Xtended Fast Defocus map estimation from a single image

Aditya Kora
Department of Electronics and Communication
Indian Institute of Technology Roorkee
Enrollment Number: 22116007

Barath Chandran.C
Department of Electronics and Communication
Indian Institute of Technology Roorkee
Enrollment Number: 22116026

Aditya Innani
Department of Electronics and Communication
Indian Institute of Technology Roorkee
Enrollment Number: 22116006

Pranjal
Department of Electronics and Communication
Indian Institute of Technology Roorkee
Enrollment Number: 22116070

***Abstract*—This paper is a part reproduction and a part extension of Fast Defocus Map Estimation(FDME) a novel and efficient algorithm for generating a defocus map from a single image. Traditional approaches often rely on pixel-level propagation of a sparse depth map, which while being effective, is computationally expensive. To overcome this, FDME introduced a framework that leverages super pixelation and transductive inference to propagate the defocus value. By dividing the image into super-pixels using SLIC (Simple Linear Iterative Clustering), FDME significantly reduces computational costs without compromising performance. The original paper states that transductive inference even enhances the obtained defocus map by evaluating the similarity between superpixels. We propose that substituting SLIC with SNIC (Simple Non-Iterative Clustering) could improve both the performance and inference time of FDME. This study explores the mathematical foundations behind gradient-based sparse defocus map estimation, implements each component of FDME, and compares the performance of SLIC and SNIC.**

I. Introduction

When we take a photo, we are converting a three-dimensional scene into a two-dimensional image. Although it seems we have lost the third dimension of depth, depth is actually preserved as defocus blur in the image. When a 3D point aligns with the camera's focal plane (the analogous of a focal point in 3d) it appears sharp because the light rays converge perfectly on the sensor. Points outside the focal plane appear blurred, this phenomenon is known as defocus blur or depth of field effect. This blur is seen on the sensor as the circle of confusion (CoC), whose diameter c is proportional to the distance of the 3D point from the focal plane. This diameter c, the distance from the focal plane, and the resulting blur level are highly correlated. By leveraging this correlation researchers developed methods for defocus blur estimation, which plays a significant role in monocular depth map computation (depth map estimation from a single image). Beyond depth estimation, defocus maps have diverse applications in image editing and enhancement, including refocusing, segmentation, bokeh effect, background de-colorization, and salient region detection.

Research on defocus map estimation has evolved over time, due to advancements in computational methods and imaging technologies. Early techniques, such as Pentland (1987) [1] relied on handcrafted features and statistical models. These approaches utilized cues like gradient sharpness and frequency domain analysis to estimate the level of defocus. In recent times deep learning has further revolutionized the field, with works like Yan et al. (2016) [2] showcasing CNN-based approaches that significantly improved spatial and contextual understanding for defocus map estimation. A significant contribution to this area was the paper Zhou et al.(2011) [**?**], in which the authors propose a simple yet effective method for recovering the defocus map from a single image by re-blurring the already defocused image using a Gaussian kernel and calculating the gradient ratio between the original and re-blurred images, they estimate the defocus amount at edge locations. The sparse defocus map is then propagated across the entire image through Laplacian Matting. However, Laplacian Matting and other methods introduced after it are computationally expensive which limit their practical applicability. This high computational cost motivates the need for a more efficient way to propagate sparse defocus maps without compromising performance.

Superpixelation was introduced as an alternative to grid-based pixel representation to group pixels into meaningful regions that emphasize boundaries. A significant advancement in this field was the Simple Linear Iterative Clustering (SLIC) algorithm introduced by R.Achanta(2011) [5]. FDME(2017) [3] improved upon the computationally expensive process of pixel-level depth propagation by dividing the image into superpixels using SLIC and propagating depth at the superpixel level. However, since SLIC involves K-means clustering, it is fundamentally iterative in nature. To address this limitation, R. Achanta(2017) [6] also proposed Simple Non-Iterative Clustering (SNIC), which uses a priority queue and would only need to attend to each pixel once.

II. Our Approach Used

Gradient-based defocus estimation methods generally involve two main phases: estimating sparse defocus blur at edges and propagating that information across the image. For the sparse defocus blur estimation, we utilize the technique introduced by Zhou et al. which has been used in various earlier studies.FOr defocus propagation, many previous works have relied on pixel-level propagation. However, pixel-level defocus information is not always neccessary for practical tasks such as foreground/background segmentation or salient region detection. To address this Fdme introduced a framework that leverages super-pixelation and transductive inference. In this work, we will closely follow the same framework but will use SNIC for super-pixelation instead of SLIC.

III. Implementation

We developed XFdme using Python, utilizing resources from Kaggle. The code is structured as a Python Notebook, and we created a deployable GUI with Streamlit. You can find the code on GitHub.

A. *Sparse defocus blur estimation*

*1) **Edge detection** :* To measure defocus blur at edge locations, we apply the Canny edge detector on the image to extract a set E of edge pixels. The Canny method involves Gaussian smoothing, non-maximum suppression, and double thresholding to ensure precise and reliable identification of edge pixels, which serves as the foundation for estimating defocus blur. In our implementation, we utilize Python's `cv2.Canny` function from the OpenCV library to directly apply the Canny edge detection algorithm.

*2) **Sparse Defocus blur calculation**:* We follow the gradient-based defocus map estimation as mentioned in Zhou et al. An image is blurred with a known Gaussian kernel, and the gradient magnitudes of the original edge pixels obtained using canny edge detection and the re-blurred version are compared by calculating their ratio. This ratio reaches its peak at the edge positions. By utilizing this maximum value, the level of defocus blur at the edge locations of the entire image can be determined.The Mathematical proof is as follows:-

To estimate the defocus blur at edge locations, the original paper focused on step edges, as they represent the primary edge type in natural images. Consequently, our analysis in this paper is also restricted to step edges. An ideal step edge with edge located at $x = 0$, amplitude A and offset B can be mathematically modeled as:

$$f(x) = A.u(x) + B, \tag{1}$$

Defocus blur can be described as the result of convolving a sharp image with a point spread function (PSF). The PSF is approximated using a Gaussian function, where the standard deviation σ quantifies the extent of defocus blur and is directly related to the diameter of the circle of confusion (CoC), denoted as c. A blurred edge is represented as:

$$i(x) = f(x) \otimes g(x, \sigma),$$

where $\otimes$ denotes the convolution operation

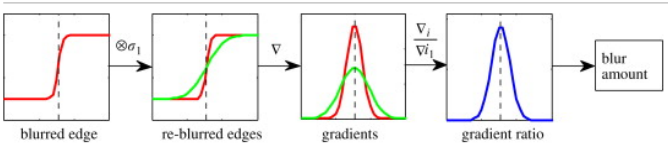

Fig. 1. 1D defocus through gradient

We first describe the blur estimation method for 1D case and then extend it to 2D image. The gradient of the re-blurred edge is

$$\nabla \tilde{i}_1(x) = \nabla \left((f(x) \otimes g(x, \sigma)) \otimes g(x, \sigma_0) \right) \tag{2}$$

where σ_0 is the standard deviation of the re-blur Gaussian kernel. From 1 we can say that

$$\nabla \tilde{i}_1(x) = \nabla \left((Au(x) + B) \otimes g(x, \sigma) \otimes g(x, \sigma_0) \right), \tag{3}$$

since we take $g(x, \sigma)$ to be a Gausian and convolving two Gausian functions gives us another Gausian function.

$$= \frac{A}{\sqrt{2\pi(\sigma^2 + \sigma_0^2)}} \left(-\frac{x}{\sigma^2 + \sigma_0^2} \right) \exp\left(-\frac{x^2}{2(\sigma^2 + \sigma_0^2)} \right), \tag{4}$$

The gradient magnitude ratio between the original and re-blurred edges is

$$\frac{|\nabla i(x)|}{|\nabla \tilde{i}_1(x)|} = \sqrt{\frac{\sigma^2 + \sigma_0^2}{\sigma^2}} \exp\left(-\left(\frac{x^2}{2\sigma^2} - \frac{x^2}{2(\sigma^2 + \sigma_0^2)} \right) \right). \tag{5}$$

This ratio is maximum at the edge location ($x = 0$) and the maximum value is given by

$$R = \frac{|\nabla i(0)|}{|\nabla \tilde{i}_1(0)|} = \sqrt{\frac{\sigma^2 + \sigma_0^2}{\sigma^2}}. \tag{6}$$

From 2 we can notice that the edge gradient depends on both the edge amplitude A and blur amount σ, while the maximum of the gradient magnitude ratio R eliminates the effect of edge amplitude A and depends only on σ and σ_0. Thus, given the maximum value R at the edge locations, the unknown blur amount σ can be calculated using

$$\sigma = \frac{1}{\sqrt{R^2 - 1}} \sigma_0. \tag{7}$$

The procedure we followed for a 2D image is as follows:

1) 1.Edge Detection: Apply Canny edge detection on the grayscaled image to obtain the edges of the image.

Fig. 2. Edge Detection.Original image-Left,Gray Scale-Middle, Canny Edge-Right.

2) 2.Gausian & Gradient:Apply a 2D isotropic gausian filter on the image,and find the gradient of the original image and the smoothened image at the edges.

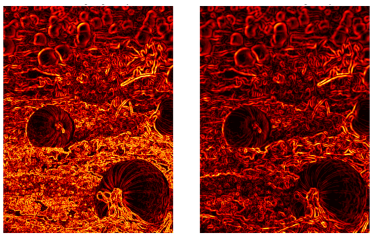

Fig. 3. Gradient of original-left and smoothened-right.We can observe that the smoothened image shows a lesser magnitude(gradient) at the edges.

3) 3.Gradient ratio: We find R the ratio of the gradient between the original image and the smoothened image at the edges(6).
4) 4.Sparse Defocus Map generation: We apply (7) on the values of R at each edge pixel to get the values of Defocus for all edges.

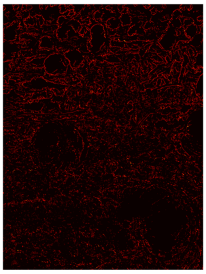

Fig. 4. Sparse Depth Map: We can observe that the intensity decreases from top to bottom as the distance from the camera decreases.

B. *Defocus Blur Propagation*

*1) **Superpixelation**:* A Superpixel refer to a group of pixels that share similar characteristics, such as color or texture, and represent a meaningful region in an image. We divide the image into superpixels and assign the same value of defocus across a superpixel at the end. We compare two Superpixelation alogorithms SLIC(Simple Linear Iterative Clustering) and SNIC(Simple Non-Iterative CLustering).SLIC treats superpixel generation as a k-means clustering problem in the 5D CIELAB space of pixel color and spatial coordinates.SNIC uses a priority queue to grow the superpixel clusters from dynamically updated centroids on the basis of the distance from these centroids in the CIELAB space. Every pixel is added to the queue and popped out of the queue only once making this algorithm non-iterative.

Fig. 5. Image after applying SLIC-Middle,SNIC-Right. The mean color of each super pixelation is shown for visualizing the blocks.We can observe that SNIC has better boundary adherence.

From the superpixels obtained from SLIC/SNIC we obtain a NxN weight matrix W where each w_{ij} is based on the feature similarity between the corresponding two superpixels in the $YCbCr$ space. This Gaussian weighting is based on the Euclidean distance between their mean color features, with smaller distances indicating greater similarity. We initially only want the weight to be non zero for adjacent super pixels which share a common edge,hence we model a weighted graph $G = (S, E, w)$.

*2) **Transductive Inference**:* The weight matrix W we obtained only has non-zero values for adjacent superpixels i.e it only captures the local relationships of feature similarity. But we need to know the global relationships between superpixels. If A and B are adjacent and W_{ab} is non zero, B and C are adjacent W_{bc} is non zero then W_{ac} must also be non zero even if a and c are not adjacent. To capture these transductive similarities we use the NxN affinity matrix A defined by.

$$A = (D - \gamma W)^{-1} I \tag{8}$$

where D is the diagonal matrix with each diagonal entry equal to the row sum of W, γ is a parameter in (0, 1), and I is the NxN identity matrix.

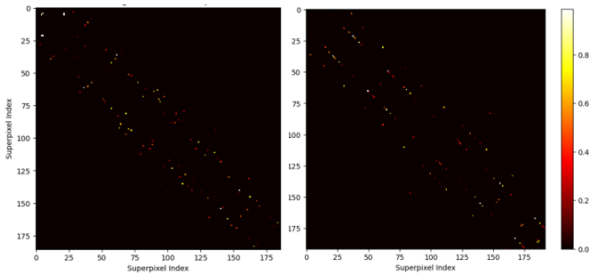

Fig. 6. Weight matrix W:SLIC Left,SNIC Right

*3) **Defocus Propogation**:* For every superpixel S_i we find an initial defocus blur by taking the median of the defocus values of all the edge pixels in that superpixel. The values are stored in a Nx1 vector.

$$f_{s_i} = \text{median}_{x \in E_i}\{f_x\}, \tag{9}$$

Taking the median of the defocus values makes the process robust to outliers in the sparse depth map. Now to include the transductive information we find the matrix product of the existing Nx1 vector and the NxN affinity matrix.

$$f_{s_i} = \hat{A} \cdot \begin{bmatrix} f_{s_1} \\ f_{s_2} \\ \vdots \\ f_{s_N} \end{bmatrix}$$

The resulting Nx1 vector gives us the final defocus value for each superpixel.

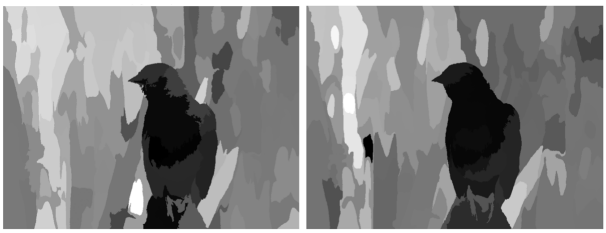

Fig. 7. Dense Defocus Map: SLIC-Left, SNIC-Right. We can observe that SNIC has more well defined distinction between foreground and background.

IV. Limitation

The disadvantage of directly using the previously implemented method is that the algorithm tends to predict areas with shadows as being defocused, which lightens those regions in the defocus map. To avoid this error, we can modify the approach to ignore these shadowed regions during the initial implementation. Later, we can incorporate those regions back into the final results, resulting in a more accurate defocus map value.

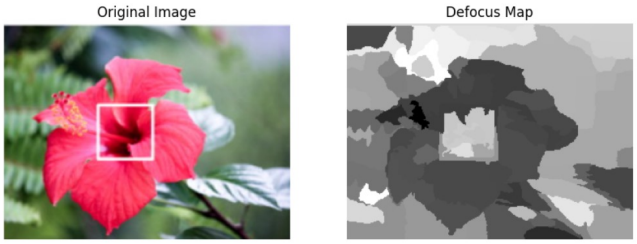

Fig. 8. Lightening in shadows

V. Conclusion

We were able to successfully implement the code for FDME and obtain a dense depth map from any given single image. We found that using SNIC instead of SLIC could improve overall performance of the framework Fig.7 as SNIC has better boundary adherence Fig.5.

VI. Contributions

1) Barath Chandran.C : Ideation, Report, GUI, Code: SNIC.
2) Aditya Kora : Ideation, Report, GUI, Code: Defocus Propagation.
3) Aditya Innani : Code: Sparse Defocus Map.
4) Pranjal Gautam : Code: Defocus Propagation.

References

[1] A. P. Pentland, "A New Sense for Depth of Field," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 4, pp. 523-531, July 1987, doi: 10.1109/TPAMI.1987.4767940.

[2] R. Yan, L. Shao, and S. Zhuo, "Defocus map estimation from a single image," *Pattern Recognition*, vol. 44, no. 9, pp. 1852-1858, 2011, doi: 10.1016/j.patcog.2011.03.009.

[3] D.-J. Chen, H.-T. Chen, and L.-W. Chang, "Fast defocus map estimation," *2016 IEEE International Conference on Image Processing (ICIP)*, Phoenix, AZ, USA, 2016, pp. 3962-3966, doi: 10.1109/ICIP.2016.7533103.

[4] S. Zhuo and T. Sim, "Defocus map estimation from a single image," *Pattern Recognition*, vol. 44, no. 9, pp. 1852–1858, 2011, doi: 10.1016/j.patcog.2011.03.009.

[5] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC Superpixels Compared to State-of-the-Art Superpixel Methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274-2282, Nov. 2012, doi: 10.1109/TPAMI.2012.120.

[6] R. Achanta and S. Süsstrunk, "Superpixels and Polygons Using Simple Non-Iterative Clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.