Supplementary Material for Depth of Field Rendering Using Multilayer-Neighborhood Optimization

Benxuan Zhang, Bin Sheng, Ping Li, and Tong-Yee Lee, *Senior Member, IEEE*

1 Impacts on High Frequency Images

Fig. 1 shows examples of our method applied on high frequency images, the image on the left is the original all-in-focus image, the image in the middle is the corresponding depth map and the image on the right is the rendered DOF result using our method. From the images, we can see that our method can apply to high frequency image and patch matching is also better used in such cases. Besides the rendered results in Fig. 16 in Section 4.4 of our paper shows that our method can also apply to low frequency and average frequency images. Patch matching not only estimates color intensity but also depth of pixels. If we use simple linear interpolation method, we can only estimate the intensity of hidden pixels but with PatchMatch algorithm we can also estimate depth. Besides, simple diffusion interpolation method does not consider the texture of surface and will result in flat interpolation surface while using PatchMatch algorithm can fill the missing intensities of holes more consistent than diffusion interpolation.

2 Depth of Field

The relationship of R_{CoC} and depth to various aperture sizes is shown in Fig. 2. First of all, we need to use the depth information to calculate the radius of CoC. While our method is based on the assumption that a depth map of the same resolution as the image is known before, thus acquisition of the depth information is not a concern in this paper. It can be obtained via stereo vision, Kinect and time-of-flight camera. Furthermore, we demonstrate that using inpainted depth map can produce realistic DOF rendering effect as shown in Section 4. In this paper, the depth values are normalized in the range 0-255 and stored in a grayscale image. Thus, a small depth value indicates that the pixel is near, whereas a large depth values indicates it is far.

- *B. Zhang and B. Sheng are with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China. E-mail: shengbin@sjtu.edu.cn.*
- *P. Li is with the Faculty of Information Technology, Macau University of Science and Technology, Macau 999078, China. E-mail: pli@must.edu.mo.*
- *T.-Y. Lee is with the Department of Computer Science and Information Engineering, National Cheng-Kung University, Tainan 70101, Taiwan. E-mail: tonylee@mail.ncku.edu.tw.*

Manuscript received 08 Jul. 2018; revised 06 Jan. 2019; accepted 18 Jan. 2019.

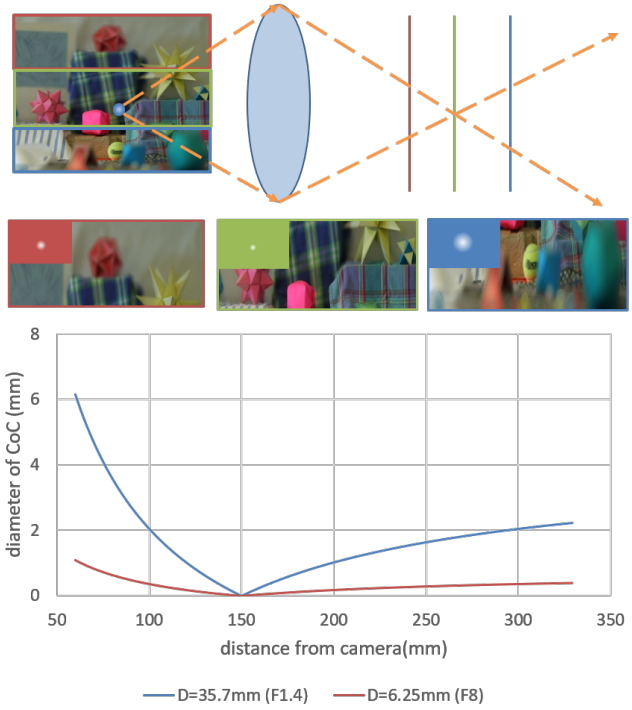

Fig. 2. Relation of R_{CoC} and depth to various aperture sizes. The figure shows the relation between distance and size of CoC in three different situations: before focused region, within focus region and after focused region. The parameters of this figure are as follows: focused depth d_f is 150 mm, focal length f is 28 mm, aperture diameter D is 35.7 mm (blue line, F-number: 1.4) and 6.25 mm (red line, F-number: 8).

The blurriness of every pixel is determined by the CoC. Its size can be computed from the thin lens camera model shown in Fig. 3. In the thin lens camera model, light beams originating from a given point in a scene go through the lens aperture, get refracted, and then converge to a point. However, this convergence point may not lie in the image plane. Instead, it may map to a circle in the image plane (i.e., the CoC). In our approach, we categorize the pixels in three cases (foreground, in-focus, and background) according to their relative depth position with the focus depth. We first estimate the color intensity of the blocked pixels using adaptive PatchMatch method. We make an estimation of blocked pixels by finding most similar patch in

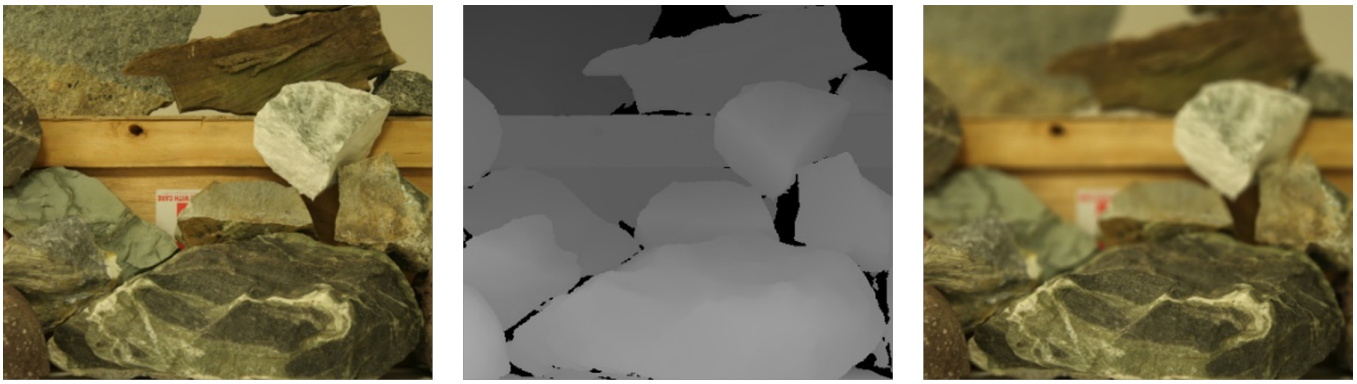

Fig. 1. Example of high frequency color image, depth map, and rendered DOF result. (a) Original all-in-focus image, (b) corresponding depth map, and (c) rendered DOF image.

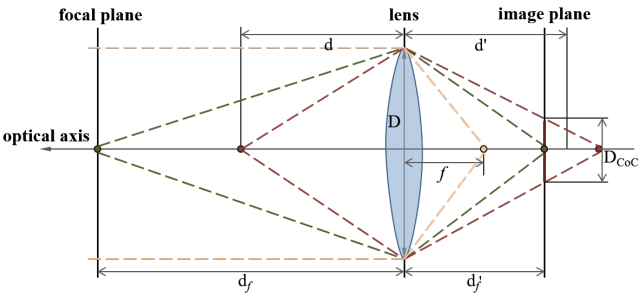

Fig. 3. Thin length model. The object out of the focal plane is projected to a circle in the image plane (i.e., the CoC). The diameter of the CoC, D_{CoC}, depends on the distance of the object from the lens, or depth d, the focused depth d_f, lens aperture size D, and focal length f.

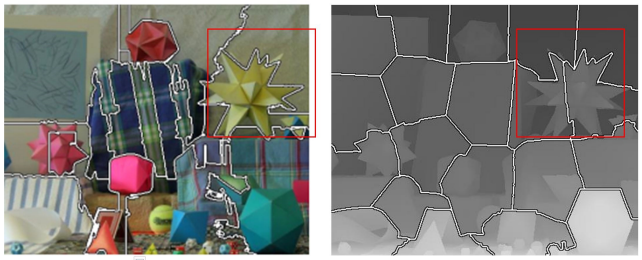

Fig. 4. This picture illustrates the segmented result of applying SLIC algorithm on color image and depth image separately. The separated result in red rectangle shows that applying SLIC algorithm on color image is better than depth image directly.

image while preserving color consistency, depth consistency and structural consistency. and Then we apply multilayer-neighborhood optimization to the "complete" image to get the blur result thus handle the color leakage and blur discontinuity problem.

Firstly, we show the effects of applying SLIC algorithm on color image and depth image as shown in Fig. 4. Results have shown that just applying SLIC algorithm on color image gives us better result especially for the segmented result in the red region. Then, we demonstrate the effects of applying recursive bilateral filtering approach to reduce color leakage problem of foreground blur. As you can see

Fig. 5. Experimental results of applying weight map directly and applying our iterated framework. (a) Use W_{max} to render the result, and (b) use iterative bilateral filtered weight map to render the result.

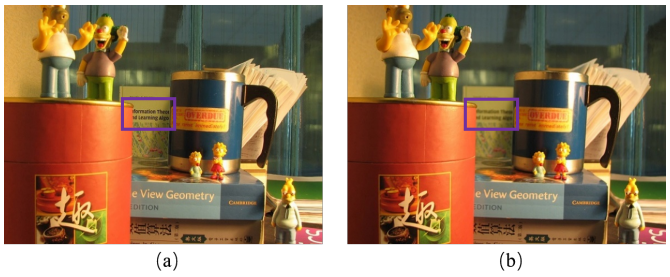

Fig. 6. DOF post-processing technique applied in image editing which makes the text clear.

from the pictures in Fig. 5, the left one is the result by applying W_{max} directly on the picture which results in visible color leakage artifacts while the image on the right is obtained by applying iterative bilateral filtering result. Thus, our framework can reduce color leakage artifacts. For the third contribution, our method can achieve in real time and we have explained the accelerated part in our algorithm with the help of GPU. Besides, as discussed in the last revision, we have added experiments on demonstrating that our algorithm can be applied in various area. Thus, it has great potential on application areas.

3 GPU Implementation

First of all, we discuss an iterative EM algorithm used for estimating blocked pixels in Section 3.2.2 of our paper.

Algorithm 1 GPU Implementation for PatchMatch Kernel

1: **Procudure**: Find similar patch
2: Get the block index BID and thread index TID;
3: Cycle through each pixel of source image S using BID and TID {
4: Random initialize NNF for q;
5: Propagation NNF to other regions;
6: Half the step size L;
7: }

Algorithm 2 GPU Implementation for Filtering Process

1: **Procudure**: Filter the image using weight map
2: Iterate through each pixel of source image S and weight map W with depth map M {
3: Apply Eq. (12) and Eq. (13) to scatter pixel intensities;
4: }

During each iteration, we accelerate the computation for Expectation step and Maximal step separately. In Expectation step, we apply Generalized PatchMatch algorithm in parallel computation as methods describe in [1], [2]. We implement approximate k-nearest neighbor PatchMatch method based on jump flooding algorithm which can be fully parallelized at patch-level. The GPU implementation of this algorithm achieves up to 100 times speedup over its CPU implementation. As shown in [3], we can apply bilateral filter in constant time. Thus, iteration of bilateral filter on weight map can be finished in constant time. The pseudocode of this parallel implementation is shown in Algorithm 1. For the algorithm described in Section 3.2.3 of our paper, we first iteratively filter the weight map by Eq. (10) of our paper. As shown in previous paper, bilateral filter can be executed in O(1) time using proposed algorithm. Thus, we can get our estimated potential weight map in short time. Finally, we need to filter the image using Eq. (12) and Eq. (13) of our paper. We then could use CUDA implementation to accelerate this process as shown in the pseudo-code in Algorithm 2.

4 More Applications

The proposed method can be used in traditional DOF rendering which can be further used in image editing task such as make the text clear or blur based on various purposes. Our proposed DOF rendering method can be used in realistic image rendering or can be used as a hint to transfer the focus of viewers. Besides, it can further be used in image editing and information visualization which makes important data clear and blur the unimportant data. Kosara *et al.* [4] proposed Semantic Depth of Field for focus-and-context display of information. Besides, DOF rendering can be used in practical medical area such as eye surgery. Besides, we have added one figure to explain one typical application of DOF effects. With DOF post-processing technique, we can edit the image and clear and blur the text on the image. As shown in Fig. 6, the image on the left is the original clear picture while the image on the right is the text-blurred image, see the purple rectangle region for details. Thus, our method can be further used in visualization applications.

As shown in Fig. 7, the images in the first row are original images in a video. The images in the second row are inpainted depth map while the bottom row contains the original depth image captured by Kinect. Our results have shown that it can achieve comparable DOF rendering results thus make proposed techniques possible in real life applications. The bottom rows are original depth map captured by Kinect. There are a lot of holes in picture which are denoted as black pixels. Thus, we use inpainting method to fill in the holes in the image to get the depth maps in the second rows. There are artifacts in inpainted depth map as you can see in the screen display part. But in most part of the images, it fills appropriate depth value for holes. Thus, we can focus on the cans and blur other areas in above figure. Finally, we show more results here in Fig. 8. The results are obtained using depth map acquired from Kinect. We tried to apply our framework on video, but video processing is the limitation of our method. The video is produced using fixed focus gap. Such method provides flexibility but might fail in video. And, the results are not satisfying because we need to adjust the focus range in each frame according to the depth of focused object. We will improve this method in the future by incorporating coherence of objects in videos or study a way for getting rid of using fixed focus range. Our method could be applied to video but still remains to be improved. In the future, we might automatically detect the object when we want to focus on and extract its depth information and apply our DOF rendering algorithm on it.

References

[1] P. Yu, X. Yang, and L. Chen, "Parallel-friendly patch match based on jump flooding," in *Advances on Digital Television and Wireless Multimedia Communications*, 2012, pp. 15–21.
[2] R. Giraud, V.-T. Ta, A. Bugeau, P. Coupé, and N. Papadakis, "SuperPatchMatch: An algorithm for robust correspondences using superpixel patches," *IEEE Trans. Image Process.*, vol. 26, no. 8, pp. 4068–4078, 2017.
[3] Q. Yang, S. Wang, and N. Ahuja, "Real-time specular highlight removal using bilateral filtering," in *ECCV*, 2010, pp. 87–100.
[4] R. Kosara, S. Miksch, and H. Hauser, "Semantic depth of field," in *IEEE Symposium on Information Visualization*, 2001, pp. 97–104.

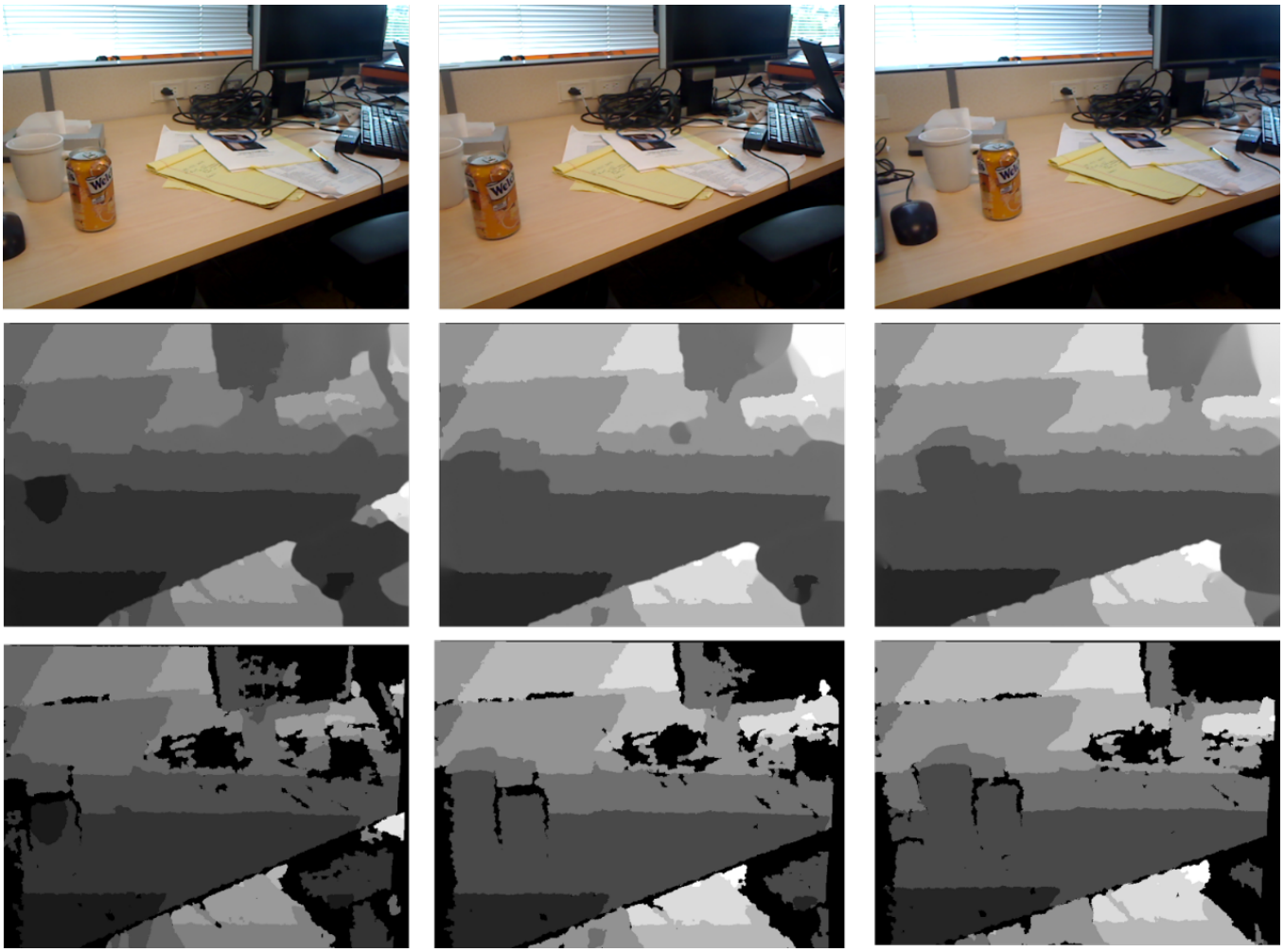

Fig. 7. This figure shows the DOF results using inpainted depth map for a consecutive video. From top to bottom are: original color images, inpainted depth maps, and original depth maps with holes acquired by Kinect.

Fig. 8. Experimental results of video. (a) Original image, (b) depth map, and (c) our results focusing on the tree in the middle.