Plenoptic Image Motion Deblurring

Paramanand Chandramouli, Meiguang Jin, Daniele Perrone, and Paolo Favaro, *Member, IEEE*

***Abstract*— We propose a method to remove motion blur in a single light field captured with a moving plenoptic camera. Since motion is unknown, we resort to a blind deconvolution formulation, where one aims to identify both the blur point spread function and the latent sharp image. Even in the absence of motion, light field images captured by a plenoptic camera are affected by a non-trivial combination of both aliasing and defocus, which depends on the 3D geometry of the scene. Therefore, motion deblurring algorithms designed for standard cameras are not directly applicable. Moreover, many state of the art blind deconvolution algorithms are based on iterative schemes, where blurry images are synthesized through the imaging model. However, current imaging models for plenoptic images are impractical due to their high dimensionality. We observe that plenoptic cameras introduce periodic patterns that can be exploited to obtain highly parallelizable numerical schemes to synthesize images. These schemes allow extremely efficient GPU implementations that enable the use of iterative methods. We can then cast blind deconvolution of a blurry light field image as a regularized energy minimization to recover a sharp high-resolution scene texture and the camera motion. Furthermore, the proposed formulation can handle non-uniform motion blur due to camera shake as demonstrated on both synthetic and real light field data.**

***Index Terms*— Plenoptic camera, light field image, motion blur, blind deconvolution.**

I. Introduction

In THE past few years, plenoptic cameras have entered into the realm of consumer photography [1], [2]. These cameras are equipped with 3D reconstruction and digital refocusing capabilities, not possible in traditional devices. This has led to an increased interest in the scientific community in high-quality reconstructions from light fields. As these commercial cameras are portable, camera shake is sometimes unavoidable and may result in blurry light field images. Similarly, moving objects can also cause the images to appear blurry.

Until now most of the research works on light field (LF) image processing have focused on depth estimation and super-resolution rather than motion deblurring. In contrast, motion blur in conventional cameras has been widely studied and current methods achieve remarkable results (see, for instance, [9], [16], [28], [48]). Unfortunately, the imaging mechanism of a conventional camera and LF camera are quite different. Due to the additional microlens array between the main lens and the sensors in the LF camera, an LF image consists of a rearranged set of views of the scene that are highly under-sampled and blurred [30]. Consequently, motion deblurring methods that are applicable to conventional images cannot be adapted in a straightforward manner.

In this paper, we propose a motion deblurring scheme for images captured from a microlens-array based light field camera. From a single motion blurred LF image, we estimate the high-resolution sharp scene texture. An LF image can be related to the scene texture through a space-variant point spread function (PSF) which models the image formation mechanism. In addition, the sharp scene texture is related to the blurry texture in terms of a motion blur PSF. Modeling the LF image generation by taking into account these effects turns out to be very computationally intensive and memory inefficient. However, we show that, for constant depth scenes, it is possible to describe a motion blurred light field image as a linear combination of parallel convolutions (see sec. III-B). As a result, the model is extremely computationally and memory efficient. To model general 3D scenes, we consider them to be composed of layers having constant depth. We develop a fast GPU implementation that estimates high resolution scene texture from light field image for each depth layer. For each layer, we estimate the space-variant motion blur PSF through blind deconvolution. Finally, we solve for the sharp scene texture within an energy minimization scheme that accounts for alignment and distortion correction. We demonstrate the performance of our algorithm on real and synthetic images.

Manuscript received February 19, 2017; revised July 5, 2017 and October 30, 2017; accepted November 6, 2017. Date of publication November 17, 2017; date of current version January 12, 2018. The work of M. Jin and P. Favaro was supported by the Swiss National Science Foundation under Grant 200021_153324. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Gene Cheung. *(Corresponding author: Paramanand Chandramouli.)*

P. Chandramouli is with the University of Siegen, 57076 Siegen, Germany (e-mail: paramanand@gmail.com).

M. Jin and P. Favaro are with the Institut für Informatik, University of Bern, 3012 Bern, Switzerland (e-mail: jin@inf.unibe.ch; paolo.favaro@inf.unibe.ch).

D. Perrone is with Chronocam, 75012 Paris, France (e-mail: dperrone@chronocam.com).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the author. The supplementary files includes tables and data relevant to the main paper. The total size of the file is 15.4 MB. Contact paramanand@gmail.com for further questions about this work.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2017.2775062

II. Related Work

Since plenoptic image motion deblurring is related to image blind deconvolution as well as light field reconstruction, we provide a brief overview of research in these topics.

A. Single Image Motion Deblurring

Motion deblurring involves the joint estimation of a sharp image and a blur kernel. Because of its ill-posedness, motion deblurring algorithms enforce suitable priors for the sharp image and the blur kernel. With these priors, the motion blur kernel is initially estimated and finally the sharp image is recovered through non-blind deblurring [19], [27], [41], [48]. Popular choices of prior include Laplace distribution [28], total variation (TV) regularization [14] or L_0 regularization [50]. Other methods encourage sharp edges by using a shock filter [16], [48] or a dictionary of sharp edges [37]. For the blur function, the choices for prior include Gaussian distribution [48], or a sparsity-inducing distribution [19]. Levin *et al.* [27] have shown that a joint MAP estimation of the sharp image and blur function cannot lead to a correct estimate for a wide range of image priors. They show that marginalizing the sharp image and performing a MAP estimation on the blur function alone can correctly estimate the blur kernel. The marginalization of the sharp image is however computationally challenging, and therefore various approximations are used in practice [9], [19], [28]. Despite the theoretical analysis of Levin *et al.* [27], many methods successfully use a joint MAP estimation and achieve state of the art results [16], [33]. Recently, Perrone and Favaro [32] have clarified this apparent inconsistency. They confirmed the results of Levin *et al.* [27] and showed that particular implementation details make many methods in the literature work and that they do not in practice solve a joint MAP estimation.

The aforementioned deblurring methods assume uniform blur across the image plane. Techniques exist that address non-uniform blur due to rotational camera motion by introducing additional dimensions in the motion blur PSF [20], [23], [38], [45]. Hirsch *et al.* [21] propose an efficient implementation scheme wherein non-uniform blur can be modeled through piece-wise uniform blurs while still retaining the global camera motion constraint. Nonetheless, it has been observed in [26] that in realistic scenarios, the performance of algorithms that explicitly handle rotational motion are not necessarily better than that of the methods that assume uniform blur. Motion blur can also vary within an image because of parallax in 3D scenes. Techniques proposed in [35] and [49] address this problem when there are two observations available and the camera motion is restricted to in-plane translations. While in [31], a deblurring scheme which addresses non-uniform blur for bilayer scenes using two blurred images is proposed, Hu et al. address the problem with a single observation [22]. Other methods attempt to solve general space-varying motion deblurring by estimating locally uniform blur and carefully interpolate them to cope with regions having poor texture [24], or by iteratively employing uniform deblurring and segmentation algorithms [25].

B. Light Field Capture

For a comprehensive discussion on light field acquisition, processing and display, one can refer to the recent survey in [47]. In this paper, we restrict our attention to microlens array-based plenoptic cameras. The basic lenslet-based plenoptic camera was first developed by Adelson and Wang for inferring scene depth from a single snapshot [8]. The portable design by Ng et al. with the use of microlens arrays triggered the development of handheld light field cameras [30]. The main drawback of image reconstruction in this camera was its limited spatial resolution. To overcome this problem, Lumsdaine and Georgiev designed the focused plenoptic camera wherein the microlenses were focused at the image plane of the main lens, thereby enabling rendering at higher resolution for certain depths [29]. Perwaß and Wietzke proposed a further modification in the design which included microlens arrays with microlenses having different focal lengths [34]. Recently, Wei et al. proposed to introduce irregularity in the lens design to achieve better sampling of light fields [44].

C. Calibration, Depth Estimation and Super-Resolution

Since plenoptic images suffer from loss of spatial resolution, various algorithms have been proposed to super-resolve scene texture. In [29], the authors propose to render high resolution texture directly from light field data by projecting the LF image onto a grid at higher resolution than the number of microlenses. In [11], Bishop and Favaro model the image formation through geometric optics and propose a PSF-based model to relate the LF image and high resolution scene texture along with the camera parameters and scene depth. They propose a two-step procedure to estimate the scene depth map using view correspondences and thence the high resolution texture. Wanner and Goldlücke propose a technique to accurately estimate disparity using an epipolar image representation and then super-resolve 4D light fields in both spatial and angular directions [43]. Broxton et al. propose a 3D-deconvolution method to reconstruct a high resolution 3D volume using a PSF-based model with applications to light field microscopy [12]. We also found out that this work suggests a fast computational scheme that might be similar to the one proposed here. However, due to lack of details (a very short explanation is given in a paragraph in [12, Sec. 3.4]) we cannot verify this. However, even in the presence of a similarity, our and this scheme were developed simultaneously [5]. Moreover, our scheme includes the case of motion blur.

Recently, many works have been published that deal with calibration, depth estimation and super-resolution [15], [17], [39], [40]. These methods are capable of dealing with practical issues in cameras such as noise, misalignments, vignetting etc. During the review of this paper, a light field deblurring algorithm was proposed by Srinivasan *et al.* [36]. They consider that the camera undergoes 3D translations and solve for camera motion and the sharp light field from a blurry light field image. They show that the estimation of scene geometry is not necessary for deblurring. The main difference between their paper and our work is that we aim to recover the high-resolution scene texture while they attempt to deblur the 4D light field function. The spatial resolution of the recovered light field in [36] corresponds to microlens resolution, which is very much coarser when compared to the texture that we recover in this paper. The difference in resolution is apparent in an example shown in Fig. 5.

Contributions: The contributions of our work can be summarized as: i) We introduce the problem of motion deblurring of light field images from a plenoptic camera and provide the first solution to it. ii) We propose a computationally and memory efficient imaging model for motion blurred LF images. iii) We will make publicly available a GPU implementation for recovering scene texture from light fields. iv) We solve a joint blind deconvolution and super resolution problem. v) We handle radial distortion correction and alignment within our energy minimization framework.

III. Imaging Model

In this section, we introduce notation and the image formation model for a motion blurred light field image. Initially, we describe the plenoptic image formation for constant depth scenario. We propose two approaches for fast implementation

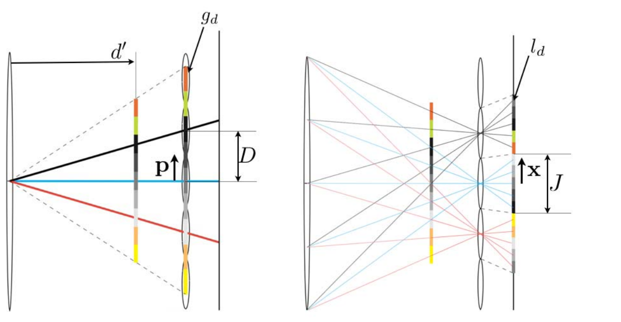

Fig. 1. LF image formation: (a) Projection of scene texture to conjugate plane. (b) LF image formation at the sensors through the microlens array.

of the imaging model. Finally, we describe our model to represent non-uniform motion blurred scenes consisting of different depth layers.

A. Plenoptic Image Formation

Figs. 1 (a) and (b) explain the image formation process in a plenoptic camera. Consider a scene with depth value d and scene texture denoted by g_d. Following [11], we consider g_d to be defined on the microlens array plane as shown in Fig. 1 (a). Note that defining the texture on the microlens plane is only a convention, and the texture could be defined on any other plane perpendicular to the optical axis. According to the thin lens model, due to the camera main lens, the scene is imaged at the conjugate plane inside the camera at a distance d' from the main lens. This is subsequently imaged by the microlens arrays resulting in the light field image l_d on the sensor plane.

Let $\mathbf{p}$ denote the coordinates of a point on the microlens array and $\mathbf{x}$ denote a pixel location on the sensor plane. We have $\mathbf{p} = [p_1\ p_2]^T$, and $\mathbf{x} = [x_1\ x_2]^T$, where p_1, p_2, x_1, x_2 are integers. A pixel of the LF image $l_d(\mathbf{x})$ is related to texture elements $g_d(\mathbf{p})$ through a space-varying PSF $h(\mathbf{x}, \mathbf{p})$ as

$$l_d(\mathbf{x}) = \sum_{\mathbf{p}} h_d(\mathbf{x}, \mathbf{p}) g_d(\mathbf{p}) \tag{1}$$

The PSF h_d depends on the depth value d and the parameters of the plenoptic camera. An explicit formula is also available in [11] and [12] and efficient code to implement it is available on our website.[1] Using the vectorial notation for the LF image $\mathbf{l}_d$ and the texture $\mathbf{g}_d$, we can write the imaging model as

$$\mathbf{l}_d = H_d \mathbf{g}_d \tag{2}$$

where H_d is a (typically very large) sparse matrix.

B. Efficient LF Image Generation

A direct implementation of eq. (1) is practically very challenging due to the memory and computation requirements. For instance, if the scene texture and the light field image were of the order of mega pixels, the number of elements necessary to store H_d would be of the order of 10^{12}. Although H_d is sparse, performing the sum and product calculation in eq. (1) would still be computationally intensive. We address these shortcomings by exploiting the structure of LF PSFs due to the periodicity of the microlens array lattice. We show two ways to arrive at exact and efficient implementations of LF image generation that are highly parallelizable: first, via parallel convolutions and second, through sparse matrix-vector products.

[1]http://www.cvg.unibe.ch/research/PlenopticDeblurring

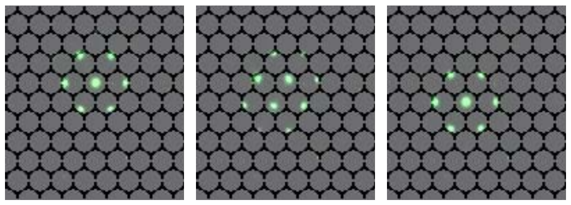

Fig. 2. Real images of point light sources from Lytro Illum camera: A point light source moves left and down in Figs. (a) to (c).

We first provide an intuitive explanation of the key idea that we exploit. Then, we present it formally. Fig. 2 (a) shows an image captured by a Lytro Illum camera when a green colored point light source was placed in front of the camera. To better visualize the periodicity we synthetically overlaid the hexagonal microlens array structure and set gaps between microlenses to black. Small diagonal translations of the light source yield images as in Figs. 2 (b) and (c). Notice that the image in Fig. 2 (c), is identical to that in Fig. 2 (a) except for a translation of one microlens. Similarly, for every position of the point light source, there exist corresponding positions (on a discrete lattice) at which we observe repeated patterns.

To formalize this periodicity, we need to introduce some basic notation. First, for simplicity, we consider a rectangular microlens arrangement. The same formulation holds also for hexagonal arrangements as shown in sec. III-C. Suppose there are $J \times J$ pixels under each microlens in the sensor plane. A pixel location $\mathbf{x}$ can be written as $\mathbf{x} = \mathbf{k}J + \mathbf{j}$, where $\mathbf{j} = [j_1\ j_2]$, $j_1, j_2 \in [0, \ldots, J-1]$, and $\mathbf{k} = [k_1\ k_2]$, where k_1 and k_2 are integers specifying the microlens under which $\mathbf{x}$ falls. It should be noted that, while all those pixels with the same $\mathbf{j}$ correspond to a view or a sub-aperture image (an image from one camera in a camera array), those with the same $\mathbf{k}$ correspond to the image within a single microlens. We decompose the coordinates of texture $\mathbf{p}$ on the microlens plane in a similar fashion. If two microlenses are D units apart, then we write $\mathbf{p} = \mathbf{b}D + \mathbf{t}$, where $\mathbf{t} = [t_1\ t_2]$, $t_1, t_2 \in [0, \ldots, D-1]$, and $\mathbf{b} = [b_1\ b_2]$, where b_1 and b_2 specify the microlens coordinates. Note that D specifies the resolution at which the scene texture is defined. A larger value of D defines texture on a finer grid, and, vice versa, a smaller value a coarser grid.

Now consider two points $\mathbf{p}$ and $\mathbf{q}$ located in different microlenses in such a way that their relative position with respect to the microlens center is the same (*i.e.*, both have the same value of $\mathbf{t}$). Then, the PSFs $h_d(\mathbf{x}, \mathbf{p})$ and $h_d(\mathbf{x}, \mathbf{q})$ will be shifted versions of each another. Technically, this is because the PSF of the light field image is given by the intersections of the blur discs of the main lens and the microlens array [11]. If there is a shift in the position of the point light source that exactly corresponds to an integer times the microlens diameter, then the resulting intersection pattern will be identical to the original intersection pattern, but with a shift because of the regularity of the microlens array as seen in Fig. 2. Hence,

$$h_d(\mathbf{x}, \mathbf{p}) = h_d(\mathbf{x} - J\mathbf{y}, \mathbf{p} - D\mathbf{y}) \tag{3}$$

for any $\mathbf{y}$, where $\mathbf{y} = [y_1\ y_2]$, y_1 and y_2 are integers. Based on this periodicity, we develop two approaches for efficient LF image generation.

1) Convolution-Based Scheme: The first observation is that the components of the PSF h_d can be completely described by eq. (3) with $J^2 \times D^2$ kernels. Secondly, the spatial spread of every kernel will be limited because of the limited extent of the main lens blur disc. Thus, we have obtained a memory efficient representation of the PSF h_d. By replacing $\mathbf{p}$ by $\mathbf{b}D + \mathbf{t}$ in eq. (1), and using eq. (3) we get

$$l_d(\mathbf{x}) = \sum_{\mathbf{t}} \sum_{\mathbf{b}} h_d(\mathbf{x} - \mathbf{b}J, \mathbf{t}) g_d(\mathbf{b}D + \mathbf{t}) \tag{4}$$

Then, by expressing $\mathbf{x}$ as $\mathbf{x} = \mathbf{k}J + \mathbf{j}$, we get

$$l_d(\mathbf{k}J + \mathbf{j}) = \sum_{\mathbf{t}} \sum_{\mathbf{b}} h_d(\mathbf{k}J + \mathbf{j} - \mathbf{b}J, \mathbf{t}) g_d(\mathbf{b}D + \mathbf{t}) \tag{5}$$

Let $\hat{g_d}(\mathbf{b}, \mathbf{t}) \doteq g_d(\mathbf{b}D + \mathbf{t})$, $\hat{l}_d(\mathbf{k}, \mathbf{j}) \doteq l_d(\mathbf{k}J + \mathbf{j})$ and $\hat{h}_{d_\mathbf{j}}(\mathbf{k}, \mathbf{t}) \doteq h_d\,(\mathbf{k}J + \mathbf{j}, \mathbf{t})$ (*i.e.*, just a rearrangement). Then, for every value of $\mathbf{j}$, we have

$$\hat{l}_d(\mathbf{k}, \mathbf{j}) = \sum_{\mathbf{t}} \sum_{\mathbf{b}} h_d\left((\mathbf{k} - \mathbf{b})J + \mathbf{j}, \mathbf{t}\right) \hat{g_d}(\mathbf{b}, \mathbf{t}) \tag{6}$$

and finally

$$\begin{aligned} \hat{l}_d(\mathbf{k}, \mathbf{j}) &= \sum_{\mathbf{t}} \sum_{\mathbf{b}} \hat{h}_{d_\mathbf{j}}(\mathbf{k} - \mathbf{b}, \mathbf{t})\, \hat{g_d}(\mathbf{b}, \mathbf{t}) \\ &= \sum_{\mathbf{t}} \left(\hat{h}_{d_\mathbf{j}}(\cdot, \mathbf{t}) * \hat{g_d}(\cdot, \mathbf{t}) \right)(\mathbf{k}) \end{aligned} \tag{7}$$

where $*$ denotes the convolution operation. Eq. (7) indicates that we can arrive at the LF image by performing convolutions for every possible value of $\mathbf{t}$ and $\mathbf{j}$. Note that these convolutions are completely independent and therefore can be executed in parallel. Also, we need only $J^2 \times D^2$ PSFs that are denoted by $\hat{h}_{d_\mathbf{j}}(\cdot, \mathbf{t})$ for the LF image generation.

2) Sparse Matrix-Based Scheme: The periodicity in eq. (3) can also be written as

$$h_d\,(\mathbf{x} + J\mathbf{y}, \mathbf{p}) = h_d\,(\mathbf{x}, \mathbf{p} - D\mathbf{y}). \tag{8}$$

By setting $\mathbf{x}$ as $\mathbf{x} = \mathbf{k}J + \mathbf{j}$ and using eqs. (1) and (8), we get

$$l_d(\mathbf{k}J + \mathbf{j}) = \sum_{\mathbf{p}} h_d(\mathbf{j}, \mathbf{p} - \mathbf{k}D) g_d(\mathbf{p}). \tag{9}$$

Let $\hat{l}_d(\mathbf{k}, \mathbf{j}) \doteq l_d(\mathbf{k}J + \mathbf{j})$ and $\mathbf{p}' \doteq \mathbf{p} - \mathbf{k}D$. Then we have,

$$\hat{l}_d(\mathbf{k}, \mathbf{j}) = \sum_{\mathbf{p}'} h_d(\mathbf{j}, \mathbf{p}') g_d(\mathbf{p}' + \mathbf{k}D). \tag{10}$$

An efficient implementation of LF generation can be obtained from eq. (10). The term $\sum_{\mathbf{p}'} h_d(\mathbf{j}, \mathbf{p}')$ denotes summing over a particular row of the basic PSF matrix H_d (eq. (2)). Every row of the matrix H_d is quite sparse since a pixel in a light field image is related to only a small number of elements of the scene texture. Consequently, a sparse matrix consisting of only $J \times J$ sparse rows, corresponding to all possible values of $\mathbf{j}$, is sufficient to represent the PSF to render one microlens. The image within a single microlens (i.e., $\hat{l}_d(\mathbf{k}, \mathbf{j})$ for a particular $\mathbf{k}$) can be obtained through the product of this sparse matrix with the texture elements g_d. For images corresponding to other microlenses, one needs to apply only a translational shift $g_d(\mathbf{p}' + \mathbf{k}D)$ as seen in eq. (10). All these operations can be performed in parallel.

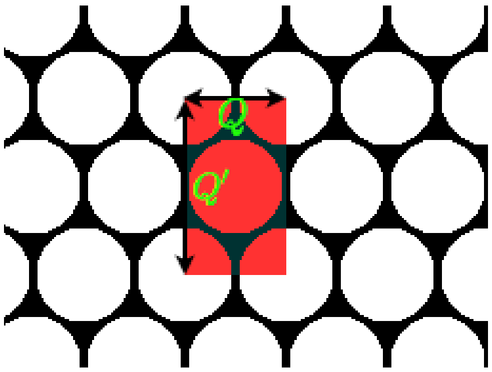

Fig. 3. Hexagonal arrangement of microlenses. Notice that the lattice is periodic with minimal period defined by the red tile.

C. Hexagonal Arrangement

In consumer plenoptic cameras, the microlenses are arranged on a hexagonal grid. Fig. 3 (a) shows an ideal hexagonal arrangement of the microlenses. The basis for our implementation schemes was that the intersection pattern of the main lens blur disc and the microlens blur disc repeats periodically as we traverse the microlens array plane. One can observe that the periodicity property holds when taking horizontal or vertical steps of Q and Q' pixels respectively, which match the dimensions of the red-shaded rectangular region in Fig. 3 (a). Similarly, as discussed in sec. III-B, in the case of microlens arrays with a rectangular arrangement, the smallest periodic step corresponds to one microlens with $J \times J$ pixels. We define the scene texture analogously. We consider that on the microlens array plane there are $P \times P'$ units per block in the hexagonal arrangement case and $D \times D$ units in the rectangular case. This implicitly defines the view index $\mathbf{j} = [j_1\ j_2]$, $j_1 \in [0, \ldots, Q'-1]$, and $j_2 \in [0, \ldots, Q-1]$, and similarly, $\mathbf{t} = [t_1\ t_2]$, $t_1 \in [0, \ldots, P' - 1]$, and $t_2 \in [0, \ldots, P-1]$. With these changes, based on our discussion in sec. III-B, one can see that the LF image generation model in eq. (1) can be implemented using either the convolution-based approach given in eq. (7) or the sparse matrix product-based scheme in eq. (10) even in the hexagonal arrangement case.

D. Motion Blur Model

A relative motion between the camera and the scene during the exposure interval causes the light field image to be motion blurred. In a general 3D scene, the averaging of a scene point at the camera image plane depends on the scene depth value. Consequently, image points corresponding to different depth values should be modeled separately. Let Ω_d denote the support of a layer in the texture domain with depth d, *i.e.*, points $\mathbf{p} \in \Omega_d$ have the same depth. For each depth layer, we assume the scene texture g_d to be a blurry instance of the latent sharp image f_d. The subscript d indicates that the values of $f_d(\mathbf{p})$ and $g_d(\mathbf{p})$ are valid only when $\mathbf{p} \in \Omega_d$. Following the works in [18], [20], [23], and [31], we consider blurring due to in-plane camera rotations and translations.

As derived in [18], the camera rotations and translations result in 2D transformations of the sharp image. Let $\mathbf{g}_d$ and $\mathbf{f}_d$ denote the vectorized forms of g_d and f_d, respectively. The blur degradation can be represented in two equivalent ways: either as a sum of rotated and then filtered instances of the sharp texture, or as a linear combination of shifted and rotated textures

$$\mathbf{g}_d = \sum_{\theta\in\Theta} B_\theta R_\theta \mathbf{f}_d = \sum_{\theta\in\Theta} F_{d_\theta}\beta_\theta \tag{11}$$

θ denotes the angle of rotation within a set of angles Θ, R_θ denotes the 2D warping matrix corresponding to the angle θ, F_{d_θ} denotes a matrix whose columns consist of different shifted instances of the rotated sharp scene texture $R_\theta\mathbf{f}_d$, and B_θ is the matrix equivalent to performing a 2D convolution with the motion blur kernel β_θ. We also have the constraint that $\sum_\theta \|\beta_\theta\|_1 = 1$. The model described in eq. (11) is similar to the slice-based generalized additive convolution model described in [18].

We relate the light field image l_d and the sharp image f_d through a depth-dependent LF PSF and a motion PSF, separately for each layer support $\mathbf{\Omega}_d$. Note that this model does not capture occlusion effects at depth discontinuities. We consider that to be beyond the scope of our paper.

IV. Plenoptic Motion Deblurring

In this section, we describe our methodology to recover a sharp scene texture from a blurry light field image. In our first step, for every depth layer, we recover the blurry scene texture g_d from the light field observation. We use this blurry texture to determine the motion blur PSF through non-uniform blind deconvolution. Finally, we determine the sharp scene texture for every layer through an energy minimization framework.

A. Texture Recovery

Given a light field observation, we initially determine the scene depth map. In our experiments, we use the software from Lytro to obtain the depth map. One could use any other depth estimation scheme as well. We apply the k-means clustering algorithm on the depth map to segment it into N_d clusters. The number of clusters is set by the user although we found $2-3$ layers to suffice in most cases. As an example, for the raw image captured from a Lytro Illum camera shown in Fig. 4 (a), the depth map (top of Fig. 4 (b)) was segmented into two layers $\mathbf{\Omega}_1$ and $\mathbf{\Omega}_2$ as shown in Fig. 4 (b) (bottom).

For each depth layer, we evaluate the entries of the LF PSF using the depth value and camera parameters. We then determine the texture $\mathbf{g}_d$ from the LF observation by restricting the spatial support according to $\mathbf{\Omega}_d$. We minimize the following objective function through gradient descent

$$\arg\min_{g_d} \frac{1}{2}\|H_d\mathbf{g}_d - \mathbf{l}_d\|_2^2 + \lambda\|\mathbf{g}_d\|_{TV} \tag{12}$$

where $\|\mathbf{g}_d\|_{TV} \doteq \sum_\mathbf{p} \|\nabla g_d(\mathbf{p})\|_2$, with $\nabla g_d \doteq [g_{d_x}\ g_{d_y}]^T$, denotes the isotropic total variation of the scene texture [10], [13], and $\lambda > 0$ regulates the amount of total variation.

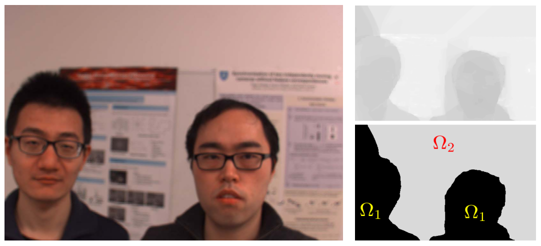

Fig. 4. (a) Raw image (zoom in to see the microlens structure). (b) The depth map shown in the top was segmented into two layers $\mathbf{\Omega}_1$ and $\mathbf{\Omega}_2$ below.

The gradient expression with respect to the data term E_{data} is given by

$$E_{\text{data}} := \frac{1}{2}\|H_d\mathbf{g}_d - \mathbf{l}_d\|_2^2, \quad \frac{\partial E_{\text{data}}}{\partial \mathbf{g}_d} = H_d^T\left(H_d\mathbf{g}_d - \mathbf{l}_d\right) \tag{13}$$

The transpose operation H_d^T can be efficiently implemented in an manner analogous to the forward operation described in eqs. (7) and (10). In our final implementation, we use the matrix-vector product version of eq. (10) because accelerated functions for GPUs that perform sparse matrix-vector product are easily available [4]. The gradient with respect to the TV prior is given by the term $-\nabla\cdot\frac{\nabla g_d}{\|\nabla g_d\|}$. This involves evaluating finite differences and other point-wise arithmetic operations, which can also be implemented on a GPU. We describe the practical details and compare the runtimes of CPU and GPU implementations of the convolution-based and matrix-vector product schemes in sec. V.

For the LF image shown in Fig. 4 (a), we obtain the texture for both layers at one-fourth the full resolution. Both the textures were merged according to the support of each layer $\mathbf{\Omega}_d$. The resultant image is shown in Fig. 5 (b). The all-in-focus image rendered by the Lytro software is shown in Fig. 5 (c). To illustrate the extent of aliasing in the sub-aperture views, we show the central view in Fig. 5 (a) (rendered using the toolbox in [6]). Note that the central view is *highly aliased* as compared to our rendering and Lytro software's rendering.

B. Motion Blur Estimation

We estimate the motion blur PSF by using the blur degradation models from eq. (11) in the following objective function

$$\arg\min_{\mathbf{f}_d,\beta_\theta} \frac{1}{2}\left\|\sum_\theta B_\theta R_\theta\mathbf{f}_d - \mathbf{g}_d\right\|_2^2 + \lambda\|\mathbf{f}_d\|_{TV}$$
$$\text{subject to}\quad \beta_\theta \succcurlyeq 0,\quad \sum_\theta\|\beta_\theta\|_1 = 1. \tag{14}$$

We follow an approach similar to the projected alternating minimization (PAM) algorithm of [32]. At each iteration, we first update the current estimate of the sharp image $\mathbf{f}_d$ through a gradient descent iteration. The gradient with respect to the data term in eq. (14), which we denote again as E_{data}, yields

$$\frac{\partial E_{\text{data}}}{\partial \mathbf{f}_d} = \sum_\theta R_\theta^T B_\theta^T\left(\sum_\theta B_\theta R_\theta\mathbf{f}_d - \mathbf{g}_d\right) \tag{15}$$

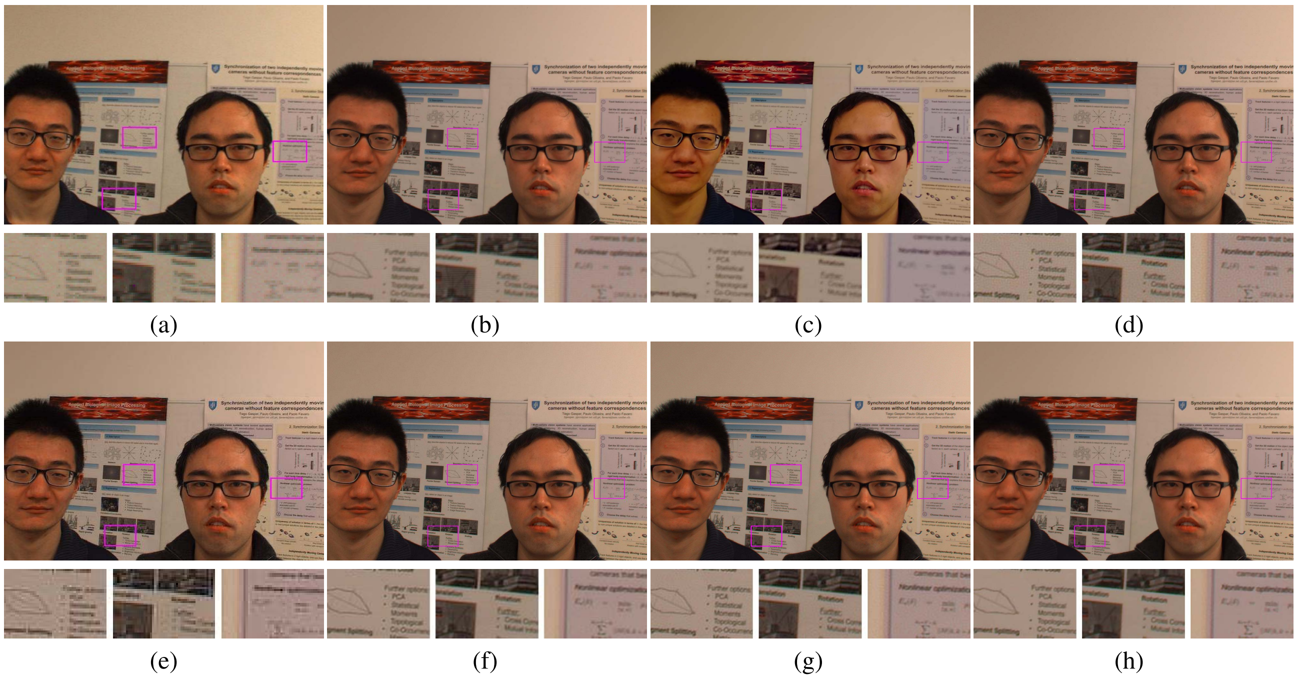

Fig. 5. Texture recovery from a light field observation: (a) Central view. (b) Our GPU implementation (no affine warping nor radial distortion correction in the objective function) with resolution defined by $P = 4$ and $P' = 7$. (c) All-in-focus image rendered by Lytro Desktop software. Reconstructed texture through realistic camera model with TV regularization $\lambda = 2{\cdot}10^{-6}$ and texture resolution defined as (d) $P = 4$ and $P' = 7$, (e) $P = 1$ and $P' = 2$, (f) $P = 2$ and $P' = 4$, and (g) $P = 3$ and $P' = 6$. (h) Recovered texture with $\lambda = 2{\cdot}10^{-4}$ and $P = 4$ and $P' = 7$.

Subsequently, we update the motion blur PSF represented by β_θ for all possible values of θ. The gradient of the data term with respect to each β_θ is given by

$$\frac{\partial E_{\text{data}}}{\partial \beta_\theta} = F_{d\theta}^T\left(\sum_\theta F_{d\theta}\beta_\theta - \mathbf{g}_d\right). \tag{16}$$

After the gradient update, we threshold negative values and apply a constant scaling for all values of β_θ so that $\beta_\theta \succcurlyeq 0$ and $\sum_\theta \|\beta_\theta\|_1 = 1$. For each depth layer d, the blind deconvolution is performed only at pixels corresponding to the layer support $\mathbf{\Omega}_d$. To estimate an accurate motion PSF, we reduce the possibility of including pixels from other layers by shrinking the support of $\mathbf{\Omega}_d$. This eliminates outliers due to depth discontinuities and errors in the estimated depth map.

Note that one could have also estimated motion blur directly from the light field image $\mathbf{l}_d$ with the data term $\left\|H_d \sum_\theta B_\theta R_\theta \mathbf{f}_d - \mathbf{l}_d\right\|_2^2$. However, we observed that in such a scenario, the convergence would be much slower and would also involve significantly more computations.

C. Sharp Texture Recovery

Once the motion blur PSF for each layer is known, one could deblur g_d to estimate the sharp scene texture. However, to improve reconstruction accuracy, we recover the scene texture directly from the light field observation rather than from the blurry texture g_d. According to our model, we can estimate $\mathbf{f}_d$ for each layer by solving

$$\arg\min_{\mathbf{f}_d} \frac{1}{2}\|H_d B \mathbf{f}_d - \mathbf{l}_d\|_2^2 + \lambda\|\mathbf{f}_d\|_{TV}. \tag{17}$$

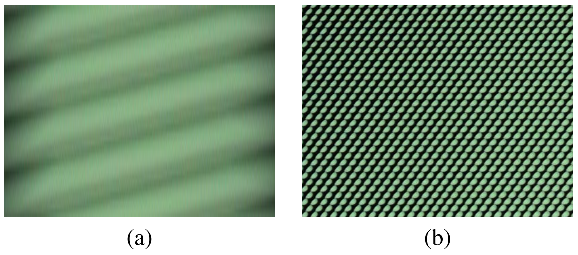

Fig. 6. Cropped white plenoptic image illustrating (a) misalignments, and (b) vignetting.

We define the motion blurring matrix B as $B = \sum_\theta B_\theta R_\theta$ to simplify the notation. Finally, we modify our objective function to incorporate practical imaging aspects such as misalignments and radial distortion.

In real cameras the microlens array is not necessarily aligned with respect to the CCD sensor plane. This effect is apparent in Fig. 6 (a) which shows a crop of size 50×7728 pixels from a raw white image captured with the Lytro Illum camera. In Fig. 6 (a), the gap between adjacent rows of microlenses is tilted instead of being strictly horizontal. Furthermore, in real plenoptic images the distance between adjacent microlens centers in terms of pixels need not be an integer. Hence, we need to apply an affine mapping which would align the raw image to a regular grid model as shown in Fig. 3 with both Q and Q' as integers. We introduce a warping matrix W that aligns the LF image $H_d\mathbf{g}_d$ to the captured raw image. The affine transformation remains fixed for a camera and typically its parameters are stored as metadata [6].

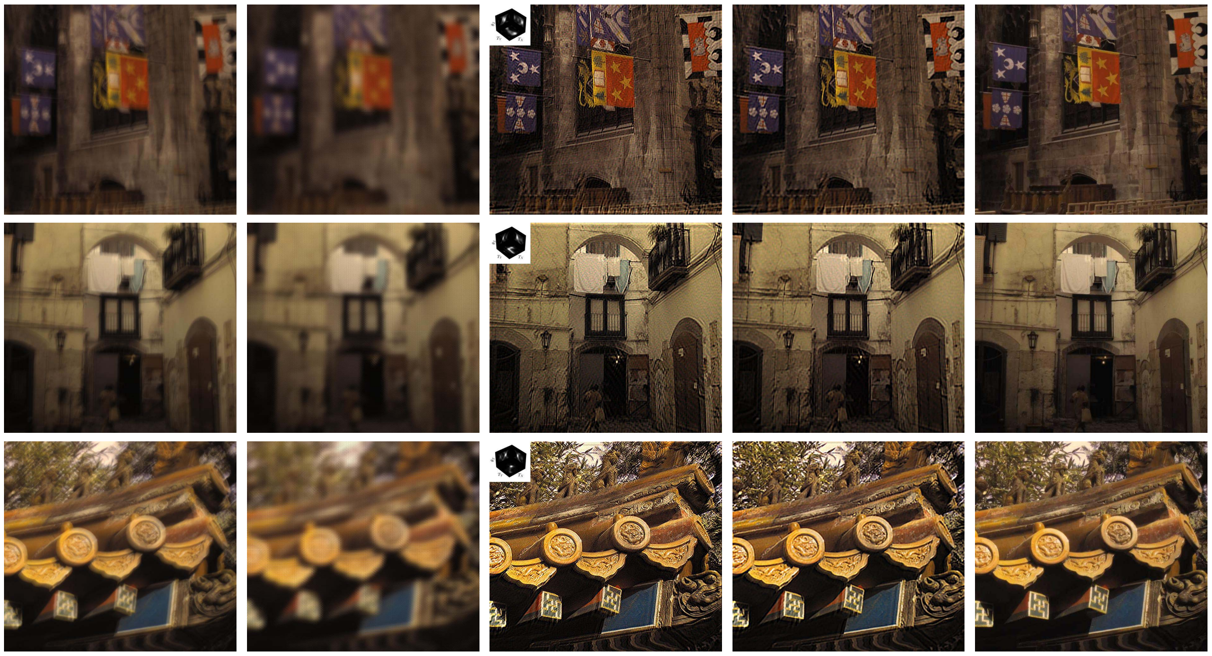

Fig. 7. Example images from the synthetic experiment. First column: motion blurred texture from the dataset of [26], second column: corresponding LF images, third column: `two-step` results along with estimated PSF, fourth column: results of the proposed `unified` deblurring, and fifth column: true texture.

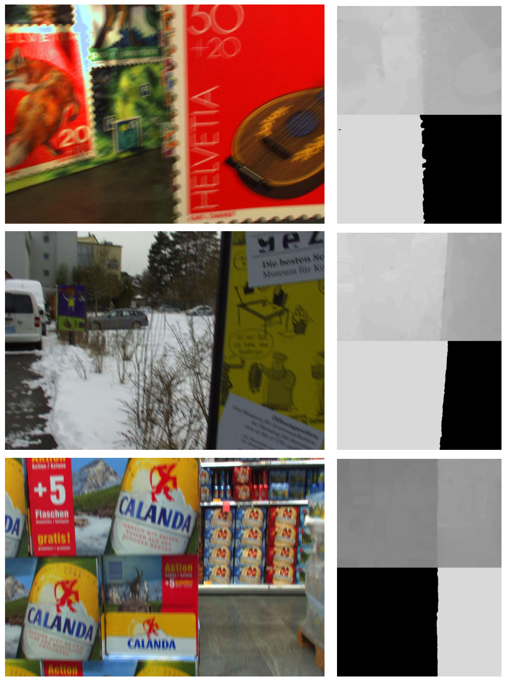

Fig. 8. Depth and texture estimation from real motion blurred light field images: recovered texture (left) and depth map above the layer segmentation (right).

At microlenses close to the corners of the plenoptic image we observe optical vignetting. This is because light rays entering the main lens at extreme angles hit surfaces inside the camera before reaching the sensor. Fig. 6 (b) shows an example of the vignetting effect on LF images. It contains a region cropped at the top left corner of a white plenoptic image. The image pixels under the influence of vignetting do not follow our image formation model. Hence, to avoid using the affected pixels, we use a mask M generated by thresholding a white image. The entries of the mask M will be zero at pixels that are affected by vignetting and one at other locations. Moreover, certain pixels in the sensor array always produce saturated intensities (hot pixels). We detect these pixels by imaging a dark scene and include them in the mask. We also mask the edge pixels of every microlens since they suffer from noise due to demosaicing [17].

Yet one more effect in real cameras is that of radial distortion. In the raw image shown in Fig. 4 (a), we can clearly observe barrel distortion (see, for example, the edges of the posters in the background). To account for this distortion, we define a transformation C, which radially distorts the motion blurred scene texture. For our experiments, we use the calibration technique available in [3] to determine the distortion parameters of our camera. By considering the effect of affine warping, masking and radial distortion, the modified objective function can be written as

$$\arg\min_{\mathbf{f}_d} \frac{1}{2}\|M \odot (W H_d C B \mathbf{f}_d - \mathbf{l}_d)\|_2^2 + \lambda \|\mathbf{f}_d\|_{TV} \qquad (18)$$

where $\odot$ denotes the Hadamard product (point-wise multiplication). The sharp scene texture for each layer is separately obtained via gradient descent. Finally, we merge these textures based on the support of each layer.

In our GPU implementation of the sharp texture recovery in sec. (IV-A), we did not include affine warping and radial distortion correction within the objective function. Affine correction was applied on the light field image to obtain an aligned light field image. This aligned image was used to recover the scene texture. Distortion correction was applied only once after recovering the scene texture. To highlight the importance of accounting for these changes, from the raw

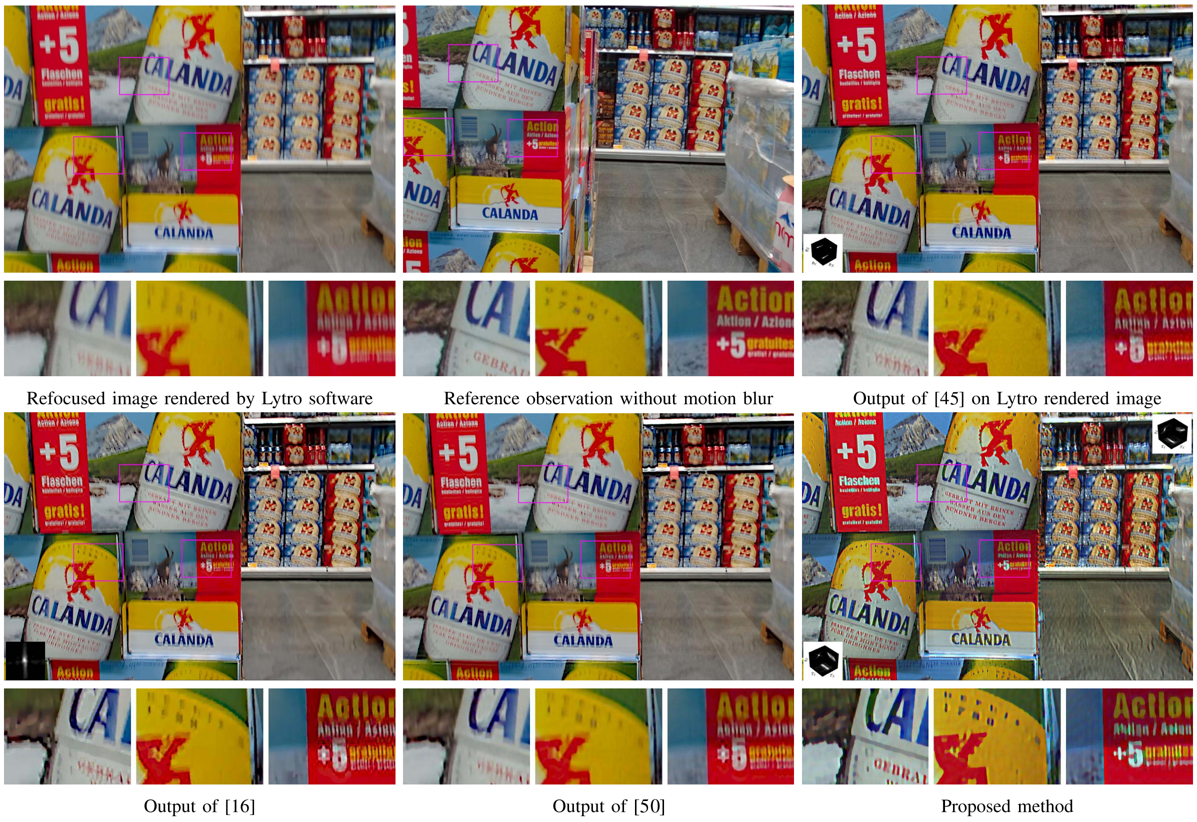
Refocused image rendered by Lytro software
Reference observation without motion blur
Output of [45] on Lytro rendered image
Output of [16]
Output of [50]
Proposed method

Fig. 9. Deblurring results on the image of boxes in a supermarket with the PSFs shown in the insets.

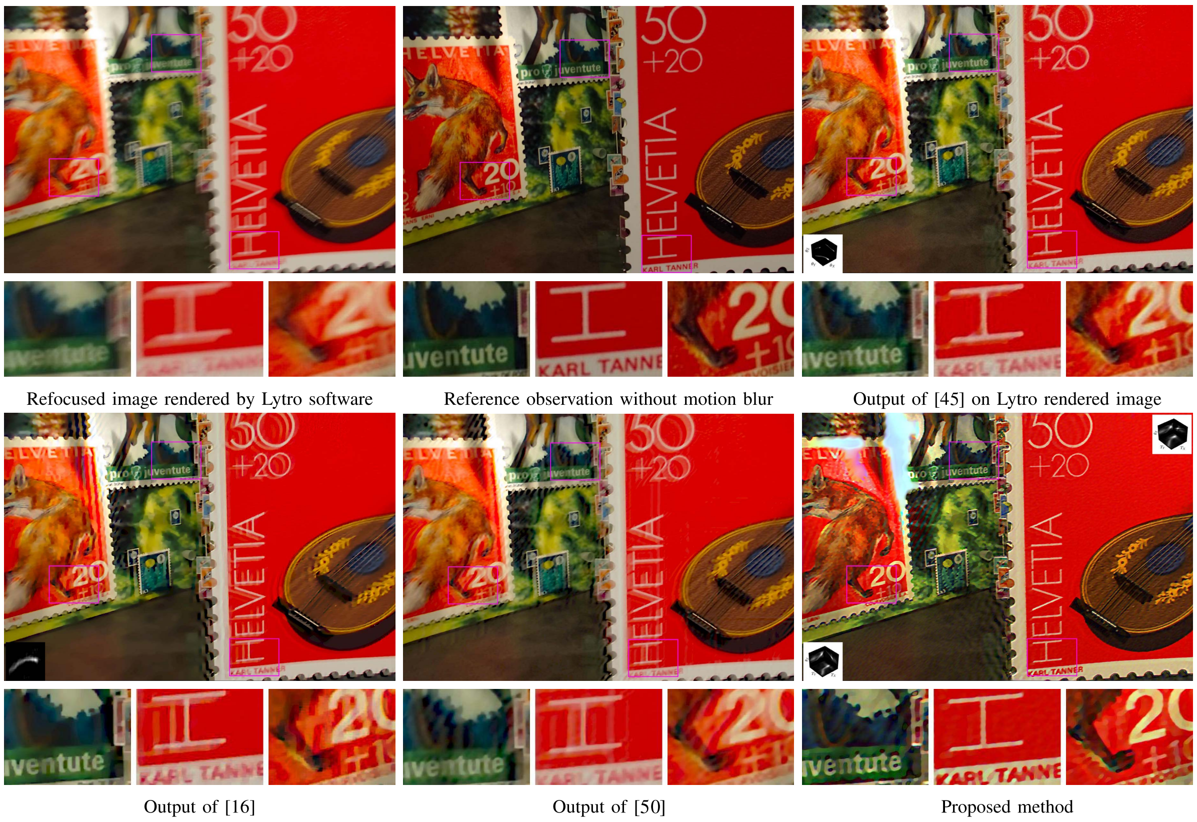
Refocused image rendered by Lytro software
Reference observation without motion blur
Output of [45] on Lytro rendered image
Output of [16]
Output of [50]
Proposed method

Fig. 10. Deblurring results on the image of slanted posters with the PSFs shown in the insets.

image of Fig. 4 (a), we estimated the scene texture according to the objective function in eq. (18) (without considering motion blur). In the resulting image shown in Fig. 5 (d), the text is more clearly readable than the Lytro rendered image in Fig. 5 (c) and our GPU-based result in Fig. 5 (b). Note that both Figs. 5 (b) and (d) were obtained at the same resolution

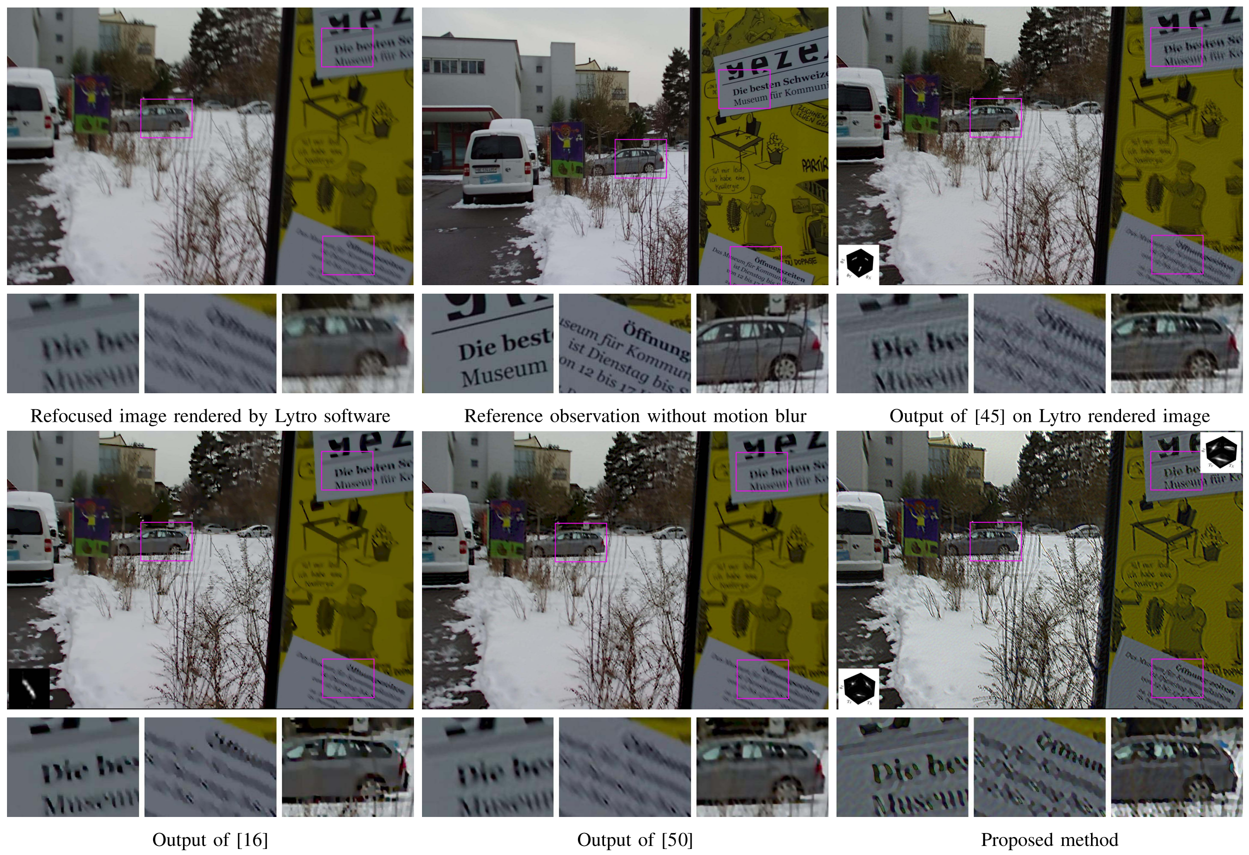
Refocused image rendered by Lytro software
Reference observation without motion blur
Output of [45] on Lytro rendered image
Output of [16]
Output of [50]
Proposed method

Fig. 11. Deblurring results on the image of a street and post with the PSFs shown in the insets.

while Fig. 5 (c) was rendered at a resolution close to our results.

D. Implementation Details

We discuss the implementation details such as super-resolution factor, noise and computational aspects of non-uniform blind deconvolution.

1) Super-Resolution Factor: The number of texture elements per microlens (denoted by D on rectangular grids and by P and P' on hexagonal grids) define the super-resolution *factor*. This factor defines both the resolution of the latent scene texture and that of the LF PSF h_d. In Figs. 5 (e), (f) and (g), we show the reconstructed texture for the resolution factors $P = 1$, $P = 2$, and $P = 3$, respectively. As the resolution factor increases, we see that the reconstructed texture tends to show more and more details. We also estimated texture for the factor $P = 5$ and $P' = 9$. However, we observed no improvement in the resolution quality with respect to the factor $P = 4$ and $P' = 7$ (Fig. 5 (d)). This is due to the inherent limit of the LF camera [11]. Therefore, in all our real experiments we reconstruct texture with resolution parameters $P = 4$ and $P' = 7$.

2) Noise: The raw images from commercially available LF cameras contain significant levels of noise [39], [40], [46]. Consequently, if the regularization is low, our reconstruction can lead to minor noise artifacts as seen in Fig. 5 (d). The effect of noise can be overcome by using a larger regularization value. In Fig. 5 (g), we set the regularization value to be $\lambda = 2 \cdot 10^{-4}$ and do not observe noise artifacts. In all our real experiments, we set $\lambda = 2 \cdot 10^{-4}$ for texture recovery.

3) Non-Uniform Blind Deconvolution: For the successful convergence of blind deconvolution, we follow the commonly used strategy of multiscale implementation. Initially, both the 3D blur kernel and the image are represented in a coarse resolution. After a set of iterations of the PAM algorithm the image and blur kernel are upsampled to a finer resolution.

We select the set of angles Θ to be a uniformly sampled set centered around zero degrees. Following [45], the sample spacing is chosen such that it corresponds to a displacement close to one pixel at the farthest point from the center of rotation. For a typical choice of range $[-2^\circ \;\; 2^\circ]$ and spacing of 0.12°, Θ would contain about 35 elements. In practice, only a small subset of these angles would be contributing to motion blur. Considering the whole set of Θ would be computationally quite intensive as well. Hence, we restrict the number of angles in Θ by defining a support [45]. At the coarsest level, the support includes the entire range of angles in Θ. At the finer scales, the support is determined by ignoring angles with negligible weights. Similar to the strategy in [45], we update the support at each scale after upsampling the 3D blur kernel.

V. Experimental Results

We evaluate our light field blind deconvolution algorithm on several synthetic and real experiments. For synthetic experiments, we use the images of the standard dataset [26] which

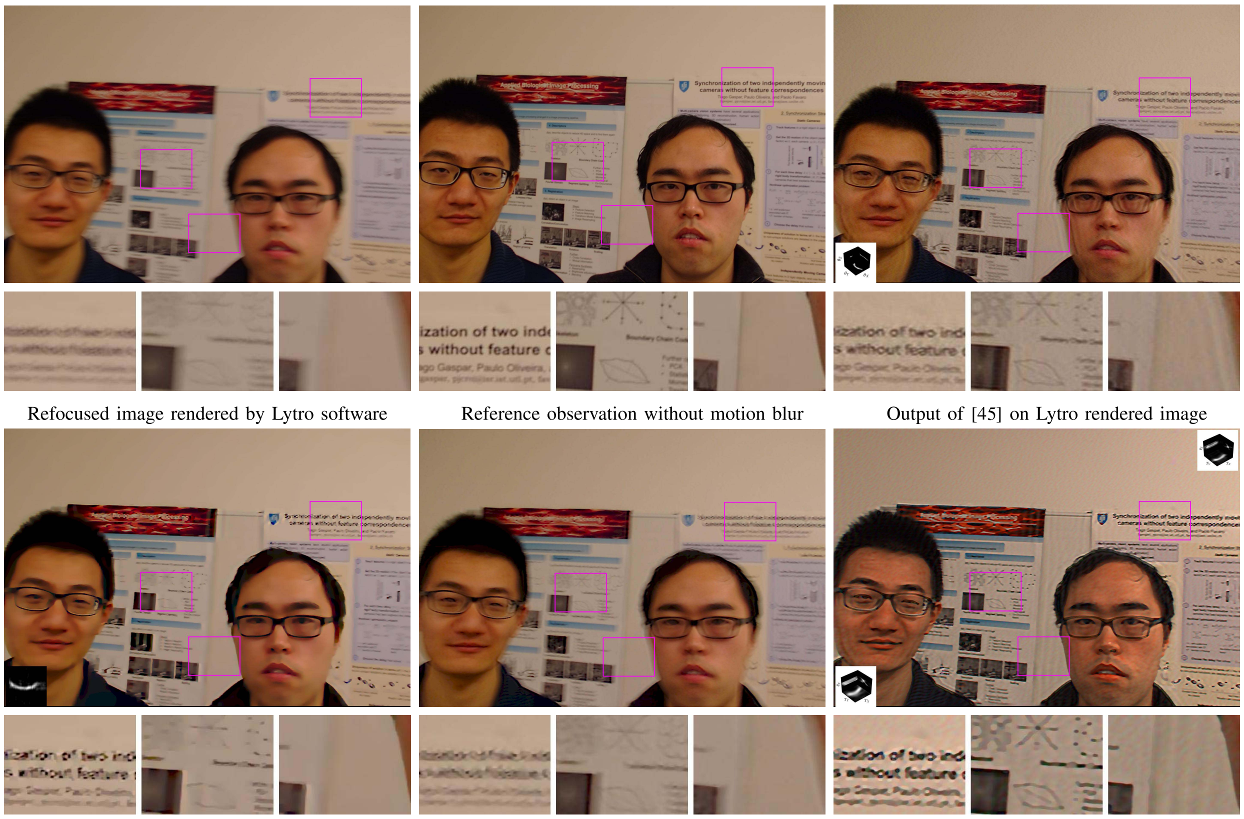
Refocused image rendered by Lytro software
Reference observation without motion blur
Output of [45] on Lytro rendered image
Output of [16]
Output of [50]
Proposed method

Fig. 12. Deblurring results on the image of two persons with the PSFs shown in the insets.

contains non-uniform motion blurred images. From the motion blurred images, we generate the corresponding motion blurred light field data according to the PSF model. We perform real experiments with data captured from a handheld Lytro Illum camera. We use a fixed set of values for the camera parameters in synthetic as well as real experiments. These values are specified in the supplementary material.

A. Synthetic Experiments

The dataset in [26] consists of realistic motion blurred images that were generated from a set of predefined camera trajectories. These camera trajectories were recorded from real camera motions. We consider each image of the dataset as the blurry texture $\mathbf{g}_d$. For every image, we randomly select a depth value ranging from 60 cm to 3 meters, and generate an LF image. For the sake of comparison, we estimate the sharp texture with two different approaches. Both share the same initial processing: First, the motion blurred texture $\mathbf{g}_d$ is recovered from the LF image (see sec. IV-B), and then, through blind deconvolution of $\mathbf{g}_d$, we estimate the motion PSF. With this motion PSF, in one approach we estimate the sharp image $\mathbf{f}_d$ from the observation $\mathbf{g}_d$ by minimizing the data term $\left\|\sum_\theta B_\theta R_\theta \mathbf{f}_d - \mathbf{g}_d\right\|_2^2$ along with TV prior. This approach is referred to as `two-step` method. In the other approach, we use the same motion PSF to estimate the sharp texture directly from the light field observation $\mathbf{l}_d$ as in eq. (17), and we call it the `unified` approach. In both approaches the values of the TV regularization parameters were the same. The regularization weight was set as $\lambda = 2 \cdot 10^{-6}$ for texture recovery, $\lambda = 1.5 \cdot 10^{-3}$ for the alternating minimization scheme, and as $\lambda = 2 \cdot 10^{-5}$ to perform the final deblurring.

TABLE I

COMPARISON BETWEEN DIFFERENT DEBLURRING APPROACHES

	PSNR		SSIM	
	`unified`	`two-step`	`unified`	`two-step`
μ	**27.2**	24.63	**0.86**	0.79
σ	**3.76**	3.96	**0.057**	0.077

In our LF PSF based model we cannot reconstruct the entire texture from an LF image due to boundary effects. There are twelve different blur kernels in the dataset of [26]. We consider only seven of these blur kernels (numbered from 1 to 7 in the dataset) by excluding the ones with high spatial extent. We tested our algorithm on three different textures. Out of the 21 different scenes, three representative examples are shown in Fig. 7. The results on all the 21 scenes are shown in the supplementary material for visual evaluation. We evaluated the performance through Peak Signal-to-Noise Ratio (PSNR) and the Structural Similarity Index (SSIM) [42] metrics using the ground truth images provided by Köhler *et al.* [26]. In Table I, we show the mean and standard deviation of the PSNR and SSIM metrics. The proposed `unified` method outperforms the `two-step` method. Note that, our PSNR values cannot be compared with that of conventional image deblurring since the input to our algorithm is a light field image in which the scene texture undergoes inherent aliasing and defocusing.

B. Real Experiments

We perform real experiments using images of motion blurred scenes captured with a Lytro Illum camera. To read images from the Lytro Illum camera, we use the Light Field Toolbox V0.4 software [6] developed by Dansereau *et al.* [17]. We first normalize the intensity of the raw image by using the white image stored in the LF camera. We capture motion blurred observations of four different real scenes from a handheld camera. In addition to blurred observations, we also capture a separate sharp observation for a direct visual evaluation of our result. In all our experiments, the scene depth ranges between 60cm to a few meters and beyond. In all images we use the same camera settings. Even for obtaining the vignetting mask, we use the stored white image of the camera. After demosaicing the raw white image, the mask M was obtained by considering the pixel locations greater than the threshold value of 0.5 in all the three color channels.

GPU Based Texture Recovery: The Lytro Illum camera has about 432×540 microlenses and the raw image has about 15 pixels per microlens. We define a regular hexagonal grid with $Q = 16$ and $Q' = 28$ (sec. III-C) and apply an affine mapping on the raw LF image to align it to the microlens array structure. The scene texture is defined on a grid with $P = 5$ and $P' = 9$, which corresponds to about one-third of the full sensor resolution. We implement the texture recovery scheme discussed in sec. IV-A on an Nvidia K20 GPU. The execution time of one gradient descent iteration is about 0.25ms per layer for a single color channel. The algorithm easily converges in about 100 iterations even with an arbitrary initial estimate. When we implement the same task on an Intel(R) Xeon(R) core CPU, through the convolution-based scheme of eq. (7), the execution time is about 6.5 seconds per iteration of gradient descent. After texture estimation, we correct for the radial distortion and downsample to a texture grid with $P = 4$, and $P' = 7$. The downsampling operation helps to reduce warping artifacts. The depth maps rendered by Lytro Desktop software, their segmentation, and texture estimation results for three out of four images of our dataset are shown in Fig. 8. The other image we use to test our algorithm is a motion blurred instance of the scene that we show in Fig. 4. For brevity, we avoid repeating a similar depth map and layer segmentation.

We estimate the motion PSF of each layer from the recovered textures (sec. IV-B). Subsequently, the sharp textures are estimated through non-blind deblurring (sec. IV-C). Finally, the layers are merged and color correction is applied by considering Lytro's rendering as the reference. We compare the performance of the proposed algorithm with other conventional image blind deblurring techniques. These techniques are applied to the all-in-focus image generated by the Lytro Desktop software. This comparison is not entirely fair, because our scheme effectively makes use of the depth map while the conventional deblurring schemes are not able to use this information. We compare our method with the schemes in [16], [45], and [50] since these techniques are known to work well even for non-uniform motion blur [26]. For the method in [50] we use the publicly available Matlab executable that also handles non-uniform motion blur [7].

The results are shown in Figs. 9-12. In all the examples, our method consistently renders a sharp estimate. The outputs from other techniques do achieve deblurring either in some specific regions or for a particular scene. However, on the whole, they fail to achieve deblurring due to factors such as depth changes and loss of information in the Lytro rendered image. In some cases, our results produce some artifacts due to inaccurate depth estimates. However, in most of the cases the results are artifact-free even at object boundaries.

VI. CONCLUSIONS

We have presented the first motion deblurring method for plenoptic cameras. Our method extends classical blind deconvolution methods to light field cameras by modeling the interaction of motion blur and plenoptic point spread functions. Moreover, our model is highly parallelizable and memory efficient. By following an alternating minimization procedure, we determine the unknown non-uniform motion blur. We estimate the sharp and super resolved texture by considering practical issues of alignment and radial distortion. Experiments show that our method successfully deals with non-uniform motion blur, and outperforms approaches based on conventional blind deconvolution.

A drawback is that our method is not applicable to scenes with continuous depth variations. Jointly handling motion blur, aliasing and defocusing effects of an LF camera, and depth variations will be left to future work. Our method can be considered as a stepping stone towards this direction.

ACKNOWLEDGEMENTS

The authors thank the reviewers for their constructive suggestions. P. Chandramouli contributed to this paper while he was at the University of Bern.

REFERENCES

[1] *Lytro*. Accessed: Aug. 2014. [Online]. Available: http://http://www.lytro.com/

[2] *Raytrix*. Accessed: Aug. 2014. [Online]. Available: http://www.raytrix.de/

[3] *Camera Calibration Toolbox for MATLAB*. Accessed: Jan. 2017. [Online]. Available: http://www.vision.caltech.edu/bouguetj/calib_doc/

[4] *cuSPARSE*. Accessed: Jan. 2017. [Online]. Available: https://developer.nvidia.com/cusparse

[5] P. Chandramouli, P. Favaro, and D. Perrone. (Aug. 2014). "Motion deblurring for plenoptic images." [Online]. Available: https://arxiv.org/abs/1408.3686

[6] *Light Field Toolbox v0.4*. Accessed: Jan. 2017. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange49683-light-field-toolbox-v0-4

[7] *Unnatural L0 Sparse Representation for Natural Image Deblurring*. Accessed: Jan. 2017. [Online]. Available: http://www.cse.cuhk.edu.hk/leojia/projects/l0deblur/index.html

[8] E. H. Adelson and J. Y. A. Wang, "Single lens stereo with a plenoptic camera," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 99–106, Feb. 1992.

[9] S. D. Babacan, R. Molina, M. N. Do, and A. K. Katsaggelos, "Bayesian blind deconvolution with general sparse image priors," in *Proc. ECCV*, 2012, pp. 341–355.

[10] A. Beck and M. Teboulle, "Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems," *IEEE Trans. Image Process.*, vol. 18, no. 11, pp. 2419–2434, Nov. 2009.

[11] T. E. Bishop and P. Favaro, "The light field camera: Extended depth of field, aliasing and superresolution," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 5, pp. 972–986, May 2012.

[12] M. Broxton *et al.*, "Wave optics theory and 3-D deconvolution for the light field microscope," *Opt. Exp.*, vol. 21, no. 21, pp. 25418–25439, 2013.

[13] A. Chambolle, "An algorithm for total variation minimization and applications," *J. Math. Imag. Vis.*, vol. 20, nos. 1–2, pp. 89–97, 2004.

[14] T. F. Chan and C.-K. Wong, "Total variation blind deconvolution," *IEEE Trans. Image Process.*, vol. 7, no. 3, pp. 370–375, Mar. 1998.

[15] D. Cho, M. Lee, S. Kim, and Y. W. Tai, "Modeling the calibration pipeline of the Lytro camera for high quality light-field image reconstruction," in *Proc. ICCV*, 2013, pp. 3280–3287

[16] S. Cho and S. Lee, "Fast motion deblurring," *ACM Trans. Graph.*, vol. 28, no. 5, 2009. Art. no. 145,

[17] D. G. Dansereau, O. Pizarro, and S. B. Williams, "Decoding, calibration and rectification for lenselet-based plenoptic cameras," in *Proc. CVPR*, 2013, pp. 1027–1034.

[18] H. Deng, D. Ren, D. Zhang, W. Zuo, H. Zhang, and K. Wang, "Efficient non-uniform deblurring based on generalized additive convolution model," *EURASIP J. Adv. Signal Process.*, vol. 22, p. 22, Dec. 2016.

[19] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman, "Removing camera shake from a single photograph," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 787–794, 2006.

[20] A. Gupta, N. Joshi, L. Zitnick, M. Cohen, and B. Curless, "Single image deblurring using motion density functions," in *Proc. ECCV*, 2010, pp. 171–184.

[21] M. Hirsch, C. J. Schuler, S. Harmeling, and B. Schölkopf, "Fast removal of non-uniform camera shake," in *Proc. ICCV*, Nov. 2011, pp. 463–470.

[22] Z. Hu, L. Xu, and M. H. Yang, "Joint depth estimation and camera shake removal from single blurry image," in *Proc. CVPR*, 2014, pp. 2893–2900.

[23] Z. Hu and M. Yang, "Fast non-uniform deblurring using constrained camera pose subspace," in *Proc. BMVC*, 2012, vol. 2. no. 3, p. 4.

[24] H. Ji and K. Wang, "A two-stage approach to blind spatially-varying motion deblurring," in *Proc. CVPR*, Jun. 2012, pp. 73–80.

[25] T. H. Kim, B. Ahn, and K. M. Lee, "Dynamic scene deblurring," in *Proc. ICCV*, 2013, pp. 3160–3167.

[26] R. Köhler, M. Hirsch, B. Mohler, B. Schölkopf, and S. Harmeling, "Recording and playback of camera shake: Benchmarking blind deconvolution with a real-world database," in *Proc. ECCV*, 2012, pp. 27–40.

[27] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman, "Understanding blind deconvolution algorithms," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 12, pp. 2354–2367, Dec. 2011.

[28] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman, "Efficient marginal likelihood optimization in blind deconvolution," in *Proc. CVPR*, 2011, pp. 2657–2664.

[29] A. Lumsdaine and T. Georgiev, "Full resolution lightfield rendering," Adobe Syst., San Jose, CA, USA, Tech. Rep. TR668, 2008.

[30] R. Ng, M. Levoy, M. Brédif, G. Duval, M. Horowitz, and P. Hanrahan, "Light field photography with a hand-held plenoptic camera," Dept. Comput. Sci., Stanford, CA, USA, Tech. Rep. CSTR 2-11, 2005.

[31] C. Paramanand and A. N. Rajagopalan, "Non-uniform motion deblurring for bilayer scenes," in *Proc. CVPR*, 2013, pp. 1115–1122.

[32] D. Perrone and P. Favaro, "Total variation blind deconvolution: The devil is in the details," in *Proc. CVPR*, 2014, pp. 2909–2916.

[33] D. Perrone, R. Diethelm, and P. Favaro, "Blind deconvolution via lower-bounded logarithmic image priors," in *Proc. EMMCVPR*, 2014, pp. 112–125.

[34] C. Perwaß and L. Wietzke, "Single lens 3D-camera with extended depth-of-field," *Proc. SPIE*, vol. 8291, p. 829108, Feb. 2012.

[35] M. Sorel and J. Flusser, "Space-variant restoration of images degraded by camera motion blur," *IEEE Trans. Image Process.*, vol. 17, no. 2, pp. 105–116, Feb. 2008.

[36] P. P. Srinivasan, R. Ng, and R. Ramamoorthi, "Light field blind motion deblurring," in *Proc. CVPR*, 2017.

[37] L. Sun, S. Cho, J. Wang, and J. Hays, "Edge-based blur kernel estimation using patch priors," in *Proc. ICCP*, Apr. 2013, pp. 1–8.

[38] Y.-W. Tai, P. Tan, and M. S. Brown, "Richardson–Lucy deblurring for scenes under a projective motion path," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1603–1618, Aug. 2011.

[39] M. W. Tao, S. Hadap, J. Malik, and R. Ramamoorthi, "Depth from combining defocus and correspondence using light-field cameras," in *Proc. ICCV*, 2013, pp. 673–680.

[40] M. W. Tao, P. P. Srinivasan, J. Malik, S. Rusinkiewicz, and R. Ramamoorthi, "Depth from shading, defocus, and correspondence using light-field angular coherence," in *Proc. CVPR*, 2015, pp. 1940–1948.

[41] M. Tomer and M. Irani, "Blind deblurring using internal patch recurrence," in *Proc. ECCV*, 2014, pp. 783–798.

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

[43] S. Wanner and B. Goldluecke, "Variational light field analysis for disparity estimation and super-resolution," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 3, pp. 606–619, Mar. 2014.

[44] L. Yi Wei, C.-K. Liang, G. Myhre, C. Pitts, and K. Akeley, "Improving light field camera sample design with irregularity and aberration," *ACM Trans. Graph.*, vol. 34, no. 4, 2015 Art. no. 152.

[45] O. Whyte, J. Sivic, A. Zisserman, and J. Ponce, "Non-uniform deblurring for shaken images," in *Proc. CVPR*, 2010, pp. 491–498.

[46] W. Williem and I. K. Park, "Robust light field depth estimation for noisy scene with occlusion," in *Proc. CVPR*, 2016, pp. 4396–4404.

[47] G. Wu *et al.*, "Light field image processing: An overview," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 7, pp. 926–954, Oct. 2017.

[48] L. Xu and J. Jia, "Two-phase kernel estimation for robust motion deblurring," in *Proc. ECCV*, 2010, pp. 157–170.

[49] L. Xu and J. Jia, "Depth-aware motion deblurring," in *Proc. ICCP*, 2012, pp. 1–8.

[50] L. Xu, S. Zheng, and J. Jia, "Unnatural L0 sparse representation for natural image deblurring," in *Proc. CVPR*, 2013, pp. 1107–1114.

Paramanand Chandramouli received the M.S. and Ph.D. degrees in electrical engineering from IIT Madras in 2008 and 2013, respectively. He was a Post-Doctoral Researcher with the University of Bern, Switzerland, from 2013 and 2017. He is currently a Post-Doctoral Researcher with the University of Siegen, Germany. His research interests are in computational photography, inverse problems, and machine learning.

Meiguang Jin received the B.Sc. degree in automation from the Beijing University of Posts and Telecommunications, Beijing, China, in 2011, and the M.Sc. degree in electrical engineering from the Pohang University of Science and Technology, Pohang, South Korea, in 2013. He is currently pursuing the Ph.D. degree with the University of Bern, Switzerland. His research interests include computer vision, machine learning, image processing, inverse problems, and computational photography.

Daniele Perrone received the M.Eng. degree in artificial intelligence from the "Sapienza" Università di Roma in 2009 and the Ph.D. degree in computer science from the University of Bern, Switzerland, in 2015. Since 2015, he has been with Chronocam, Paris, France, where he is currently involved in event-based and neuromorphic vision. His research interests are in event-based vision, neuromorphic and early vision, machine learning, 3D vision, and computational photography.

Paolo Favaro (M'97) received the Laurea degree from the University of Padova, Italy, in 1999, and the M.Sc. and Ph.D. degrees in electrical engineering from Washington University in St. Louis in 2002 and 2003, respectively. He was a Post-Doctoral Researcher with the Computer Science Department, University of California, Los Angeles, and subsequently with Cambridge University, U.K. From 2004 to 2006, he was with Siemens Corporate Research, Princeton, USA, where he was involved in medical imaging. From 2006 to 2011, he was a Lecturer and then a Reader with Heriot-Watt University and an Honorary Fellow with the University of Edinburgh, U.K. He is currently a Full Professor with the University of Bern, Switzerland, where he leads the Computer Vision Group. His research interests are in computer vision, computational photography, machine learning, and inverse problems.