S. Shirani, "Multi-focus image fusion using dictionary-based sparse representation," *Inform. Fusion*, vol. 25, pp. 72–84, 2015.

[73] C. S. Xydeas and V. S. Petrovic, "Objective pixel-level image fusion performance measure," in *Proc. Sensor Fusion: Architectures, Algorithms, and Applications IV*, vol. 4051. International Society for Optics and Photonics, 2000, pp. 89–98.

[74] J. Zhao, R. Laganiere, and Z. Liu, "Performance assessment of combinative pixel-level image fusion based on an absolute feature measurement," *Int. J. Innov. Comput. I.*, vol. 3, no. 6, pp. 1433–1447, 2007.

[75] C. Yang, J.-Q. Zhang, X.-R. Wang, and X. Liu, "A novel similarity based quality metric for image fusion," *Inform. Fusion*, vol. 9, no. 2, pp. 156–160, 2008.

[76] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, 2004.

[77] Y. Chen and R. S. Blum, "A new automated quality assessment algorithm for image fusion," *Image and Vision Computing*, vol. 27, no. 10, pp. 1421–1432, 2009.

[78] M. Haghighat and M. A. Razian, "Fast-fmi: non-reference image fusion metric," in *Proc. AICT*. IEEE, 2014, pp. 1–3.