Computers & Graphics 37 (2013) 1039–1050

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Special Section on Advanced Displays

Non-uniform image deblurring using an optical computing system

Tao Yue [a], Jinli Suo [a], Qionghai Dai [a,b,*]

[a] Department of Automation, Tsinghua University, China
[b] Tsinghua National Laboratory for Information Science and Technology (TNList), China

ARTICLE INFO

Article history:
Received 5 September 2013
Received in revised form
13 October 2013
Accepted 14 October 2013
Available online 31 October 2013

Keywords:
Projector–camera system
Non-uniform deblurring
Acceleration
Optical computing

ABSTRACT

Removing non-uniform blur caused by camera shaking is troublesome because of its high computational cost. We analyze the efficiency bottlenecks of a non-uniform deblurring algorithm and propose an efficient optical computation deblurring framework that implements the time-consuming and repeatedly required modules, i.e., non-uniform convolution and perspective warping, by light transportation. Specifically, the non-uniform convolution and perspective warping are optically computed by a hybrid system that is composed of an off-the-shelf projector and a camera mounted on a programmable motion platform. Benefitting from the high speed and parallelism of optical computation, our system has the potential to accelerate existing non-uniform motion deblurring algorithms significantly. To validate the effectiveness of the proposed approach, we also develop a prototype system that is incorporated into an iterative deblurring framework to effectively address the image blur of planar scenes that is caused by 3D camera rotation around the *x*-, *y*- and *z*-axes. The results show that the proposed approach has a high efficiency while obtaining a promising accuracy and has a high generalizability to more complex camera motions.

1. Introduction

A key issue in computational photography, image blur caused by camera shake, is a commonly occurring degradation; restoring latent sharp images from such blurry inputs is still a challenging problem. In real cases, camera shake blur can intensively vary in the spatial domain. However, because of the high complexity and computational cost of non-uniform blurring models, studies on camera shake removal formulate camera shake blur with uniform convolution and propose many uniform deblurring methods [1–9].

In recent years, several promising non-uniform blind image deblurring algorithms have been proposed, and researchers have refocused their attention on non-uniform blur models, which suffer severely from having a high computational cost. In spite of the recent progress in non-uniform deblurring, the low computational efficiency still limits the application of existing algorithms.

Most existing deblurring algorithms can be classified into two types—MAP$_{L,K}$ and variational inference-based approaches. The former iteratively calculate a blurry image from a current estimation of a sharp image and camera motion and then correct the sharp image and camera motion according to the residual between the calculated blurry image and the captured version. In contrast, the latter approach is much more complex and will be detailed in Section 3.2 later. However, these two types of approaches share some repeatedly called modules that can be calculated exactly or approximated well by non-uniform convolution, which is an approach that is quite time-consuming because it must be computed in a pixel-wise manner and performed many times during the deblurring process. Unfortunately, there is currently no accurate acceleration algorithm for this operation; thus, this dilemma motivates us to explore acceleration approaches by resorting to assistance from hardware.

Optical computing for calculations of acceleration is an area that has been well studied in optics in recent decades. Many commonly used mathematical operations can be accelerated by delicately designed optical systems [10,11]. However, there is no existing optical computing system for non-uniform convolution, which is the module that needs acceleration the most in the case of non-uniform deconvolution. Intuitively, time-consuming pixel-wise convolution, which is conducted in each iteration of non-uniform deblurring algorithms, corresponds to a spatially varying image blurring process. This approach

* Corresponding author at: Department of Automation, Tsinghua University, China. Tel.: +86 10 62788613 828.
E-mail addresses: yuetao.thu@gmail.com (T. Yue), jlsuo@tsinghua.edu.cn (J. Suo), qhdai@tsinghua.edu.cn (Q. Dai).

0097-8493/$ - see front matter © 2013 Published by Elsevier Ltd.
http://dx.doi.org/10.1016/j.cag.2013.10.011

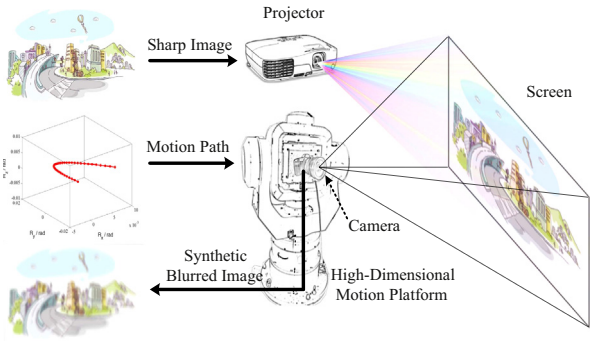

Fig. 1. The diagram of our optical computing system for fast non-uniform convolution. The system is composed of a projector, a high-dimensional motion platform and a camera that is mounted on the platform.

motivates us to build a new imaging system to physically simulate the imaging process (as shown in Fig. 1) that corresponds to the convolution exactly and, thus, reduce the computational cost. In other words, we simulate the non-uniform convolution directly instead of computing it pixel by pixel or approximating by patch-based methods. Specifically, we project the sharp image onto a planar screen as a synthetic scene and simulate the blurring process by imaging the screen, using a shaken camera driven by a programmable motion platform. Based on this system, we build an optical computing framework by incorporating the above simulation into widely used deblurring iterations for non-uniform camera shake removal under a constant depth assumption, which is widely used in the existing non-uniform deblurring algorithms [12–19], except for [20,21].

This paper is an extension of Yue et al.'s [22] work, which is the first attempt to address the efficiency issue in non-uniform deblurring by optical computing. However, the approach in the conference version is limited in two facets: first, the approach here is applicable only for non-blind deblurring, while blind deblurring is more useful in real applications; second, Yue et al. [22] derived only the optical acceleration of a $MAP_{L,K}$ framework, but did not discuss the variational-based algorithm, which is another important and state-of-the-art deblurring framework. This paper mainly extends the following aspects:

(1) completing the analysis of the most time-consuming operations in variational-based methods and formulating those operations with a non-uniform convolution;
(2) applying our optical computing framework for blind image deblurring and validating its effectiveness experimentally;
(3) giving a motion trajectory planning method from a discrete 3D kernel, which bridges the optical computing system and blind deblurring framework.

2. Related studies

Non-uniform deblurring: Recently, non-uniform blurring has attracted the interest of many researchers, and many promising algorithms have been proposed. Most of the existing methods can be roughly categorized as pixel-wise or patch-wise.

(1) Pixel-wise methods: The camera shake is arbitrary, and a 6-D motion model is a reasonable assumption. Blind estimation of a 6-D blur kernel is often computationally costly; as a result, researchers have introduced some external assistance in the kernel estimation. Tai et al. [15,16] extend the Richardson–Lucy (RL) algorithm [15] for non-uniform deblurring, and they estimate the 6-D projective motion from the user interaction. In contrast, Joshi et al. [17] measured the 6-D motion using inertial sensors and computed the latent sharp image with a sparse image prior. There are also researchers who reduce the estimation cost by assuming that there is a lower camera motion dimension. Whyte et al. [12] simplified 6-D camera motion to 3-D rotations and solved the problem by extending the uniform deblurring algorithm by the approach of Fergus et al. [2]. Gupta et al. [14] proposed the 2-D translation and 1-D rotation blur model to approximate the camera motion, and solve the problem with a RANSAC-based framework. However, non-uniformity requires the computation of the convolution in a pixel-wise manner and pursues the optimum blur kernel by exhaustive searching; thus, the above approaches all suffer from having a high computational cost, which inspires our studies on efficient non-uniform camera shake removal.

(2) Patch-based methods: To compute the time-consuming non-uniform convolution quickly, an Efficient Filter Flow (EFF) based [18] method is proposed for acceleration. Recently, a tree-structured patch-based method [21] is proposed to solve the depth-dependent non-uniform blur kernels hierarchically. Although these patch-based methods can greatly reduce the running time, this approach can lead to some artifacts in cases that have highly varying blur because the assumption of slow variance on the blur kernels is violated in such cases. Our approach is largely different from and advantageous over a patch-based approximation because the acceleration is not obtained at the expense of accuracy.

Optical computation: Optical computation attempts to perform computation with photon movements using lenses, modulators, detectors and any other optical elements. Researchers have made use of the high speed and parallelism of light transport in recent decades and have made great progress; we refer the readers to [10,11] for a survey of this field.

Earlier works [23–28] on optical computing basically focused on general-purpose computing, such as matrix multiplication, Fourier transformation, and matrix decomposition. However, with the rapid development of digital computing, the advantages of optical computing with respect to speed have been greatly weakened. However, it is still promising to design specific optical computing systems for concrete tasks that require intensive non-general calculations without accelerating their implementations. For example, O'Toole and Kutulakos [29] uses a projector–camera system to perform light transport computing [30], and Yu et al. [31] apply optical computing for pattern recognition. In addition, some optical approaches, such as coded apertures, are applied to recover the image and scene information [16,32–34].

Non-uniform deblurring algorithms are highly time-consuming with some non-general operations. They bear the property of parallelism but cannot be implemented directly with current optical computing systems. Naturally, some elegant designs are necessary for such task-specific computations; here, we design and implement the system and then validate it with a series of experiments.

3. Computationally intensive calculations in non-uniform deblurring

Targeting the development of an optical computing system for fast non-uniform deblurring, we first analyze here the

common time-consuming calculations in existing algorithms. Most of the existing image deblurring algorithms estimate the blur kernel and latent sharp image separately. In terms of mathematical derivations, the operations in latent image recovery and kernel estimation are quite similar. The only difference is that gradient information of the latent image is involved for estimating the kernel, while low-frequency information is required at an image restoration stage. Therefore, this paper focuses on discussing the kernel estimation stage, and the image restoration algorithm can be easily derived by replacing the image gradients in the kernel estimation stage with image intensities.

Many non-uniform deblurring algorithms are based on a specific motion blur model, e.g., a 3-D rotational model [12], a 2-D translation and a 1-D rotational model [14], and a 6-D perspective projection model [15–17], among others. According to the discussions in [35] reported by Levin et al., for all of these blur models, two types of kernel estimation methods can be applied: $\mathrm{MAP}_{L,K}$ and variational based. Both of the estimation methods can achieve good performance, while the computing processes are totally different. For the former type of algorithm, an additional regularizer, e.g., spatially varying terms [36,3], optimization solvers [3,37] or a delicate prediction procedure [4], are required. For the latter case, some approximations are required, such as those in [2,12]. We will show that, in spite of the large difference in their main steps, both of the methods can be accelerated with our system by replacing or approximating the intensive calculations with blurring and warping operations.

3.1. $\mathrm{MAP}_{L,K}$ type of method

Almost all of the existing $\mathrm{MAP}_{L,K}$–like deblurring algorithms optimize the latent sharp image $\mathbf{L}$ and blur kernel (or motion path) $\mathbf{K}$ by minimizing the following energy:

$$E(\mathbf{L},\mathbf{K}) = \|\hat{\mathbf{B}}(\mathbf{L},\mathbf{K}) - \mathbf{B}\|^2 + \lambda_1 J_1(\mathbf{L}) + \lambda_2 J_2(\mathbf{K}), \tag{1}$$

where $\hat{\mathbf{B}}$ is the blurring function for generating a blurry image from a latent sharp image $\mathbf{L}$ and motion $\mathbf{K}$. In our scenario, $\mathbf{K}$ is a 3D vector that denotes 3D rotation around the x-, y- and z-axes, respectively, and $\mathbf{B}$ is the observed blurry image. To simplify the mathematical representation, $\mathbf{L}$, $\mathbf{K}$ and $\mathbf{B}$ are denoted by column vectors in the following. $J_1(\cdot)$ and $J_2(\cdot)$ regularize the estimated sharp image and kernels to reduce the ill-posedness, with λ_1 and λ_2 being weighting coefficients.

Depending on the concrete form of $J_1(\cdot)$ and $J_2(\cdot)$ in Eq. (1), the objective functions can be convex or non-convex. For the former case, traditional algorithms such as gradient-based methods can be applied, for example, steepest-descent, Conjugate Gradient (CG) or a closed-form solution (in the frequency domain and inapplicable for non-uniform deblurring). For a non-convex case, with terms that favor sparse high-frequency components in recovered images or that force the camera motion trajectory to be sparse, the optimization is much more complicated. For non-convex optimization, the IRLS [38] algorithm convexifies the objective into a summation of weighted least square terms, which can be solved by CG and is widely used for deblurring due to its effectiveness. In summary, almost all of the existing non-uniform deblurring algorithms, either convex or non-convex, take the gradient of the energy E in each iteration. However, the non-uniformity makes this operation computationally intensive and becomes an efficiency bottleneck in most of the non-uniform deblurring algorithms.

For non-uniform deblurring, the prior terms $J_1(\cdot)$ and $J_2(\cdot)$ in Eq. (1) are still uniform, which means that their gradient calculations can be accelerated easily. In contrast, the data term $\|\hat{\mathbf{B}}(\mathbf{L},\mathbf{K}) - \mathbf{B}\|^2$ is non-uniform, and its derivative manipulations $\partial E/\partial \mathbf{L}$ and $\partial E/\partial \mathbf{K}$ must be calculated pixel by pixel, which is a demanding calculation.

$\partial E/\partial \mathbf{L}$ for optimizing a latent image $\mathbf{L}$: For analytical convenience, we rewrite Eq. (1) as

$$\begin{aligned} E(\mathbf{L},\mathbf{K}) &= \|\mathbf{H}\mathbf{L} - \mathbf{B}\|^2 + \lambda_1 J_1(\mathbf{L}) + \lambda_2 J_2(\mathbf{K}) \\ \mathbf{H} &= \sum_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} K_{\boldsymbol{\theta}} \cdot \mathbf{W}_{\boldsymbol{\theta}}, \\ \frac{\partial E}{\partial \mathbf{L}} &= 2\mathbf{H}^T\mathbf{H}\mathbf{L} - 2\mathbf{H}^T\mathbf{B} + \lambda_1 \frac{\partial J_1}{\partial \mathbf{L}} \end{aligned} \tag{2}$$

where $\mathbf{H}$ is the blurring matrix that is determined by the camera motion and the blur model. Specifically, $\boldsymbol{\Theta}$ denotes the discretized high-dimensional motion parameter space, and each $\boldsymbol{\theta}$ corresponds to a camera pose (described by an N-tuple for a N-degree-of-freedom blur model) during exposure; $K_{\boldsymbol{\theta}}$ denotes the weight that reflects the elapsed time that the camera spends at pose θ, i.e., $K_{\boldsymbol{\theta}}$ is exactly the elements in the motion vector $\mathbf{K}$ with the 3D pose index $\boldsymbol{\theta}$, and $\mathbf{W}_{\boldsymbol{\theta}}$ is the warping matrix that maps the reference view to the view that corresponds to the camera pose $\boldsymbol{\theta}$.

For computing $\partial E/\partial \mathbf{L}$, $\mathbf{H}^T\mathbf{H}\mathbf{L}$ and $\mathbf{H}^T\mathbf{B}$ are the key manipulations. Here, $\mathbf{H}$ is the sparse blurring matrix for non-uniform convolution. $\mathbf{H}\mathbf{L}$ can be calculated by an exact simulation of the blurring process, i.e., calculating all of the views along the motion path and integrating them with the corresponding weights. In a similar way, $\mathbf{H}^T(\mathbf{H}\mathbf{L})$ and $\mathbf{H}^T\mathbf{B}$ should be computed by conducting the blurring procedure with an inverse motion path (see [15] for details).

Notably, the convolution of residuals that contain negative values must be computed in many deblurring methods. However, the negative values cannot be modeled by a physical imaging process. To address this problem, we normalize the input to make the dynamic range from 0 to 255 and map them back after optical computing.

$\partial E/\partial \mathbf{K}$ for optimizing the non-uniform blur kernel $\mathbf{K}$: We reform Eq. (1) with a matrix representation as

$$\begin{aligned} E(\mathbf{L},\mathbf{K}) &= \|\mathbf{A}\mathbf{K} - \mathbf{B}\|^2 + \lambda_1 J_1(\mathbf{L}) + \lambda_2 J_2(\mathbf{K}) \\ \mathbf{A} &= [\mathbf{W}_{\boldsymbol{\theta}_1}\mathbf{L}\ \ \mathbf{W}_{\boldsymbol{\theta}_2}\mathbf{L} \cdots \mathbf{W}_{\boldsymbol{\theta}_n}\mathbf{L}] \\ \frac{\partial E}{\partial \mathbf{K}} &= 2\mathbf{A}^T(\mathbf{A}\mathbf{K} - \mathbf{B}) + \lambda_2 \frac{\partial J_2}{\partial \mathbf{K}} \end{aligned} \tag{3}$$

here $\theta_1 \cdots \theta_n$ is the camera pose traversing the motion parameter space, and $\mathbf{W}_{\theta_i}$ is the warping matrix that corresponds to pose θ_i. To minimize Eq. (3), the matrix $\mathbf{A}$ must be calculated many times. Warping images are quite slow even with GPU acceleration. Therefore, to derive the non-uniform motion blur kernel, a series of highly time-consuming image warping operations is performed.

The above analysis tells us that *non-uniform convolution and warping image sequences are common calculations and efficiency bottlenecks, respectively, in computing $\partial E/\partial \mathbf{L}$ and $\partial E/\partial \mathbf{K}$*. Fortunately, by a delicately designed optical computing system, both can be accelerated by a simple snapshot. Therefore, we replace these two operations with optical computing for acceleration, as blocked in Algorithm 1.

Algorithm 1. A simple non-uniform deblurring algorithm incorporating our optical computing system.

Input: Blurred image $\mathbf{B}$
Output: latent sharp image $\mathbf{L}$ and camera motion $\mathbf{K}$

Initialization: $\mathbf{L}^0=\mathbf{B}$ and $t=0$;
Repeat
  Predict sharp image by bilateral and shock filters
  Repeat following step **until** converge :
    Project $\mathbf{L}^t$, move the camera along $\mathbf{K}^t$ to predict blurred image $\hat{\mathbf{B}}^t=\hat{\mathbf{B}}(\mathbf{L}^t,\mathbf{K})=\mathbf{A}^t\mathbf{K}^t$;
    Compute residual : $\mathbf{e}^t=\mathbf{B}-\hat{\mathbf{B}}^t$;
    Project $\mathbf{e}_t$, move camera traversing motion parameter space, computer $\frac{\partial E'}{\partial K_\theta}=(W_\theta\mathbf{L}^t)^T\mathbf{e}^t$ for each θ;
    Update motion kernel : $\mathbf{K}^{t+1}=\mathbf{K}^t+\partial E'/\partial\mathbf{K}^t$;
    Set $K_\theta^{t+1}=0$ if $K_\theta^{t+1}<0$, and normalize $K_\theta^{t+1}=\frac{K_\theta^{t+1}}{\sum_\theta K_\theta^{t+1}}$.
  Repeat following step **until** converge :
    Project $\mathbf{L}^t$, move the camera along $\mathbf{K}^t$ to predict blurred image $\hat{\mathbf{B}}^t=\hat{\mathbf{B}}(\mathbf{L}^t,\mathbf{K})$;
    Compute residual : $\mathbf{e}^t=\mathbf{B}-\hat{\mathbf{B}}^t$;
    Project $\mathbf{e}^t$, move the camera along $\mathbf{K}^{t-1}$ to capture correction image $\frac{\partial E'}{\partial\mathbf{L}}=\hat{\mathbf{B}}(\mathbf{e}^t,\mathbf{K}^{t-1})$;
    Update latent sharp image : $\mathbf{L}^{t+t}=\mathbf{L}^t+\partial E'/\partial\mathbf{L}^t$;
  $t=t+1$;
Until $\|\hat{\mathbf{B}}^t-\mathbf{B}\|^2<thr$.

3.2. Variational-based methods

Imposing a sparse prior on the latent sharp image, blind deblurring can be performed by a variational Bayesian method [2,12]. In contrast to the $\mathrm{MAP}_{L,K}$ type of methods, the variational-based methods include many specific operations that cannot be computed by blurring and warping operations simply. However, detailed investigation tells us that almost all of the time-consuming operations in variational deblurring algorithms can be transformed into or approximated by blurring and warping operations and consequently accelerated by our system.

In this paper, we use the typical non-uniform variational deblurring algorithm that is proposed in [13] as a representative example. To make this paper self-contained, we briefly introduce the algorithm and refer the readers to [13] for the details. As a variational Bayesian-based method, the joint distribution $p(\mathbf{L},\mathbf{K})$ is approximated by the production of a series of Gaussian distributions $q(\beta_\sigma)\sim N(0,1/\beta_\sigma)$, $q(K_\theta)\sim N(\langle K_\theta\rangle,v_\theta^K)$ and $q(L_j)\sim N(\langle L_j\rangle,v_j^L)$ as

$$p(\mathbf{L},\mathbf{K})\approx q(\beta_\sigma,\mathbf{L},\mathbf{K})=q(\beta_\sigma)\prod_j q(L_j)\prod_\theta q(K_\theta), \tag{4}$$

where $q(\beta_\sigma)$ is the zero mean noise, which follows a Gaussian distribution, and the deviation is $1/\beta_\sigma$; $q(K_\theta)$ and $q(L_j)$ are the factorized distributions of the blur kernel and latent image, respectively; and θ and j are the indices of the kernel and image, respectively. In Eq. (4), all of the unknowns, including each element (voxels and pixels) of the blur kernel and latent image and noise, are parameterized with a probabilistic distribution with two parameters, i.e., the expectation and variance (denoted with $\langle\cdot\rangle$ and $v^{(\cdot)}$, respectively). According to the variational inference theory [39], the approximated joint distribution can be estimated by minimizing the Kullback–Leibler divergence between the posterior joint distribution and the approximated distribution:

$$C_{KL}=\int q(\beta_\sigma,\mathbf{L},\mathbf{K})\left[\ln\frac{q(\beta_\sigma,\mathbf{L},\mathbf{K})}{p(\mathbf{L},\mathbf{K})}-\ln p(\mathbf{B}|\mathbf{L},\mathbf{K})\right]d\beta_\sigma\,d\mathbf{L}\,d\mathbf{K}. \tag{5}$$

This function can be minimized by computing the optimal forms of $p(L_j)$, $p(K_\theta)$ and $p(\beta_\sigma)$ with calculus of variation and optimizing the parameters of the approximated distribution iteratively. The updated rules [13, Appendix A, Eq. 35–43] are

$$v_\theta^{\mathbf{K}}=\langle\mathbf{K}_\theta^2\rangle-\langle\mathbf{K}_\theta\rangle^2 \tag{6}$$

$$v_j^{\mathbf{L}}=\langle\mathbf{L}_j^2\rangle-\langle\mathbf{L}_j\rangle^2 \tag{7}$$

$$\langle a_{ij}\rangle=\sum_K W_{ij\theta}\langle K_\theta\rangle \tag{8}$$

$$\langle b_{i\theta}\rangle=\sum_j W_{ij\theta}\langle L_j\rangle \tag{9}$$

$$\langle\hat{B}_i\rangle=\sum_\theta\sum_j W_{ij\theta}\langle L_j\rangle\langle K_\theta\rangle \tag{10}$$

$$K_\theta^{(2)}=\langle\beta_\sigma\rangle\sum_{i,j}W_{ij\theta}^2v_j^{\mathbf{L}}+\langle\beta_\sigma\rangle\sum_i\langle b_{i\theta}\rangle^2 \tag{11}$$

$$K_\theta^{(1)}K_\theta^{(2)}=\langle\beta_\sigma\rangle\sum_i\langle b_{i\theta}\rangle(B_i-\hat{B}_i)-\langle\beta_\sigma\rangle\sum_{i,j}W_{ij\theta}\langle a_{ij}\rangle v_j^{\mathbf{L}}+\langle K_\theta\rangle K_\theta^{(2)} \tag{12}$$

$$L_j^{(2)}=\langle\beta_\sigma\rangle\sum_{i,\theta}W_{ij\theta}^2v_\theta^{\mathbf{K}}+\langle\beta_\sigma\rangle\sum_i\langle a_{ij}\rangle^2 \tag{13}$$

$$L_j^{(1)}L_j^{(2)}=\langle\beta_\sigma\rangle\sum_i\langle a_{ij}\rangle(B_i-\hat{B}_i)-\langle\beta_\sigma\rangle\sum_{i,\theta}W_{ij\theta}\langle b_{i\theta}\rangle v_\theta^{\mathbf{K}}+\langle L_j\rangle L_j^{(2)} \tag{14}$$

here we denote the blurry image and latent image as $\mathbf{B}$ and $\mathbf{L}$. To represent the predicted blurry image that is computed from a current estimation of the blur kernel and latent image, the upper hat $\hat{B}$ is used. According to the perspective geometry and blur model, $W_{ij\theta}$ represents the warping coefficient from pixel j in a latent image to pixel i under a camera pose θ. In addition, to summarize the operations, we use $[\cdot]$ to denote a variable that has the same form. For example, $[L_j]$ denotes that some variables, e.g., v_j^L and ΔL_j, have the same size as L and can be indexed with j.

There are three main time-consuming operations in [13], and we derive their calculations as follows.

First operation: The first term of Eqs. (11) and (13), respectively, i.e., $\sum_{i,j}W_{ij\theta}^2[L_j]$ and $\sum_{i,\theta}W_{ij\theta}^2[K_\theta]$. In fact, the warping coefficient $W_{ij\theta}$ is an impulse-like function and has only a large value in specific combinations of i, j and θ. Considering the rule of intensity conservation, $W_{ij\theta}$ is the discrete version of a unit impulse function, and its square can be approximated by itself. Then, the two operations become $\sum_{i,j}W_{ij\theta}[L_j]$ and $\sum_{i,\theta}W_{ij\theta}[K_\theta]$. The former operation can be computed by summing up the intensities of the warped latent image under the camera pose θ, i.e., $\sum\mathbf{W}^\theta[L]$, and the latter operation can be computed by summing up the intensities of the image blurred from a white image (denote as $\mathbf{I}$) under kernel $[\mathbf{K}]$, i.e., $\sum\hat{B}(\mathbf{I},[\mathbf{K}])$.

Second operation: Second term of Eqs. (11) and (13), respectively, i.e., $\sum_i(\sum_j W_{ij\theta}[L_j])^2$ and $\sum_i(\sum_\theta W_{ij\theta}[K_\theta])^2$. The former term can be computed directly by summing up the square of the warped image under the camera pose θ. The optical computation of the latter term is not straightforward and requires further analysis. In fact, the components under the square $(\sum_\theta W_{ij\theta}[K_\theta])$ are the Point Spread Function (PSF) of each pixel j, and the whole operation is exactly the $L2$ norm of the PSF of each pixel j. Considering that the PSFs vary smoothly across the image lattice, we can optically compute the intensities at a grid and compute the intensities of the remaining pixels by spline interpolation. In the

implementation, we project a point grid pattern (with several 1-value pixels and a 0-value for the remainder) onto the screen and compute the PSFs of the 1-value pixels with our optical computing system.

Third operation: The second term of Eqs. (12) and (14) respectively, i.e., $\sum_{i,j}W_{ij\theta}\sum_{\theta}W_{ij\theta}[K_\theta][L_j]$ and $\sum_{i,\theta}W_{ij\theta}[K_\theta]\sum_{j}W_{ij\theta}[L_j]$. Intuitively, $\sum_{ij}W_{ij\theta}(\cdot)$ means taking the summation after the warping image $\sum_{\theta}W_{ij\theta}[K_\theta][L_j]$. Considering the conservation of intensity, the warping operation should not influence the result very much; thus, we can remove the factor $W_{ij\theta}$ in the summation symbol, and the former operation becomes $\sum_{i,j}\sum_{\theta}W_{ij\theta}[K_\theta][L_j]$, which is exactly the summation of the blurred image $\hat{\mathbf{B}}([\mathbf{L}],[\mathbf{K}])$. For the latter operation, in a similar way, we remove the first $W_{ij\theta}$, and it turns into $\sum_{i,\theta}[K_\theta]\sum_{j}W_{ij\theta}[L_j]$, which is exactly the same as the former operation.

Based on the above discussions, the accelerations for computing the update rules from Eqs. (6) to (14) by using our system are given in Algorithm 2. (See [13] for additional information on the algorithm.)

Algorithm 2. Parameter update equations for variational based algorithm.

Initialization: $q(\mathbf{K})$, $q(\nabla\mathbf{L})$ and $q(\sigma^{-2})$

Update:

$v_\theta^{\mathbf{K}}=\langle\mathbf{K}_\theta^2\rangle-\langle\mathbf{K}_\theta\rangle^2$;

$v_j^{\mathbf{L}}=\langle\mathbf{L}_j^2\rangle-\langle\mathbf{L}_j\rangle^2$;

Project $v^{\mathbf{L}}$ into screen, move camera to θ, get $\mathbf{K}_{img}^1$;

Project $\mathbf{L}$ into screen, move camera to θ, get $\mathbf{K}_{img}^2$;

$\mathbf{K}_\theta^{(2)}=\langle\beta_\sigma\rangle\sum(\mathbf{K}_{img}^1+(\mathbf{K}_{img}^2)^2)$;

Project $\mathbf{L}$ into screen, drive camera move along current kernel $\langle\mathbf{K}\rangle$, get estimated blur image $\hat{\mathbf{B}}$;

Project $\langle\mathbf{L}\rangle(\mathbf{B}-\hat{\mathbf{B}})$ into screen, move camera to θ, get $\mathbf{K}_{img}^3$;

Project $v^{\mathbf{L}}$ into screen, drive camera move along current kernel $\langle\mathbf{K}\rangle$, get estimated blur image $\hat{\mathbf{B}}(v^{\mathbf{L}})$;

$\mathbf{K}_\theta^{(1)}\mathbf{K}_\theta^{(2)}=\langle\beta_\sigma\rangle\sum(\mathbf{K}_{img}^3+\hat{\mathbf{B}}(v^{\mathbf{L}}))+\langle\mathbf{K}_\theta\rangle\mathbf{K}_\theta^{(2)}$;

Project $\mathbf{I}$ into screen, drive camera move along $v^{\mathbf{K}}$, get estimated blur image $\mathbf{L}^{-1}$;

Project grid pattern into screen, drive camera move along $\langle\mathbf{K}\rangle$, get estimated blur image $\mathbf{PSF}_{map}$, compute norm of each impulse pixel, and interpolate $\mathbf{L}^2$;

$\mathbf{L}^{(2)}=\langle\beta_\sigma\rangle(\mathbf{L}^1+\mathbf{L}^2)$;

Project $\mathbf{B}-\hat{\mathbf{B}}$ into screen, drive camera move along current inverse kernel $\langle\mathbf{K}\rangle$, get $\mathbf{L}^3$;

$\mathbf{L}^{(1)}\mathbf{L}^{(2)}=\langle\beta_\sigma\rangle\mathbf{L}^3-\langle\beta_\sigma\rangle\sum\hat{\mathbf{B}}(v^{\mathbf{L}})+\langle\mathbf{L}\rangle\mathbf{L}^{(2)}$;

The above analysis tells us that *the most time consuming operations in the updating of the variational-based methods can be approximated by our optical computing system*. By delicately designing the updating process, the update rules of [13] can be implemented with our system, and the specific manipulations are shown in Algorithm 2.

4. Optical computing system

We design an optical computing system for accelerating the computationally intensive manipulations that are discussed above: a motion platform-based projector–camera module system.

4.1. A high-dimensional motion platform-based projector–camera system

The main difference between our projector–camera system and the commonly used system for other applications is that the camera of our system is mounted on a motion platform. Thus, the camera can move along a given trajectory or be fixed at a certain pose following a users' requirements. The diagram of the basic system is shown in Fig. 2.

Under the assumption of a pinhole camera model, the transformations from the projector to screen and the screen to camera are both homography. Assuming that the transformation function from the projector to screen is $\mathcal{T}_p(\cdot)$, an input image $\mathbf{L}$ is projected onto the screen to form $\mathcal{T}_p(\mathbf{L})$. Similarly, assuming that the transformation from the screen to the camera is $\mathcal{T}_c(\cdot)$, we obtain an output image from the camera as $\mathcal{T}_c(\mathcal{T}_p(\mathbf{L}))$. As the platform moves, the transformation from the screen to the camera $\mathcal{T}_c(\cdot)$ changes correspondingly, and it generates the captured image $\mathcal{T}_c^{\boldsymbol{\theta}}(\mathcal{T}_p(\mathbf{L}))$ at the camera pose θ. According to the perspective geometry, homography $\mathcal{T}_p(\cdot)$ and $\mathcal{T}_c^{\boldsymbol{\theta}}(\cdot)$ are both linear and can be represented by matrices. For convenience, we denote them as $\mathcal{T}_p$ and $\mathcal{T}_c^{\boldsymbol{\theta}}$, respectively.

If the motion platform moves during exposure, the resulting captured image can be represented by

$$\mathbf{B}=\int_{t\in\tau}\mathcal{T}_c^{\boldsymbol{\theta}_t}\mathcal{T}_p\mathbf{L}\,dt \tag{15}$$

where $\mathbf{B}$ is the captured blurry image, $\boldsymbol{\theta}_t$ is the camera position at time t and $\{\boldsymbol{\theta}_t : t\in\tau\}$ composes the whole camera motion trajectory, with τ being the exposure time range.

To facilitate the analysis, we define an origin position for the motion platform as θ_0, and then, the view $\mathcal{T}_c^{\boldsymbol{\theta}_t}\mathcal{T}_p\mathbf{L}$ that is captured at position $\boldsymbol{\theta}_t$ can be decomposed as $\mathcal{T}_c^{\Delta\boldsymbol{\theta}_t}(\mathcal{T}_c^{\boldsymbol{\theta}_0}\mathcal{T}_p\mathbf{L})$ with $\Delta\boldsymbol{\theta}_t=\boldsymbol{\theta}_t-\boldsymbol{\theta}_0$, and $\mathcal{T}_c^{\Delta\boldsymbol{\theta}}$ is the warping matrix $\mathbf{W}_{\boldsymbol{\theta}}$ in Eq. (2). In other words, the latent sharp version of the captured blurry image is the view that is captured at the origin position, and the blurry image is the integration of a sequence of views that is captured along a relative motion trajectory $\{\Delta\boldsymbol{\theta}_t\}$. Mathematically, if the whole transport matrix of our optical system at the origin position $\mathcal{T}_c^{\boldsymbol{\theta}_0}\mathcal{T}_p$ is an identity matrix, i.e., the image captured at the origin position θ_0 is exactly the same as the input image of the projector, then our high-dimensional motion platform projector–camera system can simulate the physical blurring process by directly moving the camera along the relative motion path $\{\Delta\boldsymbol{\theta}_t\}$.

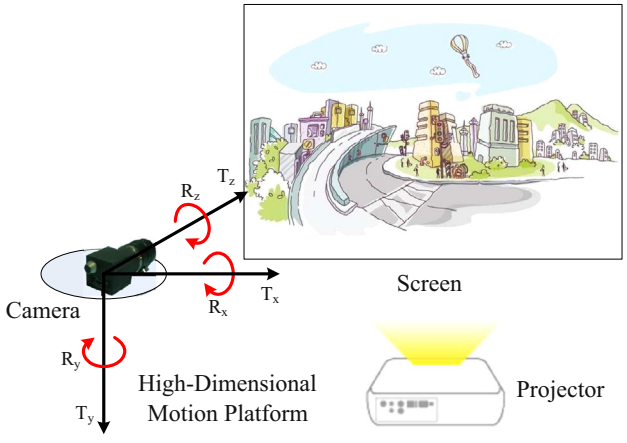

Fig. 2. Diagram of our system. The camera is mounted on the high-dimensional motion platform, and by fixing the camera to a certain pose or moving it during exposure with a motion path, the blurring and warping process can be computed quickly.

Theoretically, $\mathcal{T}_c^{\theta_0}\mathcal{T}_p = I$ indicates that the projector and the camera should be exactly the inversion of each other in optics, and optical computing systems must have precise calibration to ensure their accuracy. However, it is often difficult to find the mutually inverse projector/camera pair in practice for several reasons: (i) most commonly, the used projectors and cameras have totally different optical parameters; (ii) the precise alignment between the optical centers of the camera and the projector lenses is difficult; (iii) considering the nonlinear transformations (e.g., the gamma correction), the transport matrix-based linear model might not be sufficiently accurate in real cases. Therefore, in an implementation, we add a calibration process—both geometric and photometric—to address errors that are beyond the adopted linear model. Leaving the calibration process details to the latter experiment section, we first give the optical computation process of a time-consuming module by using our calibrated system.

Calculating a warping image sequence: The optical computation of image warping is quite simple and straightforward for our optical computing system. In practice, for a specific view pose θ_i, the snapshot by projecting $\mathbf{L}$ onto the screen and capturing it with a camera pose θ_i will obtain exactly the desired warped image $\mathbf{W}^{\theta_i}\mathbf{L}$.

Calculating a spatially varying convolution: For our optical computing system, the blurring manipulation that is caused by camera shake can also be computed simply by reproducing the physical procedure of generating a blurry image. For a given high-dimensional camera trajectory, we can capture the desired blurry image exactly by driving the motion platform along the given trajectory at a properly set exposure time with a velocity that corresponds to the elapsed times at poses along the path.

Calculating operations for the variational-based algorithm: As discussed in Section 3, after a series of approximations, the computationally intensive operations for the variational algorithm can be computed by our system, also. As shown in Algorithm 2, the update rules of each iteration can be computed by several projecting–capturing operations as well as some pixel-wise or voxel-wise manipulations.

Motion path planning: In most of the existing deblurring algorithms, camera motion parameters are represented by a set of discrete camera positions, and the corresponding intensities represent elapsed time. In other words, the camera trajectory that drives our motion platform is not directly available, and motion path planning is necessary to incorporate our optical computing system into the deblurring framework.

In this paper, a 3D rotational blur model is used, and we first construct the fully connected graph structure of the 3D kernel. Then, the motion trajectory is computed by searching for the Hamiltonian path, and the time elapse in each position is determined by the kernel intensities.

In theory, any traversal path of this connected-graph can be used as the camera trajectory here. However, to protect the platform and to reduce mechanical inaccuracy, we must reduce the abrupt direction changes along the trajectory. We define a weight on each edge that connects two vertices (camera poses) for path planning, and the weight is obviously path dependent. Path planning is performed step-wise via a Hamiltonian path search: at each time instant, the algorithm selects one vertex that has a minimum edge weight from all of the un-visited vertices and disconnects the selected vertex from the other vertices; the final path is generated by connecting the selected vertices sequentially. Specifically, we consider only the smoothness between two adjacent steps, and the edge weight at time t is defined as follows:

$$D(V_a, V_t) = \cos\left(\angle(E(V_t, V_{t-1}), E(V_t, V_a))\right), \tag{16}$$

where V_t, V_{t-1} and V_a are the current vertex, the previous vertex and an un-visited vertex, respectively. $E(\cdot,\cdot)$ is the edge between two vertices. Fig. 3 shows the motion path (b) that is estimated from the discrete 3D kernel (a). One can see that there are only 3 brute angles and that most of the sub-trajectories is smooth.

5. Implementation and experiment results

This section demonstrates our prototype optical computing system and, then, shows its performance in performing predefined non-uniform blurring and restoring blurry images after being incorporated into a deconvolution framework.

The whole system is shown in Fig. 4. For simplicity, we use a 3-axis rotating platform and adopt a 3-D rotational blur model. This rotating platform can burden an approximately 40 kg load (camera in our scenario) and moves to an orientation with any pitch, yaw and roll angle. The maximum moving velocity is 200°/s, and the rotating acceleration is 200°/s^2, which are sufficient for

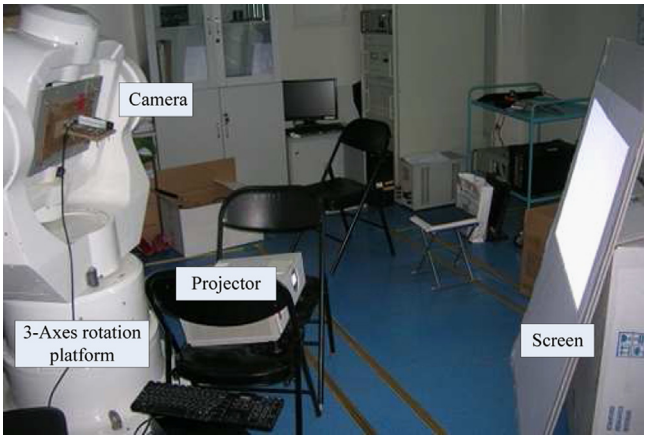

Fig. 4. The prototype of a proposed high-dimensional motion platform-based projector–camera system.

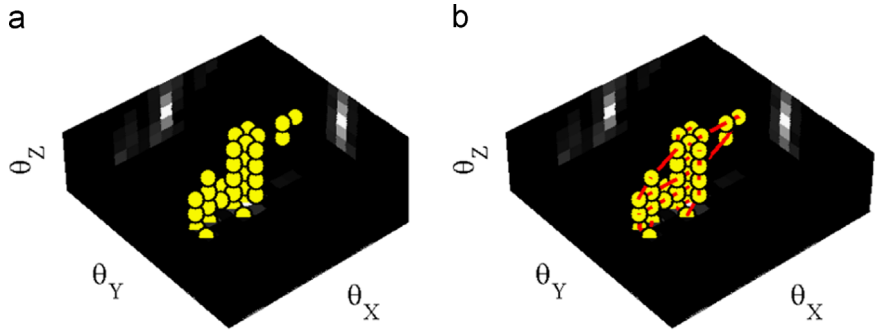

Fig. 3. 3D rotation kernel to motion path. (a) Intermediate 3D rotation kernel estimated in variational algorithm [12]. (b) Motion path estimated from the discrete 3D kernel.

our prototype system to validate the proposed approach. The camera mounted on the platform is a Point Grey FL2-08S2C. The platform and the camera are driven by control software through an RS-232 COM and IEEE 1394, respectively. We adopt software synchronization to ensure exact matching between the camera's exposure time and the given platform's motion trajectory.

5.1. Projector–camera system calibration

The hybrid system should be calibrated in terms of the geometric distortion, the intensity response curve and other distortions. Ideally, we should calibrate the projector and the camera independently. For simplicity and efficiency, we treat the projector–camera system as a black box, and the experiments show that our simple calibration method is sufficiently accurate to prevent the computing process from suffering from distortions.

Geometric calibration: Because the projector and the camera cannot be exactly complementary (with the same optical path), we need a careful calibration to ensure that there is an exact correspondence between the input image and the captured image. One straightforward method is to calibrate all of the parameters of the projector–camera system with a series of calibration patterns. However, this method is complex and could suffer from unmodeled distortions. Fortunately, we only need the geometric correspondence between the input and output image, and we can use a coordinate map to represent the correspondence instead of an explicit geometric model. In the implementation, we provide a chessboard pattern to the projector and capture the projected result with the camera. By corner matching, the correspondence between the input and output images is determined. Because the coordinate correspondence varies smoothly, we can largely reduce the computation time by interpolating the corresponding map from several corner points. Fig. 5 gives an example of our geometric calibration, with Fig. 5(a) being the original calibration pattern, Fig. 5(b) being the captured version by our projector–camera system with the motion platform fixed at the origin position, and Fig. 5(c) and (d) presenting the mapping vectors at several landmarks for the interpolation and the calibrated pattern, respectively.

Dark corner correction: Both the projector and camera suffer from dark corner effects, and the effect is more significant in such hybrid systems. To offset the dark corner, we project a constant-intensity grey image (the intensity is set to 128 to prevent under/over exposure) to the screen, and we compute the ratio image between the geometrically calibrated output image and the original input image. Fig. 6(a) and (b) shows the captured grey image and the geometrically calibrated ratio image, respectively, which can be used to remove the dark corner effect.

Intensity response calibration: Because the response curve of neither the projector nor the camera is ideally linear due to some internal transformations such as gamma correction, the intensity response of the whole system varies nonlinearly with the intensity and must be calibrated.

There are several contributing factors (e.g., the response curve of the projector, the reflection properties of the screen, and the response curve of camera) for the nonlinearity; we prefer a black-box model here. We first project an intensity gradation map onto the screen; then, we correct the geometric distortion and dark corner effect of the output image, and next, we compute the response of each intensity level by averaging all of the pixels at this level (we remove the marginal pixels). Fig. 7 gives an example of an intensity response calibration, (a) gives the original intensity step pattern, (b) shows the captured image, (c) is the calibrated version of (b) with a geometric and dark corner correction, and (d) demonstrates the calibrated intensity response curve that is derived by our intensity step map.

In addition, considering that the color channels of a common RGB camera and projector are coupled with each other and that decoupling them will increase the complexity of our system, we

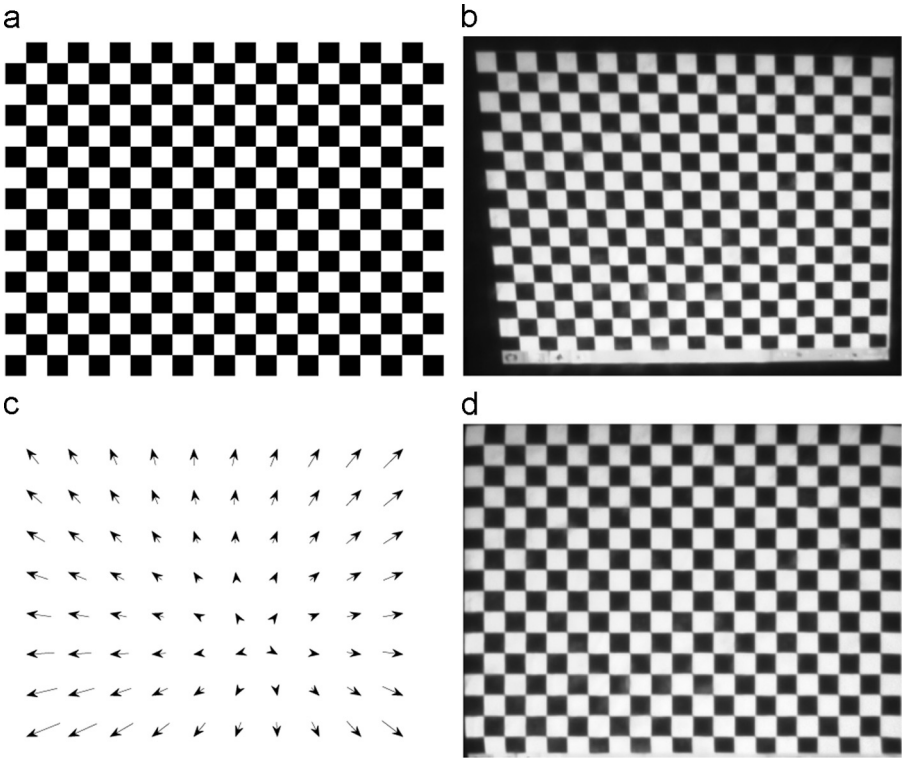

Fig. 5. Result of geometric calibration. (a) Original chessboard pattern, (b) captured pattern, (c) warping vectors from landmarks for interpolation and (d) pattern after geometric calibration.

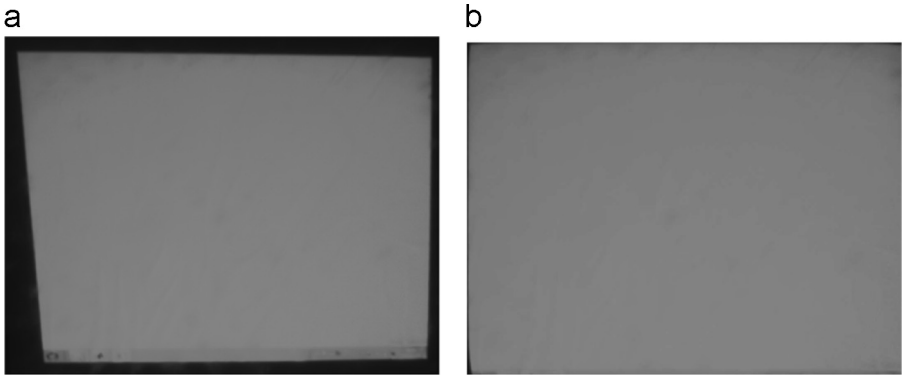

Fig. 6. Result of dark corner correction. (a) and (b) are captured images of a projected uniform-intensity pattern before and after calibration, respectively.

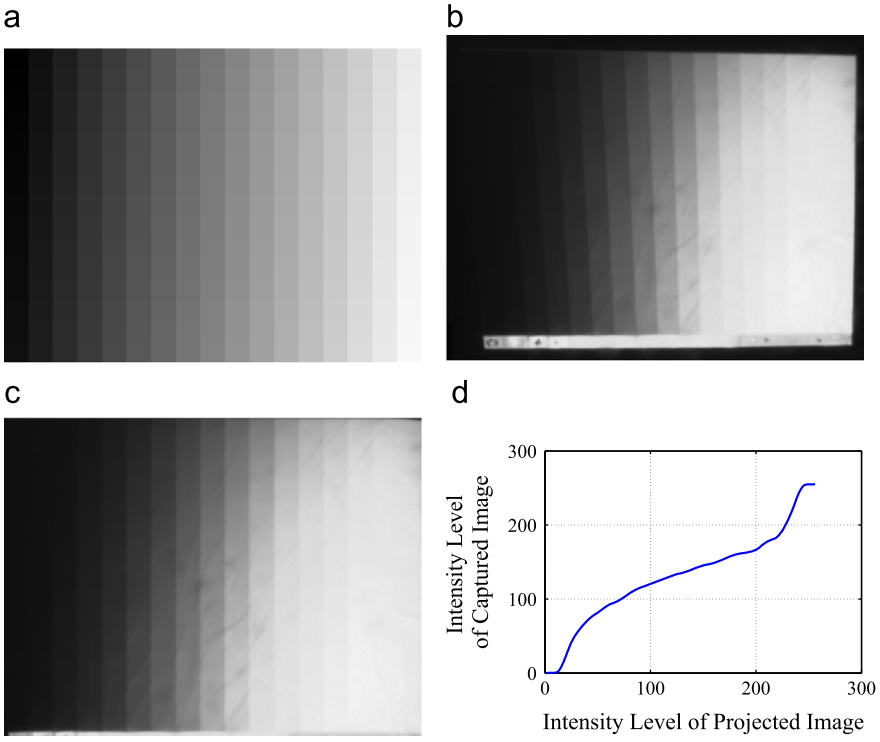

Fig. 7. Intensity response calibration. (a) Original intensity step pattern, (b) captured pattern, (c) pattern after geometric calibration and dark corner correction, and (d) intensity response curve.

calibrate only the grey-scale images in our prototype system and process each channel separately.

5.2. Experiment results on prototype validation

Accuracy of PSF: To verify the accuracy of our high-dimensional motion platform-based projector–camera system, we project a point grid onto the screen and capture the deterioration result (i.e., the PSF) by a randomly generated camera shake.

Comparing the synthetic blurred result (with the camera intrinsics and trajectory known) in Fig. 8(a) and the image captured by our high-dimensional motion platform projector–camera system in Fig. 8(b), we can see an extremely high level of similarity. Apparently, our system provides a promising approximation and thus is of sufficient accuracy for performing the blurring manipulation optically.

Accuracy of the non-uniform blur simulation: A sharp image is projected onto the screen, as shown in Fig. 9(a), for which the ground truth of the blurry version under a given camera shake is shown in Fig. 9(b). Then, the blurry image that is optically computed by our system is captured under the given motion trajectory, as shown in Fig. 9(d). For a clearer comparison, we display in Fig. 9(c) the absolute residual map between (b) and (d). The small residue validates the accuracy of our optical computing system. The residual errors are caused mainly by two factors: the dead/saturation area of the response curve (shown in Fig. 7(d)) and the distortion that is not modeled in the synthesizing ground truth blurry image, such as the radial distortion or tangential distortion.

Non-blind deblurring with our system: We incorporate our system into the framework of non-blind non-uniform motion deblurring, with a randomly given 3D camera shake trajectory. For simplicity, we adopt a non-uniform version of the RL [1,15] deblurring algorithm in this experiment. Introducing our high-dimensional motion platform projector–camera system, we can replace the pixel-wise non-uniform convolution in each iteration with two snapshots; thus, the running time can be largely reduced. Neglecting the mechanical limitations of the system, it takes slightly longer than 1/15 s for one iteration when using a 30 fps digital camera. In comparison, the pixel-based methods are

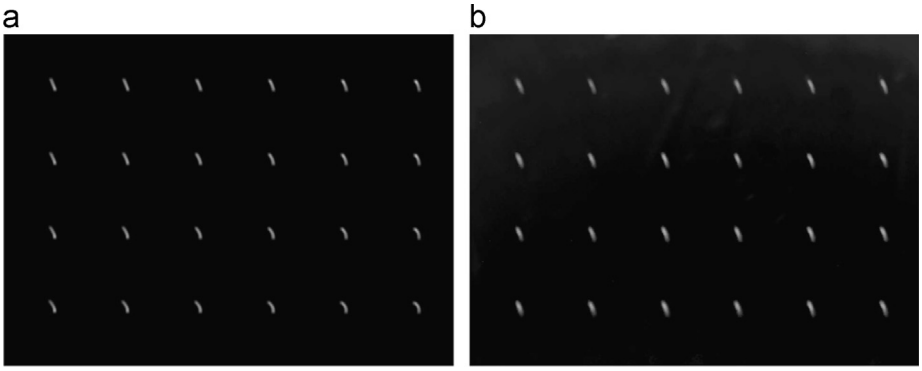

Fig. 8. Testing the PSF accuracy by using a point grid pattern. (a) Synthetic result according to the projective geometry. (b) Blurred point grid pattern by our prototype optical computing system.

Fig. 9. Result of spatially varying blurring. (a) A sharp image, (b) synthetic blurry image from projective geometry, (d) blurry result generated by our optical computing system, and (c) absolute difference between (b) and (d).

an order of magnitude slower than our system even with the GPU acceleration, and the patch-based methods implemented on the GPU are also much slower than ours, especially in the case of a large image size.

Fig. 10(a) and (e) shows the blurry image and true sharp image, respectively. The estimated sharp images and residual maps at iterations 1, 10, 20 are shown sequentially in Fig. 10(b)–(d) and (f)–(h). The increasing sharpness and decreasing residue both validate that our system can be incorporated into a deblurring framework easily, to raise the efficiency without introducing large artifacts.

Blind non-uniform deblurring: To verify our system for blind non-uniform deblurring, we incorporate the system into the algorithm by the method of Whyte et al. [12] and demonstrate its effectiveness by displaying both several intermediate variables and the final results. As discussed in Section 3, the variational-based algorithm has several specific manipulations that cannot be replaced directly but can be approximated by non-uniform convolution. Fig. 11 shows the comparison of some intermediate results between the accurate computation and the approximations by our optical computation framework. Specifically, Fig. 11(a) and (b) shows the accurate version and our approximation of the intermediate variables $\mathbf{L}^{(2)}$ that were described in Algorithm 2, and Fig. 11(c) and (d) give those of the intermediate variable $\mathbf{L}^{(1)}\mathbf{L}^{(2)}$. Note that the gradient of the latent image instead of itself is used in the kernel estimation stage; thus, each intermediate variable includes two components, i.e., the horizontal version and the vertical version. For convenience of computation and presentation,

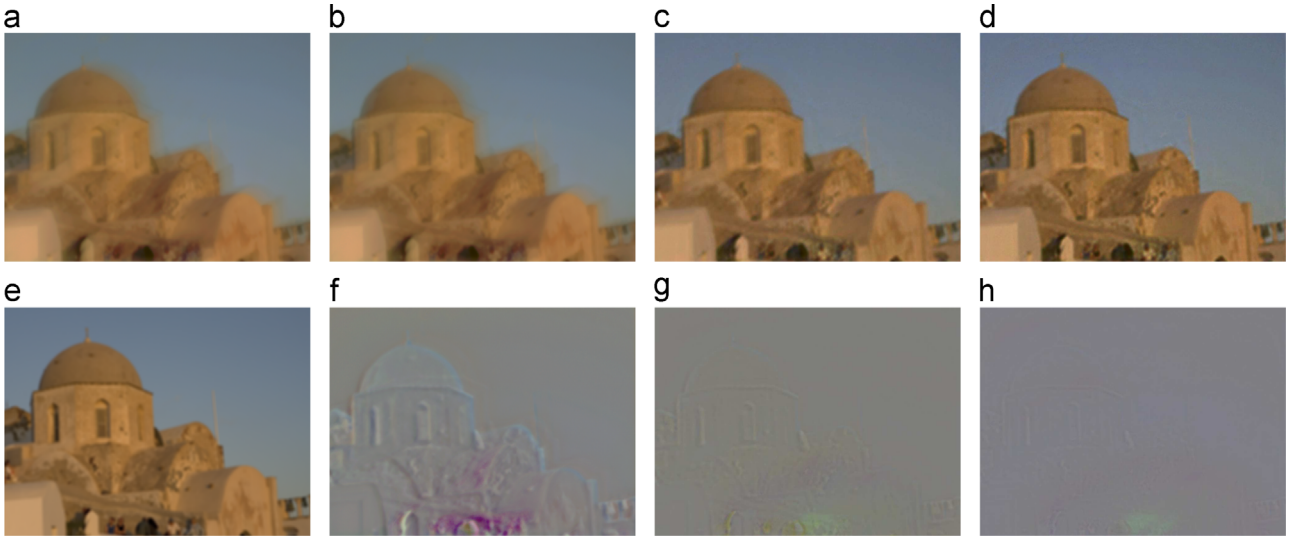

Fig. 10. Result of fast non-blind deblurring with our optical computing system. (a) Blurry image, (e) sharp image. (b)–(d) Estimated sharp image at the 1st, 10th and 20th iterations. (f)–(h) The residual error map of (b)–(d) with respect to (a).

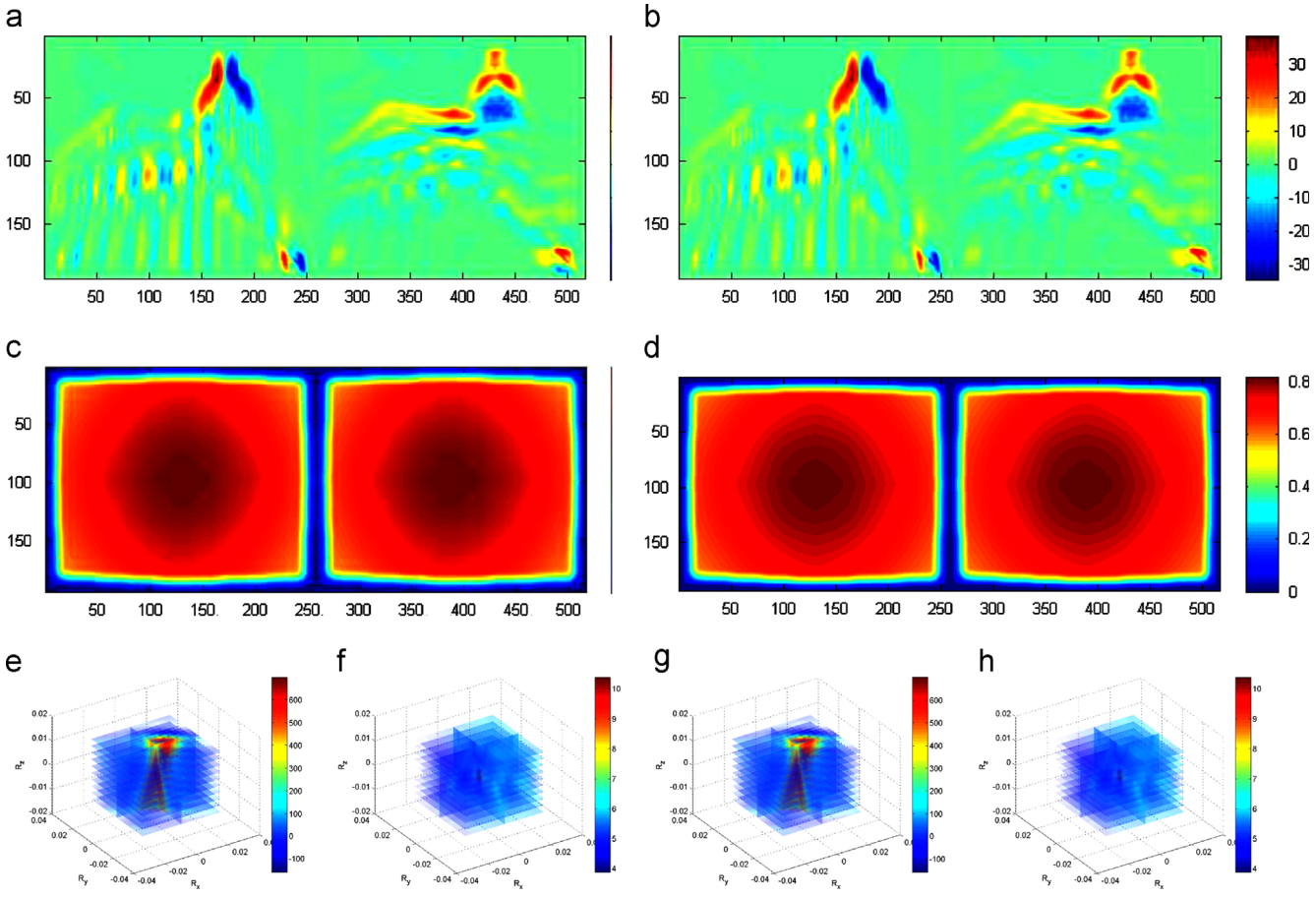

Fig. 11. Intermediate results of a variational-based blind deblurring algorithm under our optical computation framework. (a), (c), (e), and (f), respectively illustrate the ground truth of four intermediate variables—$\mathbf{L}^{(2)}$, $\mathbf{L}^{(1)}\mathbf{L}^{(2)}$, $\mathbf{K}^{(2)}$ and $\mathbf{K}^{(2)}\mathbf{K}^{(2)}$—in one update iteration. (b), (d), (g), and (h) displays the corresponding approximated results.

we place the horizontal and vertical versions of $\mathbf{L}^{(2)}$ and $\mathbf{L}^{(1)}\mathbf{L}^{(2)}$ side-by-side in Fig. 11(a)–(d). Fig. 11(e) and (f), and (g) and (h) compare the true version and our approximation of variables $\mathbf{K}^{(2)}$ and $\mathbf{K}^{(1)}\mathbf{K}^{(2)}$, respectively. Obviously, our results have a significantly high accuracy.

The final results, which include both the camera motion and the latent sharp image, are shown in Fig. 12, from which we can see that our optical computation framework obtains comparable performance. Although we apply some approximations to the algorithm for incorporating the optical computation, the fact that

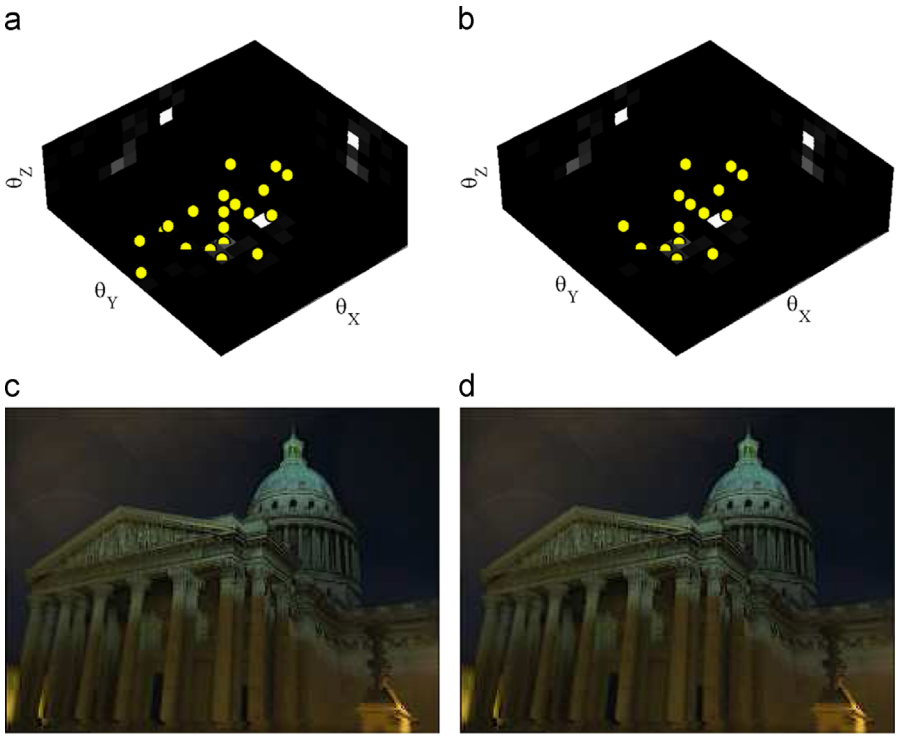

Fig. 12. Final results for the variational-based blind deblurring algorithm. (a) and (c) 3D rotation kernel and latent sharp image estimated by the algorithm in [12]. (b) and (d) Our final estimation of the 3D blur kernel and the latent sharp image, respectively.

the difference of (a) and (c) vs. (b) and (d) is not noticeable indicates that the approximation error hardly deteriorates the deblur performance. Overall, our non-uniform convolution framework can give promising results by replacing the computation-consuming operations with optical computations in the blind non-uniform deblurring algorithm.

6. Summary, analysis and discussion

This paper demonstrates an optical computing system that significantly accelerates the conventional non-uniform motion deblurring. The framework is fast, has high accuracy and additionally has flexible extensions that are worthwhile studying in the future. A discussion of some of the implementation details that are worthy of note and limitations and extensions is included in the following paragraphs.

Effect of noise. The proposed optical computing system will inevitably introduce noise. To test the influence of sensor noise during the optical computing process, we added some CCD noise to the non-uniform convolution results in the corresponding step of the RL algorithm and tested the final deblurring performance. Here, the widely used Poisson–Gaussian noise model [40] is used, as follows:

$$\mathbf{I}_o = \mathbf{I} + (\alpha\sqrt{\mathbf{I}} + \beta)\mathbf{N} \tag{17}$$

here $\mathbf{I}_o$ is the observed image, $\mathbf{I}$ is the original image, and $\mathbf{N}$ is independent random noise with a standard deviation that is equal to 1, α and β are the noise-level parameters. We vary the noise level from $\alpha = 0.01, \beta = 1.6e-5$ to $\alpha = 0.08, \beta = 2.56e-5$, and the results are shown in Fig. 13. From these results, we can see that the RL algorithm converges after 20 iterations even at the largest noise level, and the PSNR of the deblurred images slightly decreases as the noise level increases. Therefore, we recommend high projector illuminance to reduce the noise effect. In addition, the prior constraints on the latent sharp image, which are widely used in the deblurring algorithm, are also useful in the noise suppression.

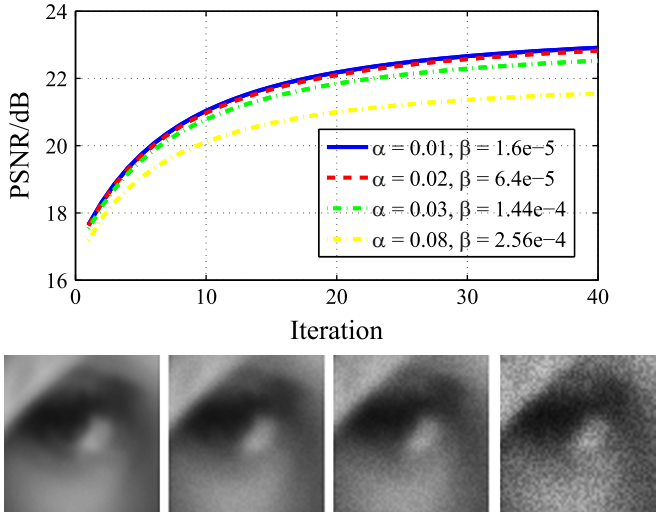

Fig. 13. Effects of the sensor noise on the final performance. We synthetically add noise to the computed blurry images to simulate noise contamination in the final optical computing process. The bottom close-ups show the noise level $\alpha = 0.01, \beta = 1.6e-5$; $\alpha = 0.02, \beta = 6.4e-5$; $\alpha = 0.03, \beta = 1.44e-4$ and $\alpha = 0.08, \beta = 2.56e-4$, respectively.

Benefits and limitations: Benefitting from the fast light transport and the parallelism of the imaging system, optical computing systems can perform specific operations very fast. Specifically for our system, each CCD unit acts as an individual computing unit, and each snapshot can achieve parallel processing of Mega- or even Giga-pixels; thus, the proposed optical computing system provides a feasible and promising solution for fast non-uniform motion deblurring.

Our prototype has three main limitations: the high cost of the motion platform, the limited frame rate of the camera and the assumption of depth independence for the blur patterns. (i) In our implementation, we adopt a high-end rotating platform that has a large angular velocity, accelerations, payload and very high precision (0.001°); we can also use a high-end 6-D platform for arbitrary motion. However, both of the above platforms are too costly for consumer imaging systems. Considering the small weight and size of the consumer cameras, the moment of inertia of the camera that is used in our system can be much smaller than the platform limitation. As a rough estimation, approximately 1/40 of the upper bound of the adopted platform is sufficient. Therefore, we can choose a low-load motion platform, in such a way that the motion velocity can be improved to shorten the running time further. (ii) For the camera's frame rate, in our experiment, the Point Grey Flea2 08S2C camera can achieve 30 fps with a resolution of 1024×768 pixels, in such a way that our system can finish at most 30 times the blurring manipulations or frame-wise inner product manipulations within 1 s. The computation can be further accelerated by using a higher frame-rate camera and projector. (iii) With a planar screen, our system cannot simulate depth-dependent blurring; hence, layered decomposition must be performed, and more snapshots are necessary in such cases.

Promising extensions: The optical computational framework can be applied not only to camera shake removal but also to the out-of-focus blur. Instead of using a motion platform, a camera with a programmable focal length can simulate the defocus blur at any of the focal settings.

Thus far, our system cannot address image blur that is caused by camera translation (which cannot be approximated by rotation), but this concern can be addressed by replacing our platform with a 6-axis Stewart platform. Studying an arbitrary camera shake removal approach and building the corresponding optical computing system are two future directions to be pursued. The efficiency superiority of our framework will be more significant in cases that have higher degrees-of-freedom camera shakes.

Appendix A. Supplementary materials

Supplementary data associated with this article can be found in the online version at http://dx.doi.org/10.1016/j.cag.2013.10.011.

References

[1] Richardson W. Bayesian-based iterative method of image restoration. J Opt Soc Am 1972;62(1):55–9.
[2] Fergus R, Singh B, Hertzmann A, Roweis S, Freeman W. Removing camera shake from a single photograph. ACM Trans Graph 2006;25(3) [Proceedings of SIGGRAPH].
[3] Shan Q, Jia J, Agarwala A. High-quality motion deblurring from a single image. ACM Trans Graph 2008;27(3) [Proceedings of SIGGRAPH].
[4] Cho S, Lee S. Fast motion deblurring. ACM Trans Graph 2009;28(5) [Proceedings of SIGGRAPH Asia].
[5] Xu L, Jia J. Two-phase kernel estimation for robust motion deblurring. In: ECCV; 2010.
[6] Cho TS, Levin A, Durand F, Freeman WT. Motion blur removal with orthogonal parabolic exposures. In: ICCP; 2010.
[7] Krishnan D, Tay T, Fergus R. Blind deconvolution using a normalized sparsity measure. In: CVPR; 2011.
[8] Cho TS, Paris S, Horn, BK, Freeman WT. Blur kernel estimation using the radon transform. In: CVPR; 2011.
[9] Cho LZS, Metaxas D, Paris S, Wang J. Handling noise in single image deblurring using directional filters. In: CVPR; 2013.
[10] Leith E. The evolution of information optics. IEEE J Sel Top Quantum Electron 2000;6(6):1297–304.
[11] Ambs P. A short history of optical computing: rise, decline, and evolution. In: International conference on correlation optics; 2009.
[12] Whyte O, Sivic J, Zisserman A, Ponce J. Non-uniform deblurring for shaken images. In: CVPR; 2010.
[13] Whyte O, Sivic J, Zisserman A, Ponce J. Non-uniform deblurring for shaken images. Int J Comput Vision 2012;98(2):168–86.
[14] Gupta A, Joshi N, Zitnick CL, Cohen MF, Curless B. Single image deblurring using motion density functions. In: ECCV; 2010.
[15] Tai Y, Tan P, Brown M. Richardson–Lucy deblurring for scenes under projective motion path. IEEE Trans Pattern Anal Mach Intell 2011;33(99):1603–18.
[16] Tai Y, Kong N, Lin S, Shin S. Coded exposure imaging for projective motion deblurring. In: CVPR; 2010.
[17] Joshi N, Kang S, Zitnick C, Szeliski R. Image deblurring using inertial measurement sensors. ACM Trans Graph 2010;29(4) [Proceedings of SIGGRAPH].
[18] Hirsch M, Sra S, Scholkopf B, Harmeling S. Efficient filter flow for space-variant multiframe blind deconvolution. In: CVPR; 2010.
[19] Hu Z, Yang MH. Fast non-uniform deblurring using constrained camera pose subspace. In: BMVC; 2012.
[20] Sorel M, Flusser J. Space-variant restoration of images degraded by camera motion blur. IEEE Trans Image Process 2008;17(2):105–16.
[21] Xu L, Jia J. Depth-aware motion deblurring. In: ICCP; 2012.
[22] Yue T, Suo J, Ji X, Dai Q. Optical computing system for fast non-uniform image deblurring. In: CCD; 2013.
[23] Heinz RA, Artman JO, Lee SH. Matrix multiplication by optical methods. Appl Opt 1970;9(9):2161–8.
[24] Dias A. Incoherent optical matrix–matrix multiplier. NASA Langley Res Center Opt Inform Process for Aerospace Appl 1981:71–83.
[25] Athale RA, Collins WC. Optical matrix–matrix multiplier based on outer product decomposition. Appl Opt 1982;21(12):2089–90.
[26] Huang H, Liu L, Wang Z. Parallel multiple matrix multiplication using an orthogonal shadow-casting and imaging system. Opt Lett 1990;15(19):1085–7.
[27] Guilfoyle P, Stone R. Digital optical computer ii. In: Optical enhancements to computing technology. International Society for Optics and Photonics; 1991.
[28] Lewis A, Albeck Y, Lange Z, Benchowski J, Weizman G. Optical computation with negative light intensity with a plastic bacteriorhodopsin film. Science 1997;275(5305):1462–4.
[29] O'Toole M, Kutulakos K. Optical computing for fast light transport analysis. ACM Trans Graph 2010;29(6) [Proceedings of SIGGRAPH].
[30] Lefebvre D, Arsenault H, Roy S. Nonlinear filter for pattern recognition invariant to illumination and to out-of-plane rotations. Appl Opt 2003;42(23):4658–62.
[31] Yu F, Jutamulia S, Lin T, Gregory D. Adaptive real-time pattern recognition using a liquid crystal TV based joint transform correlator. Appl Opt 1987;26:1370–2.
[32] Levin A, Fergus R, Durand F, Freeman WT. Image and depth from a conventional camera with a coded aperture. ACM Trans Graph 2007;26(3):70 [Proceedings of SIGGRAPH].
[33] Zhou C, Nayar S. What are good apertures for defocus deblurring? In: ICCP; 2009.
[34] Masia B, Presa L, Corrales A, Gutierrez D. Perceptually optimized coded apertures for defocus deblurring. Comput Graph Forum 2012;31(6):1867–79.
[35] Levin A, Weiss Y, Durand F, Freeman WT. Understanding and evaluating blind deconvolution algorithms. In: CVPR; 2009.
[36] Joshi N, Szeliski R, Kriegman DJ. Psf estimation using sharp edge prediction. In: CVPR; 2008.
[37] Xu L, Zheng S, Jia J. Unnatural l_0 sparse representation for natural image deblurring. In: CVPR; 2013.
[38] Stewart C. Robust parameter estimation in computer vision. SIAM Rev 1999;41(3):513–37.
[39] Miskin J, MacKay DJ. Ensemble learning for blind image separation and deconvolution. Adv Independent Component Anal 2000:123–41.
[40] Foi A, Trimeche M, Katkovnik V, Egiazarian K. Practical Poissonian–Gaussian noise modeling and fitting for single-image raw-data. IEEE Trans Image Process 2008;17(10):1737–54.