The Visual Computer
https://doi.org/10.1007/s00371-022-02609-9

ORIGINAL ARTICLE

Single image defocus map estimation through patch blurriness classification and its applications

Fernando Galetto[1] · Guang Deng[1]

Accepted: 24 June 2022

Abstract
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 solve 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 500,000 image patches of size 32×32 which does not require human labelling. The dataset is 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 and defocus magnification limited to images that present a clear distinction between defocus levels.

Keywords Defocus blur estimation · CNN · Adaptive image enhancement · Shallow depth of field

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 [11–15].

Defocus blur estimation has been actively studied in the literature. Some estimation methods used multiple images [15,16] or special hardware [17–20] 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 in the spatial domain 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. [21] uses the relationship between the gradient and the local contrast to estimate an initial defocus map which is refined by using a Markov Random Field. Other researchers generate a coarse map using a ratio of gradient magnitudes at the edges [22–26]. The map is refined by using a matting algorithm [27,28], guided filter [29] or segmentation-based algorithms [26,30]. Although these methods have demonstrated successful applications for some images, they cannot differentiate between close edges [23]. In [3], a statistical method is proposed to overcome such limitation by formulating the filter kernel estimation problem as a SURE

✉ Fernando Galetto
f.galetto@latrobe.edu.au

Guang Deng
d.deng@latrobe.edu.au

[1] Department of Engineering, La Trobe University, Bundoora, VIC 3086, Australia

Published online: 25 July 2022

minimization problem. This method does not rely on edge information. Yi et all. [31] proposed a sharpness metric based on the distribution of uniform local binary patterns in focus and out-of-focus regions for defocus segmentation.

Another common approach is to determine the defocus map by analyzing the image in the frequency domain [32,33]. The idea is to calculate the likelihood of a group of pixels being blurred with a certain kernel. Shi et al. [13,34] proposed two methods to estimate the just noticeable blur level present in natural images by using dictionaries. However, these methods cannot handle high degrees of spatially varying defocus blur. In [35] the defocus is measured by applying discrete cosine transform (DCT) to the original image and its re-blurred version to finally calculating the ratio of their norms. Based on such information, the image is then segmented into in-focus and out-of-focus regions.

In recent years, we have witnessed more and more successful applications of deep neural networks (DNN) such as image classification [36,37], object detection [38], semantic segmentation [39], image restoration [40,41] and image super-resolution [42] to just name a few. The use of DNNs for defocus blur estimation or detection has also been studied. In [9], a convolutional neural network (CNN) is used to estimate the optimum values for the grid wrapping algorithm (GWIS) [43] 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 [44–49] a CNN architecture is proposed for an end-to-end binary defocus map estimation. These networks are trained using either natural or synthetic images labelled at pixel-level to segment focused regions from blurred regions. But they do not distinguish different levels of defocus. In [50] Park et al. use hand crafted features and a CNN to extract deep features from multi-scale patches to estimate the spatially varying defocus blur on a single image. The features are concatenated and classified through a fully connected layer.

Li et al. [51] also propose a single image CNN-based method for spatially varying defocus blur by using synthetic data to train the network. They also use domain transfer to bridge the gap between real and synthetic data. In two recently published methods [52,53], a patch-based quality map is generated. The quality map is similar to the blur map when the distortion on the image is defocus blur only.

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 [44–49], we treat the blurriness estimation as a 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 dataset used to train the model does not require human labelling since the defocus levels are synthetically generated from natural sharp images. 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 two 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 nonlinearly transformed to estimate the weights for the combination of the sharpened and blurred images to create a shallow depth of field effect.

The organization of this paper is summarized as follows. In Sect. 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 Sect. 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 Sect. 4, we present a comparison of the proposed method with 7 other methods for defocus estimation. In Sect. 5 we present 2 algorithms to demonstrate the usefulness of the defocus map on 2 typical computational photography applications: adaptive image enhancement and generating effect of shallow depth of field. We present experimental results and comparisons for qualitative and quantitative analysis. We summarize the main ideas and contributions of this work in Sect. 6.

2 Modelling of defocus and problem formulation

In this section, we briefly review the optical model of out-of-focus and show how it can be approximated by a Gaussian filter [21,54–56]. Although a practical optical system 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 [21]. To model the defocus by a Gaussian filter, we limit our discussion on the limited depth of field property. As shown in Fig. 1,

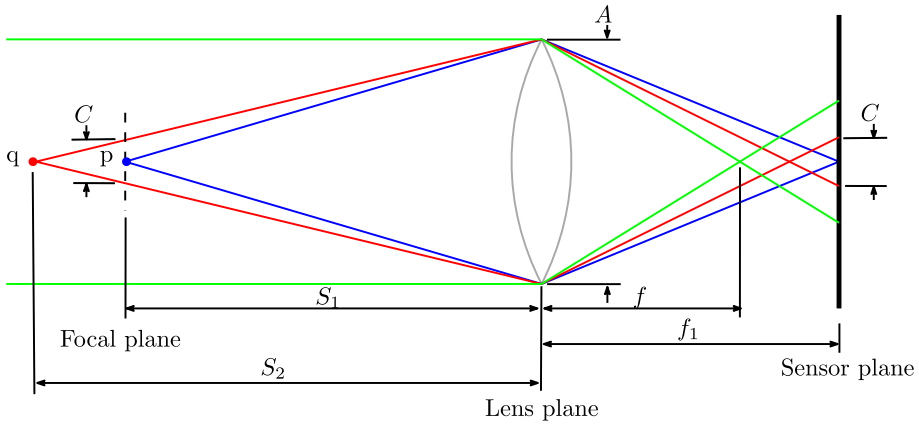


Fig. 1 Optical model for defocus blur. A is the diameter of the aperture. f is the focal length and f_1 is the distance between the sensor plane and the lens plane. S1 and S2 are the distances from the lens plane to the point p and q, respectively. C is the diameter of the circle of confusion for the point q

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.

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. 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, and x, y are spatial variables. 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 [21,54–56]. 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 be implicitly measure by the scale parameter $\sigma_{x,y}$ of the Gaussian smoothing kernel. In other words, if we know the parameter $\sigma_{x,y}$ for an image patch, we can infer its blurriness and reveal its relative distance from the focal plane. Thus, the problem of inferring depth information or the spatially varying blurriness from a single image reduces to determining the parameter $\sigma_{x,y}$ of the Gaussian smoothing kernel. The patch-based blurriness map is refined to obtain a pixel-based blurriness map.

3 Defocus map estimation via CNN-based blurriness classification

The block diagram in Fig. 2 shows the proposed method for defocus map estimation. The first block is a CNN-based classifier which takes a patch of 32 × 32 pixels as input and outputs the blur level of the patch. The classifier is applied to pixels in an image using the sliding window algorithm. The result is an image of patch blurriness. Then a refinement block processes the image obtained 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

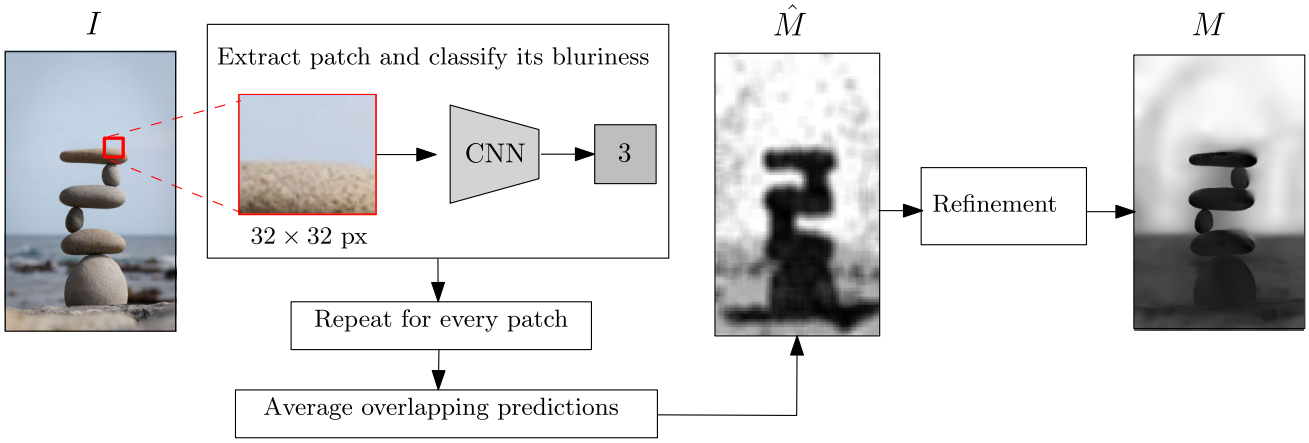


Fig. 2 Block diagram of the proposed defocus map estimation algorithm

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 [24] 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 model, 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 $\sigma = 1, 2, \ldots, 19$. 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 Fig. 3, 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. The justification of why 20 levels of blurriness are selected can be seen in Fig. 3, the difference in blurriness for patches where $\sigma > 15$ is almost imperceptible for the human eye so 20 levels of blurriness is the optimal equilibrium between efficiency and complexity. The linear scale is also arguable but it was here chosen for the sake of simplicity.

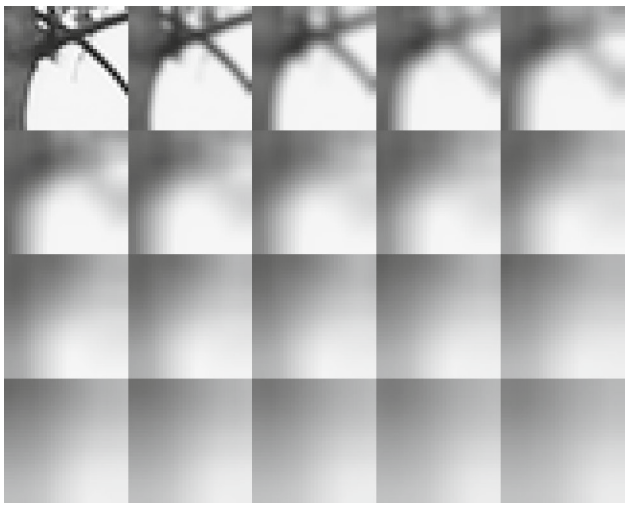


Fig. 3 Sample of a training patch with different levels of blurriness. Top left shows the least blurred while the bottom right shows the most blurred

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 entire DIV2K [57] database which has 800 high-resolution images of diverse contents in the training set and 100 high resolution images in the validation set. 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 [58] as a measure of the sharpness for the pth patch, which is calculated as

Table 1 Number of image patches in the data set

Data set	Patches
Training	392, 240
Validation	98, 040
Testing	49, 920

$$\phi_{\mathrm{p}} = \frac{1}{N}\sum_{q}(\nabla B_{\mathrm{p}}(q) - \overline{\nabla B_{\mathrm{p}}})^2 \tag{5}$$

where ∇ is the Laplacian operator, $\overline{\nabla B_{\mathrm{p}}}$ is the mean value of $\nabla B_{\mathrm{p}}(q)$ over the all pixels in the patch, q is the pixel index, and N is the number of pixels in the patch. Patches satisfying $\phi_{\mathrm{p}} > 1000$ 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. 80% of the patches obtained from the DIV2K training set were used to create the training set while the remaining 20% are used for the validation set. Patches from the DIV2k Validation set are used as the testing set. 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. As such, the trained network has an input array of size (32×32) and the network output is an integer ranging from 0 to 19 indicating the level of blurriness. 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, EfficientNetB0, NASNetMobile, DenseNet121, and ShufflenetV2 [37,59–62] to find a model which has good accuracy and is of small size. We have trained these models and present results in Table 2. We can see that the MobileNetV2 [37] has a relatively higher accuracy for the test data (0.97) and is computationally more efficient with low complexity and low memory requirements (the model size is 28MB). This indicates that it is a more efficient and accurate tool for the estimation of the defocus map. ShufflenetV2 [62] 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. 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.

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 $\alpha = 0.001$, exponential decay rates for the 1st and 2nd moment estimates are $\beta_1 = 0.9$ and $\beta_2 = 0.999$ and a stability constant $\hat{\epsilon} = 10^{-8}$ [63]. After 100 epochs, the classification accuracy is 97.04% on the test set. The accuracy and loss for the validation and training set after each training epoch is shown in Fig. 4.

Table 2 Results of accuracy and loss of training, validation, and test after running 100 epochs for different models

	MobileNetV2 [37]	EfficientNetB0 [59]	NASNetMobile [60]	DenseNet121 [61]	ShufflenetV2 [62]
Train loss	0.0117	0.0075	0.0073	0.0124	0.114
Train accuracy	0.9958	0.9975	0.9976	0.9957	0.9563
val loss	0.1201	0.0944	0.2717	0.6755	1.9462
val accuracy	0.9699	0.9793	0.9585	0.8996	0.6818
Test loss	0.161	0.0794	0.2243	0.5359	1.8099
Test accuracy	*0.9704*	0.9798	0.9582	0.905	0.6902
Model Size	*28MB*	49MB	55MB	86MB	33MB

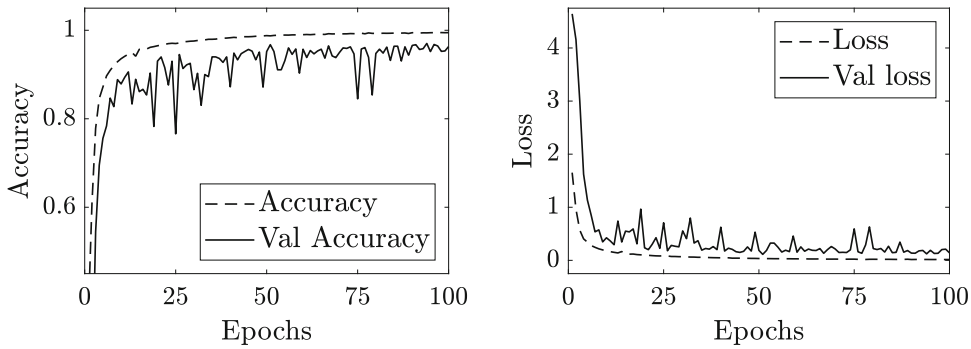


Fig. 4 Accuracy and loss over epochs using MobileNetV2 on training and validation set

Table 3 Model evaluation of MobileNetV2

Class	Precision	Recall	$F1$-score	Support
0	1.00	0.99	1.00	2496
1	0.99	1.00	0.99	2496
2	1.00	0.98	0.99	2496
3	0.98	0.98	0.98	2496
4	0.98	0.98	0.98	2496
5	0.97	0.99	0.98	2496
6	0.99	0.96	0.98	2496
7	0.96	0.99	0.97	2496
8	0.95	0.98	0.97	2496
9	0.99	0.94	0.97	2496
10	0.99	0.98	0.99	2496
11	0.99	0.98	0.98	2496
12	0.99	0.98	0.98	2496
13	0.96	0.99	0.97	2496
14	0.96	0.92	0.94	2496
15	0.98	0.99	0.98	2496
16	0.94	0.94	0.94	2496
17	0.92	0.98	0.95	2496
18	0.98	0.95	0.96	2496
19	0.97	0.97	0.97	2496
Accuracy			0.97	49,920
Macro avg	0.97	0.97	0.97	49,920
Weighted avg	0.97	0.97	0.97	49,920

Table 3 displays results of the MobileNetV2 including the precision, recall and $F1$-score for each class when evaluated using the testing set. The model can predict with high levels of accuracy on the testing dataset. However, we can identify two sources of error that are not considered here: (1) error due to noise in the image, which can lead to classify the patch as sharper than it is and (2) error because the patch being classified is not represented by those examples in the dataset, which could result in inaccurate predictions. We minimize the effect of these two types of errors by using a refinement algorithm.

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 $I_{\mathrm{p}}(q)$. 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 16 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. For simplicity, we define the blurriness of the pixel at location q as the average of classification results due to all patches containing the pixel $I(q)$:

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

where $|\Omega_q|$ is the number of patches which have the pixel $I(q)$. Since the classification result O_{p} is in the interval $[0, 19]$, the blur-map $M(q)$ is also in the same interval.

We now discuss the computational complexity. Assuming the CNN takes T seconds to process a patch, the total inference time for an image is given by:

$$t = T \times \frac{N}{s^2} \tag{7}$$

where N is the number of pixels in the image and s is the step-size. The time T depends on the programming language, level of code optimization, and the available computing resources. For instance, it takes 9.9ms to process one patch on a Macbook pro 2022 with 16gB ram and an M1 Pro CPU running a MATLAB toolbox called "Deep Learning Toolbox Converter for TensorFlow Models".

The optimal step-size depends on the image resolution and the complexity of the image. Generally, for images of large size, the step size should be 32 pixels to avoid long processing times. For images of small size, better results are obtained by using a step size of 4 pixels such that so small objects with different blur levels are preserved. As a rule thumb, we suggest using a step size that can preserve the main structure of the image.

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 [27] which produces the alpha matting matrix. However, this method is computationally very expensive. The Guided Image Filter (GIF) The Guided Image Filter (GIF) [29] has demonstrated a performance comparable to that of the matting Laplacian algorithm at a lower computational cost. The weighted version of the GIF called Weighted Guided Image Filter (WGIF) [64] improves the performance of the GIF and produces fewer halo effects. However both

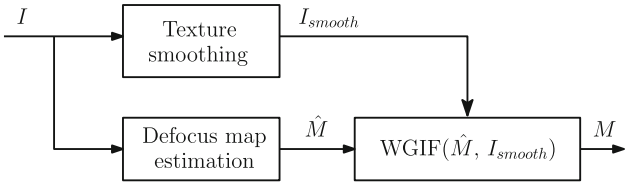


Fig. 5 Defocus map refinement

GIF and WGIF transfer details from the guidance image to the processed image, which is an undesirable feature for our applications because texture information from the original image will be transferred to the refined defocus map leading to wrong predictions in pixel blurriness. To overcome this limitation, we propose to use a modified version of the WGIF [64]-based algorithm to refine the defocus map. 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 5 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. For the sake of simplicity, to produce the guidance image I_{smooth} we use the self-guided WGIF [64] iteratively.

Figure 6 shows an example of the refinement algorithm. The blur map shown in Fig. 6b was estimated using the proposed CNN approach with a step size of 16 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 $N_{\text{iter}} = 7$. The guidance image is shown in Fig. 6c and the refinement result is shown in Fig. 6g. 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 those fish in the background.

Figure 6 also shows the result of refining the blur map with other methods such as an iterative WGIF [64] using the original image as guidance, mutually guided image faltering muGIF [65] and matting Laplacian [17] to visually appreciate the effectiveness of our method. All four methods clearly remove the blocking effect and wrong predictions in the blur map. However, the WGIF does not preserve edge consistency. The smoothing effect of muGIF is very strong, it removes all the background objects and shrinks the main object's silhouette. Apart from being computationally too expensive, the matting Laplacian algorithm properly preserves the objects in the background but it does transfer some undesirable details from the original image to the defocus map (brightness on the upper side and tail of the closest fish). Our refinement method does preserve defined edges and smooths objects at a lower computational cost than the muGIF and matting Laplacian algorithms. The running times to produce the results in Fig. 6 are presented in Table 4, which provides evidence of the relatively faster running time of the proposed algorithm.

Another important characteristic of a our refinement method is that it accurately preserves the defocus level classified by the CNN, i.e., after the refinement process is applied

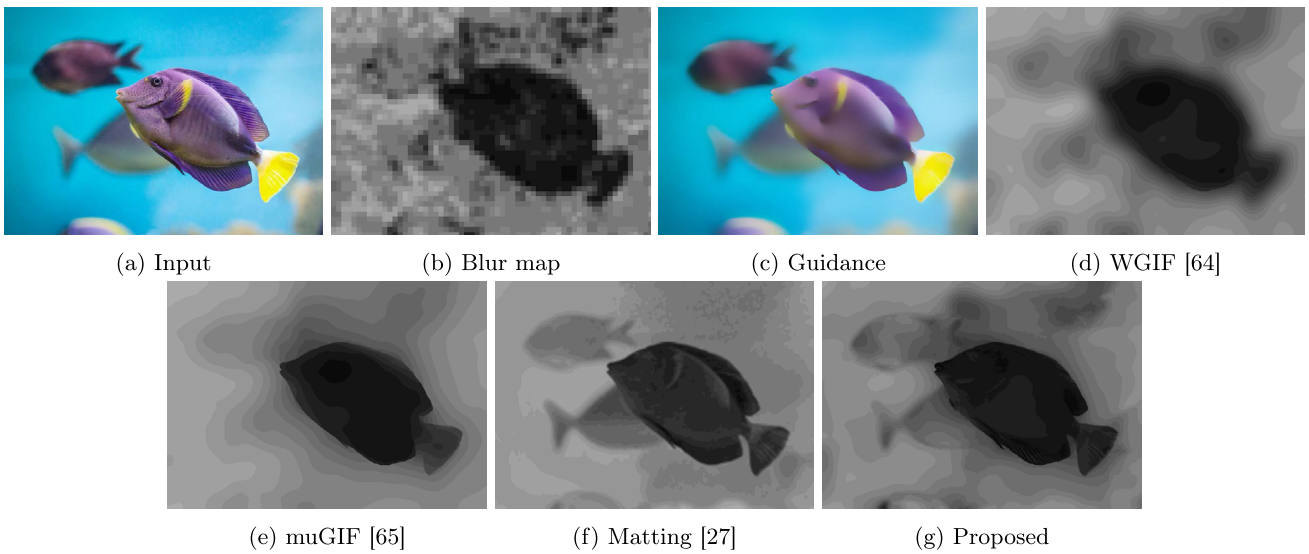


Fig. 6 Defocus blur map refinement. **a** Input image, **b** Blur map, **c** Guidance image, **d–g** Refined blur maps by WGIF, muGIF, Matting laplacian and the proposed method

Table 4 Running time comparison between different refinement methods (coded in MATLAB) for an input size of 1000×700 pixels running on a PC with Apple M1 Pro Chip with 16GB ram

Method	Time (S)
WGIF	0.022
muGIF	2.462
Matting	33.737
Proposed	0.491

the effect of wrong predictions are mitigated and the edges are improved while the defocus prediction values are preserved. This can be better appreciated in Fig. 7a, where we created a binary version of the original defocus map. The binary image is obtained by assigning a value 0 to every pixel q where $M(q) \leq \lambda$ and 1 otherwise. The parameter λ is a user defined threshold ($\lambda = 4$ in the figure). In the figure we can see all the pixels with a level of blurriness lower or equal than 4 are shown in black color. We can clearly see that most of those pixels belong to the biggest fish in the image and a small number of pixels belong to the wrong predictions. Figures 7b to e show the same threshold ($\lambda = 4$) applied to the defocus maps refined using the WFIG, muGIF, Matting Laplacian and the proposed method. Our method is the only one which can segment the main object in focus correctly, correct the edges, and remove incorrect predictions.

4 Comparison of defocus estimation algorithms

In this section, we compare qualitatively and quantitatively 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 five handcrafted features: "Edge-based defocus blur estimation with adaptive scale selection," from Karaali et al. [25], "Just noticeable defocus blur detection and estimation" from Shi et al. [34], "Lbp-based segmentation of defocus blur" from Yi et al. [31], "Defocus map estimation from a single image" from Zhuo et al. [23] and "Fast defocus map estimation" (FDM) from Chen et al. [26] and a CNN-based method from Lee et al. "Deep Defocus Map Estimation Using Domain Adaptation" (DMNet) [51]. The parameters settings in this section have been chosen to achieve the best result possible in terms of defocus estimation and edge preservation.

4.1 Comparison with classical methods

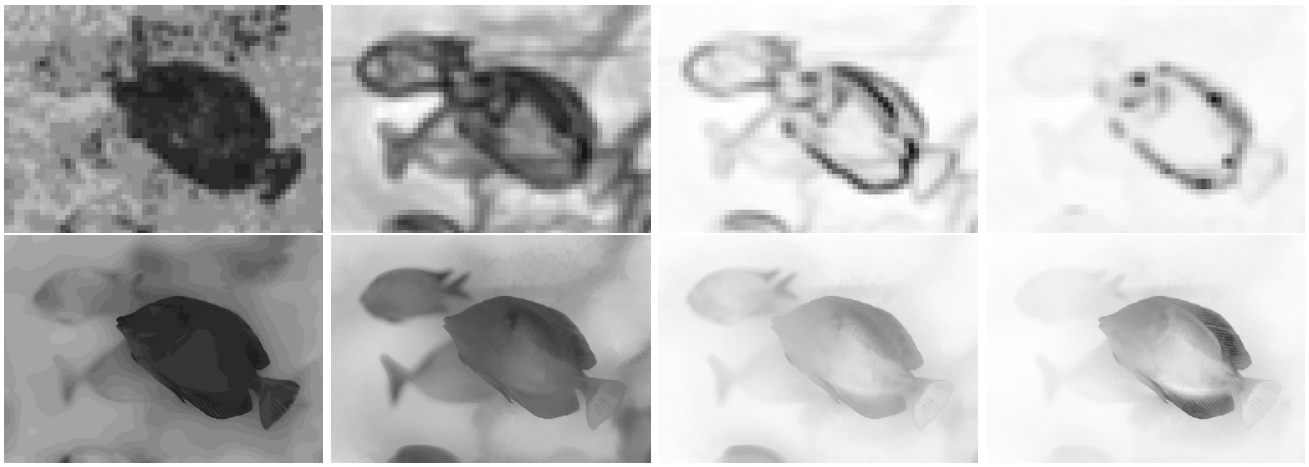
We studied three methods: entropy, standard deviation, and variance of Laplacian [58]. The ideas of these three methods are similar. The blurriness of a pixel is measured by a quantity such as the entropy or standard deviation of a patch of pixels centred at that pixel. For the variance of Laplacian, 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. 8. 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 gray scale indicating it is wrongly classified as out of focus by this method.

4.2 Comparison with modern methods

The bird image shown in Fig. 9a presents different levels of defocus blur. Karaali's method is able to detect the area in focus but fails to propagate the levels and properly refine the map. The defocus map produced by Shi's method with $\sigma = 2, \sigma_r = 0.2, \sigma_s = 10$ and $N_{\text{iter}} = 3$ does not

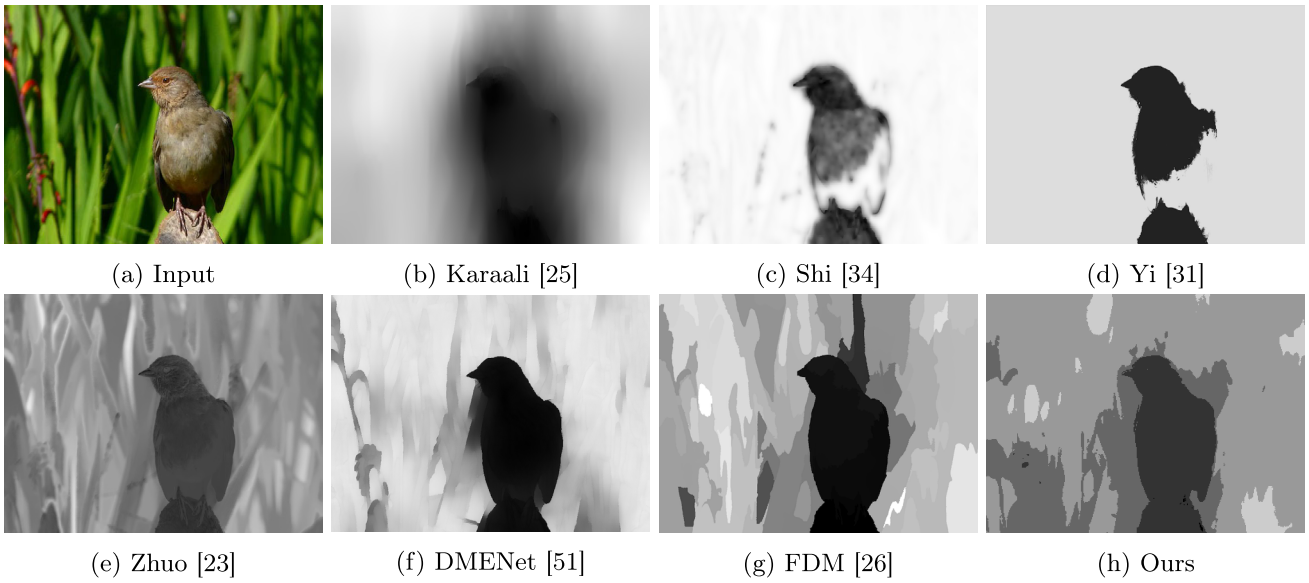
(a) CNN prediction (b) WGIF [64] (c) muGIF [65] (d) Matting [27] (e) Proposed

Fig. 7 Binary defocus maps obtained by setting a threshold $\lambda = 4$ and assigning a 0 to those pixels q where $M(q) \leq \lambda$ and 1 otherwise. **a** Binary defocus map from original prediction, **b–e** Binary defocus map from the map refined by WGIF, muGIF, Matting Laplacian, and our method, respectively. This image shows the effectiveness of our method in preserving the defocus level predicted by the CNN



(a) Proposed method (b) Entropy (c) Standard deviation (d) Variance of the Lap. [58]

Fig. 8 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 Sect. 3.3 is shown in the bottom row. **a** Our method. **b** Entropy. **c** Standard deviation. **d** Variance of the Laplacian

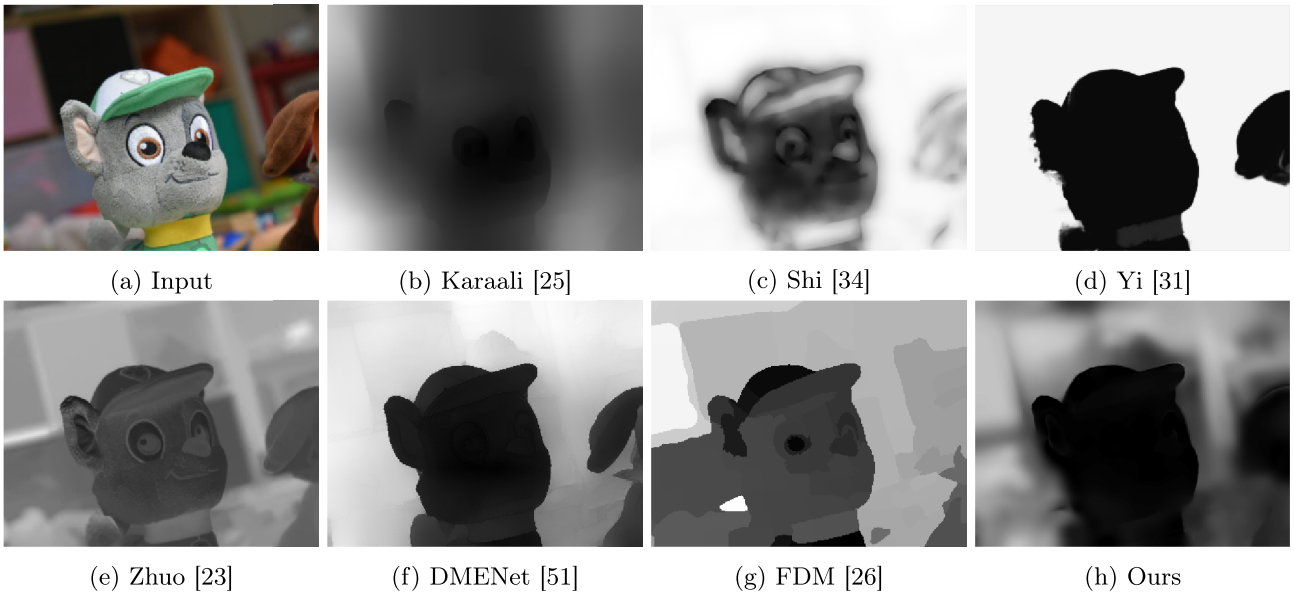


(a) Input (b) Karaali [25] (c) Shi [34] (d) Yi [31]

(e) Zhuo [23] (f) DMENet [51] (g) FDM [26] (h) Ours

Fig. 9 Blur map comparisons, **a** Input image, **b** Karaali [25] ($\sigma_1 = 1, \sigma_2 = [1 : 0.5 : 5]$), **c** Shi [34] ($\sigma = 2, \sigma_r = 0.2, \sigma_s = 10, N_{iter} = 3$), d) Yi [31] ($\beta = 0.25, s_1 = 7, s_2 = 11, s_3 = 15, T_{\text{LBP}} = 0.025$), **e** Zhuo [23] ($e = 0.01, \sigma = 1, \lambda = 0.001, M = 10$), **f** DMENet [51], **g** FDM [26] ($\sigma = 0.75$), **h** Proposed method

capture all the in-focus pixels in the image. Yi's algorithm produces a similar result than image (c) when implemented using a threshold $T_{\text{LBP}} = 0.025$, square local regions of size $s_1 = 7, s_2 = 11, s_3 = 15$ for each level and setting the weight $\beta = 0.25$ for the multi-scale inference step. Zhuo's algorithm is shown in image (e). We set the edge threshold as $e = 0.01$ and the standard deviation as $\sigma = 1$ for the blur detection step. The interpolation step was performed using $\lambda = 0.001$ and a maximum blur level $M = 10$. Zhuo's method successfully identify that the bird is in focus. However a few details from the background are transferred to the defocus map after the interpolation step. Chen's method (FDM) uses Zhuo's approach to estimate the blur level on edges, but uses the SLIC algorithm [30] for the interpolation. In our simulation, the SLIC parameters are set as: 200 super-pixels and a standard deviation $\sigma = 0.75$. As shown in the figure, for this particular image the focused region is not well defined. In comparison, the DMNet has a very good performance in capturing both background and foreground blurriness levels. The result produced by the proposed algorithm is shown in figure (h). Our method can represent the different levels of blurriness in the bird image according to the CNN predic-



(a) Input (b) Karaali [25] (c) Shi [34] (d) Yi [31]

(e) Zhuo [23] (f) DMENet [51] (g) FDM [26] (h) Ours

Fig. 10 Blur map comparisons, **a** Input image, **b** Karaali [25] ($\sigma_1 = 1, \sigma_2 = [1 : 0.5 : 5]$), **c** Shi [34] ($\sigma = 2, \sigma_r = 0.2, \sigma_s = 10, N_{iter} = 3$), **d** Yi [31] ($\beta = 0.25, s_1 = 7, s_2 = 11, s_3 = 15, T_{\text{LBP}} = 0.025$), **e** Zhuo [23] ($e = 0.01, \sigma = 1, \lambda = 0.001, M = 10$), **f** DMENet [51], **g** FDM [26] ($\sigma = 0.75$), **h** Proposed method

tion. 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 the last row of Fig. 10. We can see that Karaali's and Shi's algorithms have similar performance as those in the previous example while Yi's algorithm produces a better result on this image. It fully captures and segments focused pixels in the foreground. The methods of Zhuo and Chen 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.

4.3 Accuracy evaluation

Quantitatively evaluating the accuracy of the defocus map produced for our method and comparing with others is a challenging task. There is no current dataset that allows to evaluate different levels of defocus. In this section, we perform an experiment using the Blur Detection Dataset CUHK [66] to show the effectiveness of our approach and the comparison with others [23,25,34,50,51].

The CUHK Blur Detection Dataset contains 1000 human labeled images for binary blur detection at a pixel level. The images are divided into two categories: blurred due to motion and blurred due to defocus. In this experiment, we only pick the first 200 images with defocus blur since DMENet is partially trained with the remaining images in the dataset. The ground truth images are binary maps, where a "0" implies an in-focus pixel and a "1" suggests an out-of-focus pixel. To evaluate our algorithm and to avoid re-training our model, we post process each defocus map with a threshold operation to get a compatible binary map. The binary image is obtained by assigning a value "1" to every pixel q where $M(q) \geq \lambda$ and "0" otherwise. The parameter λ is set to each image individually to achieve the highest accuracy possible. The final accuracy value is then calculated by averaging the percentages of properly classified pixels on each image. We compare our results with those reported in [51] in Table 5. Our new patch-based approach can achieve 87.71% accuracy slightly over-passing DMENet [51] whom is able to achieve 87.35% accuracy with a significantly more complex model.

Table 5 Accuracy on CUHK dataset [66]

Method	Accuracy (%)
Zhuo et al. [23]	72.96
Karaali et al. [25]	76.54
Shi et al. [34]	77.81
Park et al. [50]	84.08
DMENet [51]	87.35
Proposed	87.71

For a more comprehensive comparison, we present the Precision-Recall curve in Fig. 11. To obtain these values, we convert the defocus maps to binary maps using different thresholds levels (λ) ranging from M_{min} to M_{max}, where

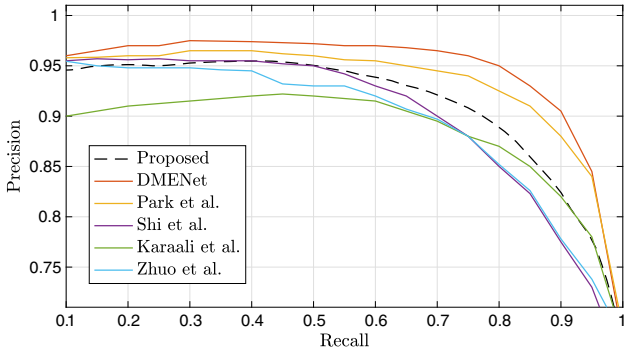


Fig. 11 Precision vs Recall comparison on a subset of CUHK dataset [66]

$M_{\min}$ and $M_{\max}$ are the minimum and maximum defocus levels detected in each image. The precision vs recall metric shows that the proposed method performs better than Karaali, Zhuo and Shi in this particular dataset, but it is outperformed by DMENet and Park which are able to achieve a lower false positive rate and a lower false negative rate than our method. We remark that our method is not designed for making a binary defocus map. The comparison presented in this section is intended to show the capability of the proposed method in applications in which a binary decision of whether a pixel is in focus has to be made.

5 Applications

In this section, we demonstrate applications of the proposed defocus map estimation in adaptive enhancement and defocus magnification.

5.1 Adaptive enhancement

Unsharp masking (UM) [67] is frequently used for image enhancement through image sharpening. The Unsharp Masking (UM) algorithm can be described as

$$J(q) = I(q) + \lambda Z(q) \tag{8}$$

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 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 nonlinear or edge-aware filters [67–69] and a pixel adaptive λ that produces a spatially varying enhancement [10,70].

We follow the same idea of using a pixel adaptive λ and propose an adaptive unsharp masking algorithm for images with multiple defocus levels 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:

$$\lambda(q) = \lambda_{\max} \times \lambda_1(q) \times \lambda_2(q) \tag{9}$$

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

$$\lambda_1(q) = \frac{1}{1 + e^{-\alpha_1(M(q)-\beta_1)}} \tag{10}$$

$$\lambda_2(q) = 1 - \frac{1}{1 + e^{-\alpha_2(M(q)-\beta_2)}} \tag{11}$$

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. 12. In this example, three out of the four parameters (α_1, β_1, α_2 and β_2) are kept fixed while one varies. The scale of the blur map is linearly normalized from the interval [0, 19] to [0, 1] for better visualization. Figure 12a, b c and d show the effect of varying β_1, β_2, α_1 and α_2, respectively, while keeping the other parameters fixed. The two parameters β_1 and β_2 indicate the points where the gain is $0.5 \times \lambda_{\max}$, while the other two parameters α_1 and α_2 set the growth speed of the curve.

In Fig. 13, we present an example of obtaining the gain map through a non-linear transformation of the defocus map. Figure 13a shows the input image with different levels of blurriness. For example, the bird in the foreground is relatively sharp in comparison with the branches and the background which is heavily defocused. Enhancing this image is a real challenge. Figure 13b and Fig. 13c show the defocus blur map and the gain map respectively. The gain map was calculated using the nonlinear transformation setting $\alpha_1 = 46, \beta_1 = 0.1, \alpha_2 = 183, \beta_2 = 0.27$ and $\lambda_{max} = 2$. We aim to adaptively enhance the image by preventing over-sharpening in focused regions and avoiding sharpening in blur sections.

Figure 13 also shows the comparison of our method with the classical unsharp masking (UM) and two newer methods contrast adaptive sharpening (CAS)[1] and Generalized

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

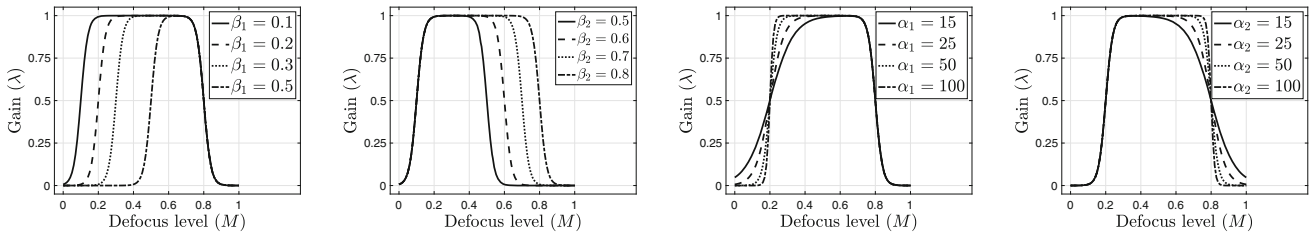


Fig. 12 Nonlinear transformation of the defocus level. For easy visualization, the defocus level is normalized and the parameter settings are: **a** $\alpha_1 = 50$, $\beta_2 = 0.8$ and $\alpha_2 = 50$, **b** $\alpha_1 = 50$, $\beta_1 = 0.1$ and $\alpha_2 = 50$, **c** $\beta_1 = 0.2$, $\beta_2 = 0.8$ and $\alpha_2 = 50$, **d** $\alpha_1 = 50$, $\beta_1 = 0.2$ and $\beta_2 = 0.8$

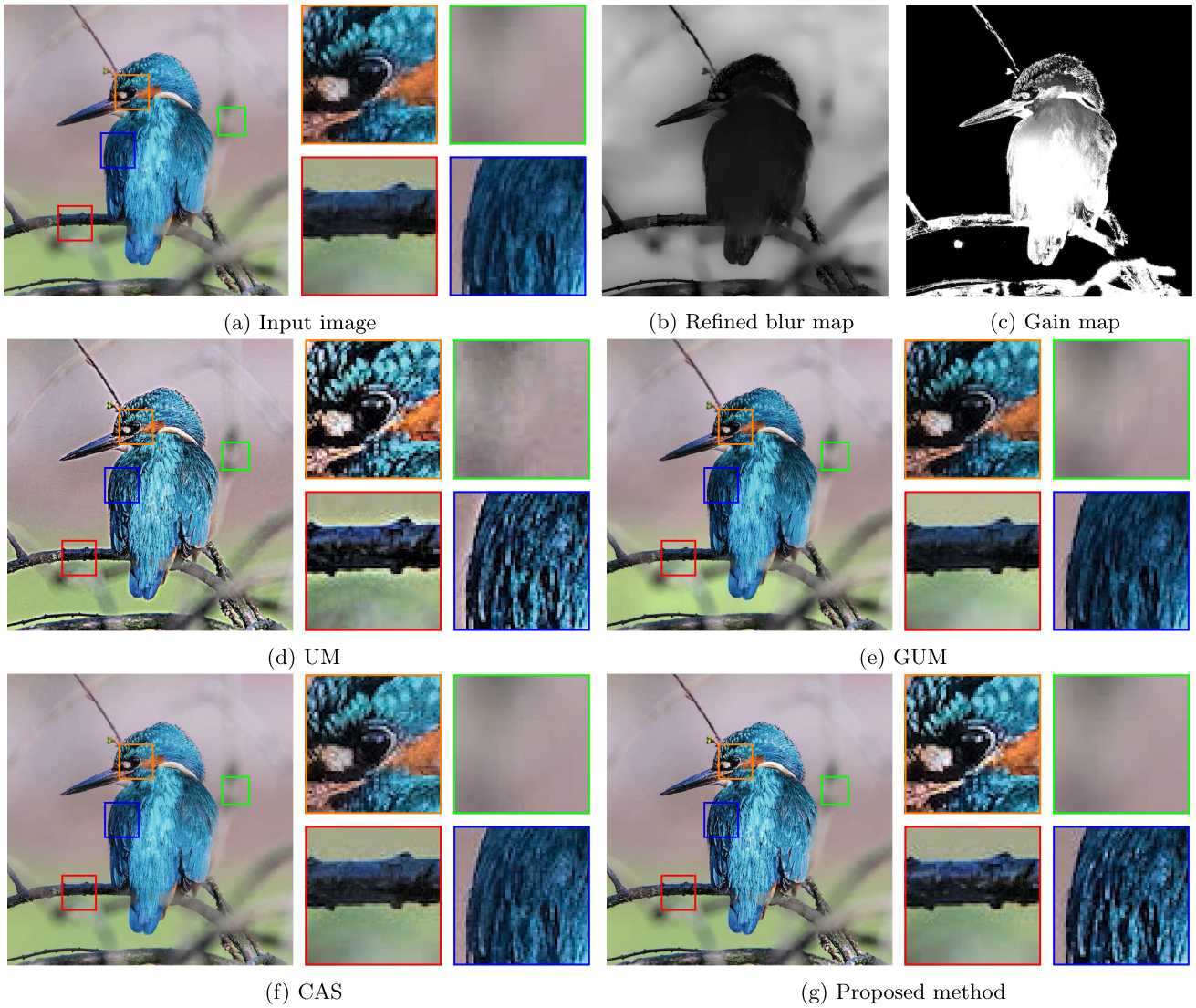


Fig. 13 Adaptive image enhancement comparison, **a** Original image, **b** Refined blur map (step size of 16 pixels), **c** Gain map ($\alpha_1 = 46$, $\beta_1 = 0.1$, $\alpha_2 = 183$, $\beta_2 = 0.27$ and $\lambda_{\max} = 2$), **d** UM algorithm $\lambda = 2$, **e** GUM ($\kappa = 5 \times 10^{-4}$, $\lambda = 7$), **f** CAS ($\lambda = -0.125$), **g** Proposed method ($\alpha_1 = 46$, $\beta_1 = 0.1$, $\alpha_2 = 183$, $\beta_2 = 0.27$ and $\lambda_{\max} = 2$)

Unsharp Masking (GUM) [71]. The parameter values for all the methods have been selected to produce a visually perceptible enhancement with minimum distortion and artifacts. We can see that applying UM with a fixed global gain $\lambda = 2$ produces an unpleasant sharpening in the background (green box) and over-sharpening of areas in focus such as

Table 6 Image quality assessment of results in Fig. 13. Best results are indicated in bold fonts

Metric	Original	UM	GUM	CAS	Proposed
BRISQUE ↓	20.83	44.41	24.89	55.19	**20.61**
IL-NIQE ↓	26.72	25.331	26.0949	26.0814	**24.4754**
NBIQA ↑	66.244	66.4285	63.5012	**76.4527**	72.0607
PSS ↓	0.091	0.1161	0.0921	0.1407	**0.0887**
BLIINDS-II ↓	42.5	40	**37**	52.5	43

the indicated by the orange, red and blue boxes. The CAS algorithm was implemented using a gain $\lambda = -0.125$. We can see in Fig. 13f that the algorithm performs very well in regions with medium and high defocus blur. However, it cannot deal with the blur areas properly producing a noisy result(green and red boxes). The GUM algorithm (Fig. 13e) 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 increases. The result of applying the proposed method is shown in Fig. 13g. It does not sharpen the heavily blurred background (green box) and prevents over-sharpening of highly focused pixels by limiting the sharpening gain (orange, red and blue boxes).

We perform the image quality assessment (IQA) of the images shown in Fig. 13 and present the results in Table 6. We use a series of no-reference (NR) IQA methods such as BRISQUE [72], IL-NIQE [73], NBIQA [74], PSS [75] and BLIINDS-II [76] to objectively evaluate the performance of the proposed algorithm. These metrics take into account features from the spatial domain or transform domain to assess the quality and naturalness of an image. We should point out that these quality measurements use different scales. For example, while for the BRISQUE method a smaller score means a better image quality, for the NBIQA method a larger score means a better quality. In Table 6 we can see that our method achieves the best result on BRISQUE, IL-NIQUE and PSS, and second-best on NBIQA, this indicates that the proposed method produces a more natural-looking image enhancement than the other three methods. However, BLIINDS-II gives the second best score to UM algorithm leaving our method in 3rd position. In fact, from Fig. 13 we can see that UM algorithm produces excessive halo artifacts due to enhancement in high contrast regions such as shown in the red box. The BLIINDS-II score for the UM result is thus debatable. Therefore, such quality metrics do not always generalize well as a full-reference metric. It is always advisable to perform a visual comparison when possible. Indeed, in image enhancement applications in which the user of the image is human, he/she will manipulate parameters of the algorithm to adjust the image until a satisfactory outcome is produced. This a very subjective process which cannot be replaced by a quality metric.

5.2 Defocus magnification

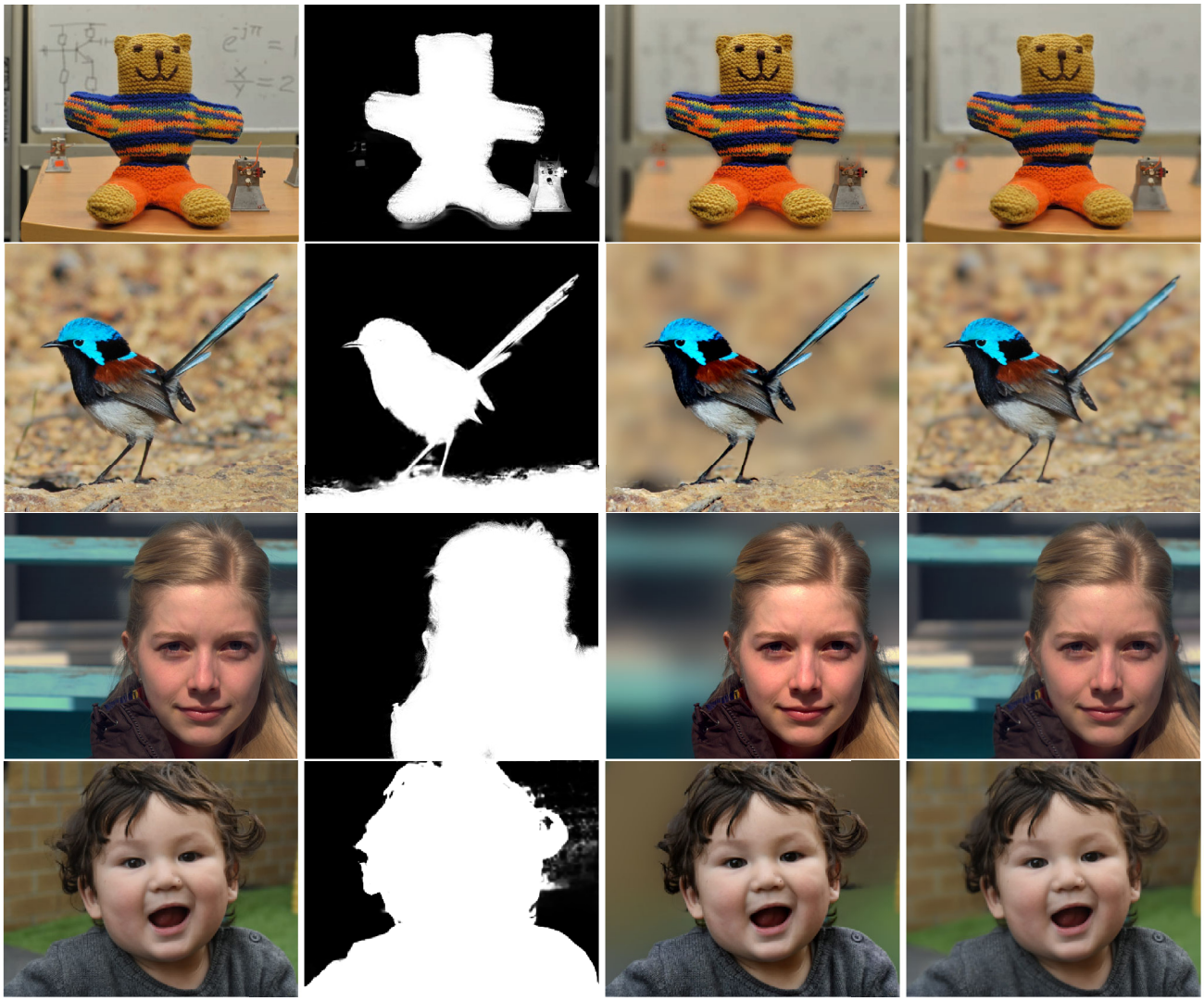
In photography, the shallow depth-of-field (DoF) technique is used to make the main object stand out from the background by producing an image in which in-focus objects are with great details and contrast, while out-of-focus objects are greatly smoothed. In a camera, the DoF is controlled by a combination 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. Enhancement of an image by producing/increasing the SDoF effect is thus a practical image processing problem.

Computer vision algorithms were recently developed to achieve the shallow depth of field effect. For example, in DeepLens [77], the depth information from a RGB-D sensor was used to train a CNN to create the SDoF effect from an image that has everything in focus. Sakurikar et al. [78] proposed an end-to-end GAN architecture that selectively increases the level of blurriness in the input image while keeping the edges of the in-focus object intact. Also, in [79] an object segmentation based algorithm differentiates main objects from the those in the background. The latter are blurred to create the SDoF effect.

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 $S(p)$,

$$R(q) = W(q)S(q) + (1 - W(q))B(q) \tag{12}$$

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 [29] or one of its weighted versions [64,80]. 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 to the overall greater degree of smoothness. To better control the smoothing result, we iterate the guided filter in a self-guidance mode. In our experiments presented in Fig. 14, we use $r = 33$, $\epsilon = 128$, and 5 iterations. The sharpened image S is produced by the



(a) input image (b) Weight map (c) Proposed (d) DeepLens [77]

Fig. 14 SDoF effect example. **a** Original image, **b** Weight map, **c** SDoF result, **d** DeepLens result

classical unsharp masking algorithm with a fixed $\lambda = 0.25$ for illustration purpose.

The weight map W is determined by the nonlinear mapping:

$$W(q) = 1 - \exp\left(-\left|\frac{M(q) - 10}{\sigma}\right|^{\gamma}\right) \tag{13}$$

where σ and γ are two parameters.

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 (12), this requires that $W(q) = W_0 \to 1$. 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 $W(q) = W_1 \to 0$. 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 (13). 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. 14 which show that the proposed algorithm can effectively produce the SDoF effect. In the original images shown in Fig. 14a the foreground is in focus while the background is slightly out of focus. After applying the nonlinear 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. 14b. The resulting SDoF image is shown in Fig. 14c. 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. A more realistic result can be produced by using different smoothing techniques to produce B such as the method proposed in [81].

A notable limitation of this algorithm is that the input image must present a clear distinction between defocus levels to be able to perform the blur-based segmentation. However, the method can still be applied partially if the main object is segmented manually or with an additional machine learning model.

6 Conclusion

In this paper, we address a challenging 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 into 20 different levels of blurriness. Using a sliding window algorithm and a novel weighted guided filter-based refinement, a pixel level defocus map is obtained.

We have tested and validated the proposed method by comparing it qualitatively and quantitatively with classical and state-of-the-art methods. We have also demonstrated successful applications of the blur map estimation in adaptive image enhancement, and producing shallow of depth of field effects.

We remark that the underlying assumption of this work imposes a limitation to its application to selectively smooth objects which are in focus. This is a problem of first performing object selection/recognition, and then performing smoothing. The proposed blurriness estimation is not designed for the selection of objects which are in focus. However, once the object is selected, the proposed refinement and enhancement method can still be applied.

Data Availability The datasets generated during the current study are available in the Github repository: https://github.com/fergaletto/DMC_TVC

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Pentland, A.P.: A new sense for depth of field. IEEE Trans. Pattern Anal. Mach. Intell. **4**, 523–531 (1987)
2. Mather, G.: Image blur as a pictorial depth cue. Proc. R. Soc. B **263**(1367), 169–172 (1996)
3. Xue, F., Blu, T.: A novel sure-based criterion for parametric PSF estimation. IEEE Trans. Image Process. **24**(2), 595–607 (2014)
4. Chan, S.H., Nguyen, T.Q.: Single image spatially variant out-of-focus blur removal, In: 2011 18th IEEE International Conference on Image Processing, pp. 677–680 (2011)
5. Cheong, H., Chae, E., Lee, E., Jo, G., Paik, J.: Fast image restoration for spatially varying defocus blur of imaging sensor. Sensors **15**(1), 880–898 (2015)
6. Bae, S., Durand, F.: Defocus magnification, In: Computer Graphics Forum, vol. 26, no. 3. Wiley Online Library, pp. 571–579 (2007)
7. van Zyl Marais, I., Steyn, W.H.: Robust defocus blur identification in the context of blind image quality assessment. Sign. Process. Image Commun. **22**(10), 833–844 (2007)
8. Al-Obaidi, F.E.: Image quality assessment for defocused blur images. Am. J. Sign. Process. **5**(3), 51–55 (2015)
9. Nasonov, A., Krylov, A., Lyukov, D.: Image sharpening with blur map estimation using convolutional neural network. Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.: ISPRS Arch. **42**(2/W12), 161–166 (2019)
10. Ye, W., Ma, K.-K.: Blurriness-guided unsharp masking. IEEE Trans. Image Process. **27**(9), 4465–4477 (2018)
11. Ziou, D., Deschenes, F.: Depth from defocus estimation in spatial domain. Comput. Vis. Image Underst. **81**(2), 143–165 (2001)
12. Lin, J., Ji, X., Xu, W., Dai, Q.: Absolute depth estimation from a single defocused image. IEEE Trans. Image Process. **22**(11), 4545–4550 (2013)
13. Shi, J., Tao, X., Xu, L., Jia, J.: Break ames room illusion: depth from general single images. ACM Trans. Graph. **34**(6), 1–11 (2015)

14. Tang, H., Cohen, S., Price, B., Schiller, S., Kutulakos, K.N.: Depth from defocus in the wild, In: Proceedings of IEEE CVPR, pp. 2740–2748 (2017)
15. Bailey, S.W., Echevarria, J.I., Bodenheimer, B., Gutierrez, D.: Fast depth from defocus from focal stacks. Vis. Comput. **31**(12), 1697–1708 (2015)
16. Suwajanakorn, S., Hernandez, C., Seitz, S.M.: Depth from focus with your mobile phone, In: Proceedings of IEEE CVPR, pp. 3497–3506 (2015)
17. Levin, A., Fergus, R., Durand, F., Freeman, W.T.: Image and depth from a conventional camera with a coded aperture. ACM Trans. Graph. **26**(3), 70-es (2007)
18. Zhou, C., Lin, S., Nayar, S.K.: Coded aperture pairs for depth from defocus and defocus deblurring. Int. J. Comput. Vis. **93**(1), 53–72 (2011)
19. Vu, D.T., Chidester, B., Yang, H., Do, M.N., Lu, J.: Efficient hybrid tree-based stereo matching with applications to postcapture image refocusing. IEEE Trans. Image Process. **23**(8), 3428–3442 (2014)
20. Xin, S., Wadhwa, N., Xue, T., Barron, J.T., Srinivasan, P.P., Chen, J., Gkioulekas, I., Garg, R.: Defocus map estimation and deblurring from a single dual-pixel image, In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 2228–2238 (2021)
21. Tai, Y.-W., Brown, M.S.: Single image defocus map estimation using local contrast prior, In: Proceedings of the IEEE/CVF International Conference on Computer Vision. IEEE, pp. 1797–1800 (2009)
22. Zhang, X., Wang, R., Jiang, X., Wang, W., Gao, W.: Spatially variant defocus blur map estimation and deblurring from a single image. J. Vis. Commun. Image Represent. **35**, 257–264 (2016)
23. Zhuo, S., Sim, T.: Defocus map estimation from a single image. Patt. Recognit. **44**(9), 1852–1858 (2011)
24. Hu, H., De Haan, G.: Low cost robust blur estimator, In: 2006 International Conference on Image Processing, pp. 617–620 (2006)
25. Karaali, A., Jung, C.R.: Edge-based defocus blur estimation with adaptive scale selection. IEEE Trans. Image Process. **27**(3), 1126–1137 (2017)
26. Chen, D.-J., Chen, H.-T., Chang, L.-W.: Fast defocus map estimation, In: 2016 IEEE International Conference on Image Processing (ICIP), IEEE, pp. 3962–3966 (2016)
27. Levin, A., Lischinski, D., Weiss, Y.: A closed-form solution to natural image matting. IEEE Trans. Patt. Anal. Mach. Intell. **30**(2), 228–242 (2007)
28. Chen, Q., Li, D., Tang, C.K.: KNN matting. IEEE Trans. Patt. Anal. Mach. Intell. **35**(9), 2175–2188 (2013)
29. He, K., Sun, J., Tang, X.: Guided image filtering. IEEE Trans. Patt. Anal. Mach. Intell. **35**(6), 1397–1409 (2012)
30. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Süsstrunk, S.: Slic superpixels compared to state-of-the-art superpixel methods. IEEE Trans. Patt. Anal. Mach. Intell. **34**(11), 2274–2282 (2012)
31. Yi, X., Eramian, M.: Lbp-based segmentation of defocus blur. IEEE Trans. Image Process. **25**(4), 1626–1638 (2016)
32. Chakrabarti, A., Zickler, T., Freeman, W.T.: Analyzing spatially-varying blur, In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 2512–2519 (2010)
33. Zhu, X., Cohen, S., Schiller, S., Milanfar, P.: Estimating spatially varying defocus blur from a single image. IEEE Trans. Image Process. **22**(12), 4879–4891 (2013)
34. Shi, J., Xu, L., Jia, J.: Just noticeable defocus blur detection and estimation, In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 657–665 (2015)
35. Javaran, T.A., Hassanpour, H., Abolghasemi, V.: Automatic estimation and segmentation of partial blur in natural images. Vis. Comput. **33**(2), 151–161 (2017)
36. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks, Proc. Adv. Neural Inf. Process. Syst., 1097–1105 (2012)
37. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: inverted residuals and linear bottlenecks, In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4510–4520 (2018)
38. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-CNN: towards real-time object detection with region proposal networks. IEEE Trans. Patt. Anal. Mach. Intell. **39**(6), 1137–1149 (2016)
39. Lateef, F., Ruichek, Y.: Survey on semantic segmentation using deep learning techniques. Neurocomputing **338**, 321–348 (2019)
40. Jin, Z., Iqbal, M.Z., Bobkov, D., Zou, W., Li, X., Steinbach, E.: A flexible deep CNN framework for image restoration. IEEE Trans. Multimed. **22**(4), 1055–1068 (2019)
41. Anwar, S., Li, C.: Diving deeper into underwater image enhancement: a survey. Sign. Process.: Image Commun. **89**, 115978 (2020)
42. Yang, W., Zhang, X., Tian, Y., Wang, W., Xue, J.-H., Liao, Q.: Deep learning for single image super-resolution: a brief review. IEEE Trans. Multimed. **21**(12), 3106–3121 (2019)
43. Nasonova, A., Krylov, A.: Deblurred images post-processing by Poisson warping. IEEE Sign. Process. Lett. **22**(4), 417–420 (2014)
44. Ma, K., Fu, H., Liu, T., Wang, Z., Tao, D.: Deep blur mapping: exploiting high-level semantics by deep neural networks. IEEE Trans. Image Process. **27**(10), 5155–5166 (2018)
45. Zhao, W., Zhao, F., Wang, D., Lu, H.: Defocus blur detection via multi-stream bottom-top-bottom network. IEEE Trans. Patt. Anal. Mach. Intell. **42**(8), 1884–1897 (2019)
46. Jiang, Z., Xu, X., Zhang, C., Zhu, C.: Multianet: a multi-attention network for defocus blur detection, In: Proceedings of International Workshop on Multimedia Signal Processing (MMSP). IEEE, pp. 1–6 (2020)
47. Tang, C., Liu, X., An, S., Wang, P.: Br^{2}net: Defocus blur detection via a bidirectional channel attention residual refining network. IEEE Trans. Multimed. **23**, 624–635 (2021)
48. Tang, C., Liu, X., Zhu, X., Zhu, E., Sun, K., Wang, P., Wang, L., Zomaya, A.: R^{2}mrf: defocus blur detection via recurrently refining multi-scale residual features. Proc. AAAI Conf. Artif. Intell. **34**(07), 12063–12070 (2020)
49. Zhai, Y., Wang, J., Deng, J., Yue, G., Zhang, W., Tang, C.: Global context guided hierarchically residual feature refinement network for defocus blur detection. Sign. Process. **183**, 107996 (2021)
50. Park, J., Tai, Y.W., Cho, D., Kweon, I.S.: A unified approach of multi-scale deep and hand-crafted features for defocus estimation, In: Proceedings of IEEE CVPR, pp. 2760–2769 (2017)
51. Lee, J., Lee, S., Cho, S., Lee, S.: Deep defocus map estimation using domain adaptation, In: Proceedings of IEEE Conference CVPR, pp. 12222–12230 (2019)
52. Bohra, M., Maheshwari, S., Gandhi, V.: Texturetomtf: predicting spatial frequency response in the wild. Sign. Image Video Process. **14**(6), 1163–1170 (2020)
53. Ying, Z., Niu, H., Gupta, P., Mahajan, D., Ghadiyaram, D., Bovik, A.: From patches to pictures (paq-2-piq): Mapping the perceptual space of picture quality, In: Proceedings of IEEE CVPR, pp. 3575–3585 (2020)
54. Sakamoto, T.: Model for spherical aberration in a single radial gradient-rod lens. Appl. Opt. **23**(11), 1707–1710 (1984)
55. Favaro, P., Soatto, S.: A geometric approach to shape from defocus. IEEE Trans. Pattern Anal. Mach. Intell. **27**(3), 406–417 (2005)
56. Kubota, A., Aizawa, K.: Reconstructing arbitrarily focused images from two differently focused images using linear filters. IEEE Trans. Image Process. **14**(11), 1848–1859 (2005)
57. Agustsson, E., Timofte, R.: Ntire 2017 challenge on single image super-resolution: Dataset and study, In: Proceedings of IEEE CVPR, pp. 126–135 (2017)

58. Pech-Pacheco, J.L., Cristobal, G., Chamorro-Martinez, J., Fernandez-Valdivia, J.: Diatom autofocusing in brightfield microscopy: a comparative study, In: Proceedings 15th International Conference on Pattern Recognition, vol. 3, pp. 314–317 (2000)
59. Tan, M., Le, Q.V.: Efficientnet: Rethinking model scaling for convolutional neural networks, arXiv preprint arXiv:1905.11946, (2019)
60. Zoph, B., Vasudevan, V., Shlens, J., Le, Q.V.: Learning transferable architectures for scalable image recognition, In: Proceedings of IEEE CVPR, pp. 8697–8710 (2018)
61. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks, In: Proceedings of IEEE CVPR, pp. 4700–4708 (2017)
62. Ma, N., Zhang, X., Zheng, H.-T., Sun, J.: Shufflenet v2: practical guidelines for efficient cnn architecture design, In: Proc eedings of ECCV, pp. 116–131 (2018)
63. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980, (2014)
64. Li, Z., Zheng, J., Zhu, Z., Yao, W., Wu, S.: Weighted guided image filtering. IEEE Trans. Image Process. **24**(1), 120–129 (2014)
65. Guo, X., Li, Y., Ma, J., Ling, H.: Mutually guided image filtering. IEEE Trans. Patt. Anal. Mach. Intell. **42**(3), 694–707 (2018)
66. Shi, J., Xu, L., Jia, J.: Discriminative blur detection features, In: Proceedings of IEEE Conference on CVPR, pp. 2965–2972 (2014)
67. Ramponi, G.: A cubic unsharp masking technique for contrast enhancement. Sign. Process. **67**(2), 211–222 (1998)
68. Mitra, S.K., Li, H., Lin, I.-S., Yu, T.-H.: A new class of nonlinear filters for image enhancement, In: Proceedings of IEEE ICASSP, pp. 2525–2526 (1991)
69. Ramponi, G., Strobel, N.K., Mitra, S.K., Yu, T.-H.: Nonlinear unsharp masking methods for image contrast enhancement. J. Electron. Imaging **5**(3), 353–367 (1996)
70. Polesel, A., Ramponi, G., Mathews, V.J.: Image enhancement via adaptive unsharp masking. IEEE Trans. Image Process. **9**(3), 505–510 (2000)
71. Deng, G.: A generalized unsharp masking algorithm. IEEE Trans. Image Process. **20**(5), 1249–1261 (2011)
72. Mittal, A., Moorthy, A.K., Bovik, A.C.: No-reference image quality assessment in the spatial domain. IEEE Trans. Image Process. **21**(12), 4695–4708 (2012)
73. Zhang, L., Zhang, L., Bovik, A.C.: A feature-enriched completely blind image quality evaluator. IEEE Trans. Image Process. **24**(8), 2579–2591 (2015)
74. Ou, F.-Z., Wang, Y.-G., Zhu, G.: A novel blind image quality assessment method based on refined natural scene statistics, In: In 2019 IEEE International Conference on Image Processing (ICIP), pp. 1004–1008 (2019)
75. Min, X., Zhai, G., Gu, K., Fang, Y., Yang, X., Wu, X., Zhou, J., Liu, X.: Blind quality assessment of compressed images via pseudo structural similarity, In: In 2016 IEEE International Conference on Multimedia and Expo (ICME), IEEE, pp. 1–6 (2016)
76. Saad, M.A., Bovik, A.C., Charrier, C.: Blind image quality assessment: a natural scene statistics approach in the DCT domain. IEEE Trans. Image Process. **21**(8), 3339–3352 (2012)
77. Wang, L., Shen, X., Zhang, J., Wang, O., Lin, Z., Hsieh, C.-Y., Kong, S., Lu, H.: Deeplens: shallow depth of field from a single image, arXiv preprint arXiv:1810.08100, (2018)
78. Sakurikar, P., Mehta, I., Narayanan, P.: Defocus magnification using conditional adversarial networks, In: 2019 IEEE Winter Conference on Applications of Computer Vision (WACV). IEEE, pp. 1337–1346 (2019)
79. Wadhwa, N., Garg, R., Jacobs, D.E., Feldman, B.E., Kanazawa, N., Carroll, R., Movshovitz-Attias, Y., Barron, J.T., Pritch, Y., Levoy, M.: Synthetic depth-of-field with a single-camera mobile phone. ACM Trans. Graph. **37**(4), 1–13 (2018)
80. Sun, Z., Han, B., Li, J., Zhang, J., Gao, X.: Weighted guided image filtering with steering kernel. IEEE Trans. Image Process. **29**, 500–508 (2020). https://doi.org/10.1109/tip.2019.2928631
81. McGraw, T.: Fast bokeh effects using low-rank linear filters. Vis. Comput. **31**(5), 601–611 (2015)

Fernando Galetto received his B.Eng. degree in electronics engineering from National Technological University, Cordoba, Argentina in 2015, and the M.Eng degree in electronics engineering in 2019 from La Trobe University, Melbourne, Australia, where he is currently working toward the Ph.D. His research interests include computer vision and underwater image processing.

Guang Deng currently a Reader and Associate Professor of electronic engineering with La Trobe University, Melbourne, VIC, Australia. His current research interests include Bayesian signal processing, lossless image compression and generalized linear systems.