Multi-Shot Deblurring for 3D Scenes

M. Arun, A. N. Rajagopalan
Department of Electrical Engineering
Indian Institute of Technology Madras, Chennai, India
{ee14s002,raju}@ee.iitm.ac.in

Gunasekaran Seetharaman
Information Directorate
AFRL/RIEA, Rome NY, USA
gunasekaran.seetharaman@us.af.mil

Abstract

The presence of motion blur is unavoidable in hand-held cameras, especially in low-light conditions. In this paper, we address the inverse rendering problem of estimating the latent image, scene depth and camera motion from a set of differently blurred images of the scene. Our framework can account for depth variations, non-uniform motion blur as well as mis-alignments in the captured observations. We initially describe an iterative algorithm to estimate ego motion in 3D scenes by suitably harnessing the point spread functions across the blurred images at different spatial locations. This is followed by recovery of latent image and scene depth by alternate minimization.

1. Introduction

With light hand-held imaging devices becoming the norm of the day, motion blur has become a dominant artifact in captured images. A naive solution to alleviate this problem would be to capture multiple images and choose the best (most focused) among them. But this relies on the assumption that there is an unblurred image in the captured sequence. Since the burst-mode is a common feature in smart phone cameras, it is straight-forward these days to capture several images in one go. Many works [6, 15, 20] also exist that specifically address the problem of removing blur due to camera shake in a given image. While these are some of the ways to mitigate the effect of motion blur, the downside is that they discard the information about the scene as well as camera motion that are implicitly embedded in the blur.

Smart phone cameras are inherently limited in their size and have a small aperture. Since low light and indoor scenes need to be exposed longer to gather sufficient light, motion blur is difficult to avoid in these scenarios. In fact, all the captured images can turn out to be blurred. When we examine the forward problem of rendering motion blur in a scene, many physical properties of the scene need to be known. This includes knowledge of the latent (clean) image of the scene, camera motion, and the scene depth. Inverse rendering is about inferring these properties from a set of motion blurred images. It is this challenging task that we attempt in this paper.

Researchers have attempted this problem but under simplifying assumptions on camera motion and scene depth. Among the works that use two or more images, the simplest strategy to avoid motion blur is termed as '$Lucky\ imaging$' which is about capturing multiple images at a time and picking the best one. Since this does not take full advantage of all the observations, it led to the developemnt of multi-image methods for motion deblurring. Šroubek et al. [16], Ito et al. [9] and Zhang et al. [23] estimate a common latent image from a set of uniformly blurred images. Cho et al. [4] use two space-variantly blurred images and iteratively estimate the latent image and camera motion. Zhang et al. [22] propose a joint alignment, deblurring and resolution-enhancement frame work using a set of non-uniformly blurred images. Among methods that work on a single image, most of the approaches assume space-invariant blur and recover the sharp latent image by deconvolution [6, 15, 20]. The convolution model is very restricted and is valid only when the scene is planar and the camera motion consists of in-plane translations. However, in a real scenario, blur due to hand-shake tends to be space-variant [12]. In fact, even small degrees of rotations can result in non-uniform blur. In a depth-varying scene, even in-plane translations can cause space-variant blurring through position dependent scale changes of the PSF. In the literature on non-uniform blurring, the projective transform model has received considerable attention. Tai et al. [18] proposed a modified Richardson Lucy deconvolution method for this model. Whyte et al. [19] model the transformation space by considering camera rotations alone within a variational Bayesian framework. Gupta et al. [7] and Paramanand et al. [14] model camera motion considering in-plane translations and rotations.

It must be highlighted that most of these methods cannot handle depth-varying scenes. Because Whyte et al. [19] employ a purely rotational model, their deblurring is independent of scene depth. Consequently, it cannot recover

the depth map. Cho et al. [5] and Paramanand et al. [14] proposed non-uniform deblurring for the limited case of bilayer scenes. Moreover, conventional methods also assume the images to be aligned [23]. However, in hand-held scenarios, along with intra-frame motion (i.e., space varying motion blur) one also encounters inter-frame motion, leading to mis-alignments among the observations. Aligning such blurred images is inherently a difficult proposition.

1.1. Overview of our method

The main contribution of our work is to develop a scheme to recover the latent image, camera motion and scene depth from an all-blurred set consisting of multiple non-uniform motion blurred images. Following others, we too assume a static scene and in-plane translations and rotations for camera motion. This is a reasonable assumption as small out-of-plane rotations also can be well-approximated by in-plane translations [19]. We allow for the frames to have independent inter-frame and intra-frame motion which throws up the challenge of realistically modeling mis-alignments. We first estimate the camera motion corresponding to each image using the blur kernels obtained from points at different depths in the blurred observation by modeling both inter-frame and intra-frame motion within a single camera pose space. With the estimated camera motion, we iteratively solve for latent sharp image and scene depth.

Even though our approach bears some similarities to [14], our method is not restricted to bilayer scenes. Also, we do not assume knowledge of latent image unlike [13]. Among the approaches closest to our method are those by Lee et al. [11] and Hu et al. [8]. Lee et al. estimate scene depth from a set of non-uniformly blurred images. They formulate depth estimation with *known* camera motion which is obtained aprori by a camera localization algorithm. In contrast, we estimate the camera motion from the blurred images themselves, since motion blur itself is principally caused by camera shake. Hu et al. estimate depth and latent image simultaneously from a single non-uniformly blurred image. They solve it as a segment wise depth estimation problem by assuming a discrete-layered scene where each segment corresponds to one depth layer. Thus, any errors in multi-layer segmentation will directly impact the accuracy of depth estimation. They also require user input for segmenting the layers. Moreover, they cannot handle continuously varying depths such as an inclined plane. In contrast, our method is automatic and can also handle inclined plane. However, we need multiple blurred images of the scene.

The rest of the paper proceeds as follows. In Section 2, we briefly review the forward image formation model. Section 3 discusses how global camera motion can be estimated from locally estimated blur cues with special emphasis on depth-varying scenarios. Section 4 gives details about our iterative estimation framework for recovering the latent image and scene depth. Experiments are presented in Section 5 and we conclude with Section 6.

2. Motion Blur Model: Background

Consider a static scene. Let f be the latent image and Γ denote the motion trajectory space of the camera. Γ defines a set of rotations and translations (Θ, T) based on the position of the camera with respect to a reference position. Then the motion blurred image g can be expressed as

$$g = \int_{t=0}^{t=\Delta t} \Gamma_t(f) dt, \tag{1}$$

where Δt is the exposure time and $\Gamma_t(f)$ represents the result of a geometric transformation applied on the latent image f at time instant t. The transformation itself depends on the parameters Θ_t and T_t defined by Γ. The trajectory space can be sampled to obtain N discrete camera poses $(\Theta_i, T_i) \in \mathcal{T}$, where $\mathcal{T}$ is the set of all transformations and $1 \leq i \leq N$. The Transformation Spread Function (TSF) [13] or equivalently the Motion Density Function (MDF) [7] is a function that maps the set of all transformations $\mathcal{T}$ to a non-negative real number i.e., $w : \mathcal{T} \rightarrow \mathbb{R}^+$. The assigned real number w_i to the i^{th}transformation represents the fraction of exposure time the camera spent in that transformation. The motion blurred image g of Eq. (1) can also be equivalently expressed as

$$g = \sum_{i=1}^{N} w_i . \mathcal{H}_{\Theta_i, T_i}(f), \tag{2}$$

where $\mathcal{H}_{\Theta_i, T_i}(f)$ is the transformed latent image for a transformation defined by (Θ_i, T_i).

It is well-known that general camera motion can be reasonably approximated by 3 degrees of motion, viz., in-plane translations and in-plane rotation [12]. We too adopt this reduced motion model. If we define the parameter Θ as the rotation angle about the Z axis and T as translations along X and Y axes, then $\Theta = \theta_Z$ and $T = [t_X \quad t_Y \quad 0]^T$. The set of all transformations $\mathcal{T}$ is then a 3D space defined by the axes t_X, t_Y and θ_Z. The homography $H(\Theta_i, T_i)$ then can be written as

$$H(\Theta_i, T_i) = \begin{bmatrix} \cos({\theta_Z}^i) & -\sin({\theta_Z}^i) & {t_X}^i \\ \sin({\theta_Z}^i) & \cos({\theta_Z}^i) & {t_Y}^i \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

As rotations are involved, the motion blur induced will vary as a function of the spatial location of the image.

Assume a point source at location (x, y) and let $h_{(x,y)}$ denote the Point Spread Function (PSF) at that location. For each transformation $(\Theta_i, T_i) \in \mathcal{T}$ the point source is transformed to a new coordinate (x_i, y_i) based on $H(\Theta_i, T_i)$ and

weighted by w_i. Hence, the PSF at each pixel (x, y) in discrete form is given by

$$h_{(x,y)} = \sum_{i=1}^{N} w_i . \mathcal{H}_{\Theta_i, T_i}(\delta_{x,y}), \tag{4}$$

where $\delta_{x,y}$ denotes 2D Kronecker delta and represents a point source at (x, y). The relation in Eq. (4) between the TSF and the PSF will be leveraged subsequently.

3. Camera Motion Estimation in 3D scenes

Since motion blur in the blurred images is independent, camera motion needs to be estimated for each blurred image separately. However, given that we are dealing with multiple blurred observations, we postulate that for effective latent image recovery, their motion must be computed with respect to a common reference. We achieve this by judiciously exploiting the information available across all the images as described next.

We first estimate the PSF at N_p different spatial locations in N_b blurred images i.e., PSFs at identical locations across the images are estimated simultaneously. Even though the camera motion is the same for a given observation, each scene point experiences a different transformation depending on its own individual depth. Assuming the blur and depth variations to be uniform within a small patch around a point, a blurred patch can be represented *locally* as a convolution of a latent patch with PSF. While one could use a single-image blind space-invariant technique to determine the PSF locally at a given location, we choose to adopt the multi-channel framework of [16] as it can provide PSFs with respect to a common latent patch. Thus the PSFs at locations p_j $(j = 1, 2, .., N_p)$, in each blurred image g_n $(n = 1, 2, ..., N_b)$ can be estimated by minimizing

$$\mathcal{E}(f_{p_j}, h^1_{p_j}, h^2_{p_j}, ..., h^{N_b}_{p_j}) = \sum_{n=1}^{N_b} \parallel h^n_{p_j} * f_{p_j} - g^n_{p_j} \parallel_2^2$$
$$+\lambda_1 \parallel \nabla f_{p_j} \parallel_1 + \lambda_2 \sum_{l,m \in [1,N_b], l \neq m} \parallel h^m_{p_j} * g^l_{p_j} - h^l_{p_j} * g^m_{p_j} \parallel_2^2$$
$$j = 1, 2, ..., N_p \tag{5}$$

where λ_1 and λ_2 are regularization parameters, f_{p_j} is the estimated latent image patch around spatial location p_j, and $h^n_{p_j}$ is the estimated PSF at location p_j for the blurred image g_n using blurred patch $g^n_{p_j}$.

Because the PSFs at N_p different locations are estimated independently, this can result in undesired shifts among the estimated PSFs within a single image. This is due to the fundamental fact that $t(f_{p_j}) * h_{p_j} = f_{p_j} * t(h_{p_j})$, where $t(.)$ denotes a shift that can vary across p_j locations. We employ the blur kernel alignment algorithm of [14] to align the PSFs estimated from a particular image with respect to the PSF from a reference location. After this alignment step, all the PSFs correspond to a common latent image f.

Translational components alone are affected by depth variations. The PSF h at a point (x, y) can be expressed as

$$h_{(x,y)} = \sum_{i=1}^{N} w_i . \mathcal{H}_{\Theta_i, k(x,y)T_i}(\delta_{x,y}) \tag{6}$$

where $k(x, y)$ is a scale factor with respect to a reference depth d_0 and $k(x, y) = \frac{d_0}{d(x,y)}$ [13]. The nearest depth plane from camera is taken as d_0, and hence $k \in [0, 1]$. An estimate of k yields the relative depth of the scene. From Eq. (4) we know that the PSF at a location is the weighted sum of individual transformations from the TSF space applied to a point source at that location. This can be expressed as

$$h^n_{p_j} = M^n_{p_j} w^n, \tag{7}$$

where the i^{th} column of $M^n_{p_j}$ is the result of applying the transformation corresponding to w^n_i to the point source at p_j. Since entries of this matrix $M^n_{p_j}$ will depend on the unknown relative depth k_{p_j} at location p_j, this precludes direct estimation of camera motion from PSFs. It should also be noted that there is an inherent ambiguity between translations and relative depth. This is because $(\hat{k}, \hat{T})$ and $(\frac{\hat{k}}{s}, Ts)$ can result in the same shift. Hence, under unknown relative depth, camera motion estimation is ambiguous. In [14] to avoid this problem, an additional step is introduced to ensure that all the patches are extracted from the same depth layer. While this makes sense for the bilayer scenes in [14], it will be impractical to impose such a constraint in our multi-layered scenario. Here, we refrain from restricting all k_{p_j} to have the same value. Instead, we use multiple blurred observations to our advantage by utilizing PSFs at identical locations but from different blurred images to estimate the relative depths at these locations. This ensures that all the individual TSFs will solve for a common relative depth at a given location which helps to improves robustness.

TSF update: We initialize all the k_{p_j}s as unity and construct the matrix $M^n_{p_j}$. By stacking all $h^n_{p_j}$s calculated at different locations of g_n and concatenating the corresponding $M^n_{p_j}$s, we arrive at h^n and $\mathcal{M}^n$, respectively. The TSF w^n corresponding to g_n can be estimated by minimizing the cost function

$$\min_{w^n} \parallel h^n - \mathcal{M}^n w^n \parallel_2^2 + \lambda \parallel w^n \parallel_1, \tag{8}$$

where λ is a regularization parameter. Leveraging the fact that the number of transformations is very few, note that we have imposed a sparsity constraint on w^n [14]. We use the alternating direction method of multipliers (ADMM) algorithm [1] to solve the cost function (8). This yields the TSF for each of the blurred images.

We proceed to model both inter-frame and intra-frame motions within a single camera pose space as both are caused by camera shake. Šroubek et al. [17] accommodate translational shift between the observed uniformly blurred images in the PSF space. We extend this notion to the projective camera motion model by modeling both inter-frame and intra-frame motion within a single camera pose space. Consider two blurred images g_1 and g_2. If $g_1 = \sum_{i=1}^{N} w_i^1 . \mathcal{H}_{\Theta_i^1, T_i^1}(f_1)$ and $g_2 = \sum_{i=1}^{N} w_i^2 . \mathcal{H}_{\Theta_i^2, T_i^2}(f_2)$ such that $f_2 = \mathcal{H}_{\hat{\theta},\hat{t}}(f_1)$ where $(\hat{\theta}, \hat{t})$ are the inter-frame rotation and translation, (Θ^1, T^1) and (Θ^2, T^2) are the two sets of camera poses, and w^1 and w^2 are the corresponding TSFs that caused the blur in g_1 and g_2. Then g_2 can be written as $g_2 = \sum_{i=1}^{N} w_i^2 . \mathcal{H}_{\Theta_i^1+\hat{\theta}, T_i^1+\hat{t}}(f_1)$. Thus, both the camera poses can be accommodated within a common pose space (Θ^1, T^1). When we estimate the TSF for g_2 with respect to f_1, it will include inter-frame motion too. This results in a common reference for all the TSFs. Therefore, each blurred image can be represented by a weighted average of transformed versions of a single latent f. This also helps in inverse rendering the latent image.

Scale factor update: With the camera motion w^n for each image estimated from the previous step, we generate the PSF at a location p_j using Eq. (6) for different values of k_{p_j}. Because we have multiple blurred images, all the kernels generated using the current estimate of the TSFs at location p_j in each image are compared with the originally estimated kernels $h_{p_j}^n$ so as to update k_{p_j}. The update step for the scale factor is given by

$$\min_{k_{p_j}} \sum_{n=1}^{N_b} \parallel h_{p_j}^n - \sum_{i=1}^{N} w_i^n . \mathcal{H}_{\Theta_i, k_{p_j} T_i}(\delta_{p_j}) \parallel_2^2 \quad j = 1, 2, ..., N_p \tag{9}$$

Since all the TSFs should solve for the same values of depth at these N_p locations, the above cost encourages movement towards the actual relative depth value at each of these locations.

Using the updated k_{p_j} values each w^n is then reestimated using (8) and the above sequence of steps is iterated until the k_{p_j}s converge. This is summarized in Algorithm 1.

Algorithm 1 Camera Motion Estimation

Input: Blurred images g_1, g_2,..., g_{N_b}
for each location p_j, $j = 1$ to N_p **do**
  Estimate $h_{p_j}^1, h_{p_j}^2, ..., h_{p_j}^{N_b}$ from $g_{p_j}^1, g_{p_j}^2, ..., g_{p_j}^{N_b}$
end for
Select a reference position p_r
for each image g_n , $n = 1$ to N_b **do**
  Align $h_{p_1}^n, h_{p_2}^n, ..., h_{p_{N_p}}^n$ with respect to $h_{p_r}^n$
end for
Initialize all k_{p_j}s to unity
repeat
  Estimate $w^1, w^2, ..., w^{N_b}$ by solving Eq. (8)
  Update $k_{p_1}, k_{p_2}, ..., k_{p_{N_p}}$ based on $w^1, w^2, ..., w^{N_b}$ using Eq. (9)
until $k_{p_1}, k_{p_2}, ..., k_{p_{N_p}}$ converge
Output: Estimated TSFs $w^1, w^2, ..., w^{N_b}$.

4. Latent Image and Depth Estimation

As each TSF w^n is estimated with respect to a common reference, we can solve for the common latent image f from the observed blurred images by minimizing the following cost function.

$$\sum_{n=1}^{N_b} \parallel \mathcal{K}^n \hat{f} - \hat{g}_n \parallel_2^2 \quad + \quad \lambda_{TV} \parallel \nabla f \parallel_1 \tag{10}$$

where λ_{TV} is a regularization parameter. Here $\hat{f}$ and $\hat{g}_n$ are vector forms of the latent image f and blurred image g_n respectively, while $\mathcal{K}^n$is a large but sparse matrix representing space-variant blurring. Each row of $\mathcal{K}^n$ can be viewed as a blurring operation performed by w^n at the corresponding location in the image. Thus, $\mathcal{K}^n$ depends on the relative depths across the entire image. Note that the relative depth is known only at p_j, $j = 1, \ldots, N_p$ and *not at all points*. Initially, we assume $k = k_0 = 1$ at all the points except the p_js for which the scale is known and estimate f using ADMM [1].

With estimated latent image f, the relative depth k of the scene is recovered using motion blur as a cue [13]. Depth estimation is formulated as a multi-label MRF optimization problem on a regular 4-connected grid where the labels are the discrete relative depth values between 0 and 1. This is solved by the α- expansion graph cut algorithm [3, 10, 2]. Given $\mathcal{V}$ number of pixels and $\mathfrak{e}$ as the set of edges connecting adjacent pixels, to assign a depth label $\hat{k}$ at a location l_i, we minimize the following energy

$$\min_{\hat{k}} \sum_{l_i \in \mathcal{V}} DC_{\hat{k}}(l_i) \quad + \quad \lambda_s \sum_{l_i, l_j \in \mathfrak{e}} SC(\hat{k}, k(l_j)), \tag{11}$$

where $DC_k(l_i)$ is the data cost for assigning label $\hat{k}$ at pixel location l_i, $SC(\hat{k}, k(l_j))$ is the smoothness cost applied and depends on the label $k(l_j)$ assigned to neighboring pixel l_j and λ_s is a regularization parameter. The data cost for assigning a label $\hat{k}$ at a pixel l_i is defined as

$$DC_{\hat{k}}(l_i) = \sum_{n=1}^{N_b} \parallel g_n(l_i) - g_n^{\hat{k}}(l_i) \parallel_2^2, \tag{12}$$

where $g_n^{\hat{k}} = \sum_{i=1}^{N} w_i^n . \mathcal{H}_{\Theta_i, \hat{k} T_i}(f)$ is the blurred image generated using TSF w^n assuming $\hat{k}$ as the relative depth at all points. Here we enforce an edge-aware smoothness

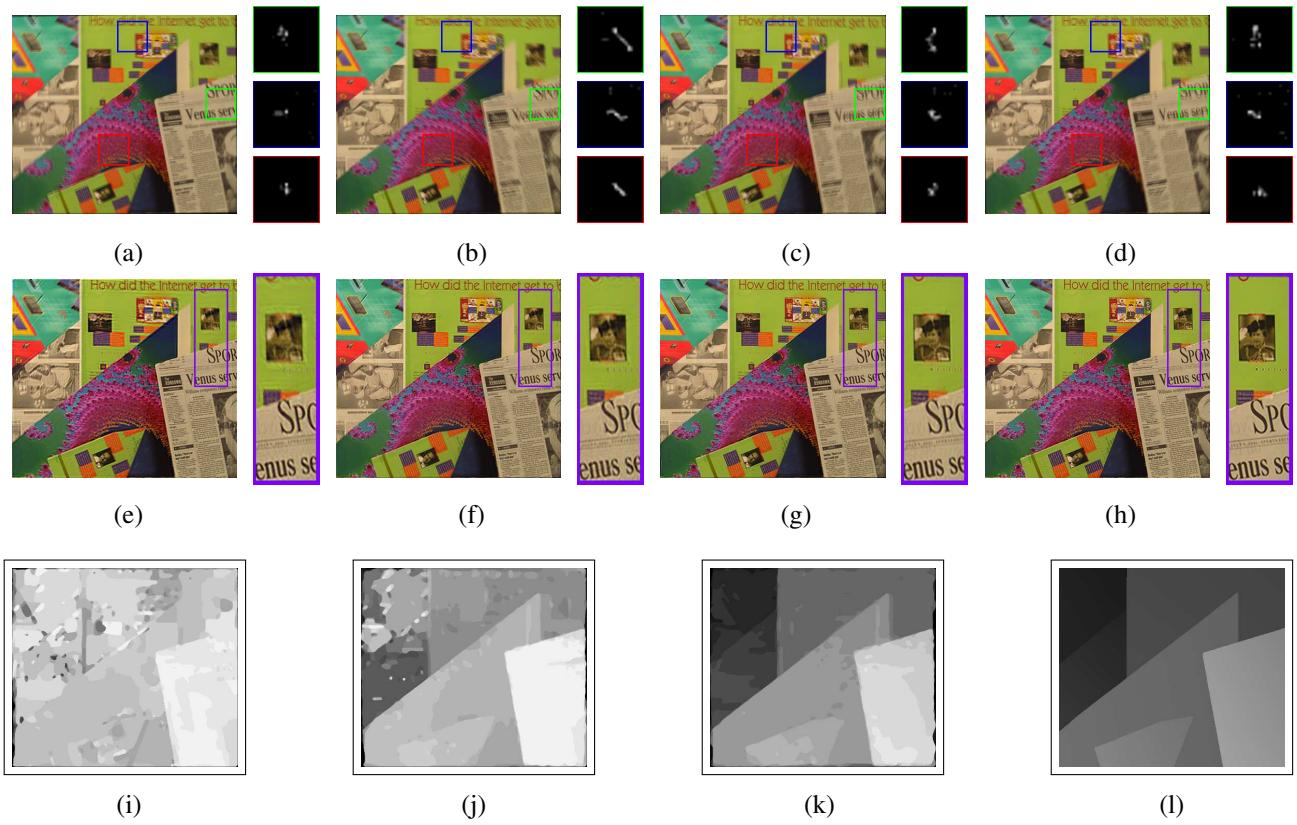


Figure 1. Synthetic experiment: (a)-(d): Generated blurred images with estimated PSFs at three different locations. Estimated latent image and relative depth in (e) and (i): 1^{st} iteration, (f) and (j): 4^{th} iteration, (g) and (k): 10^{th} iteration. (h): Ground truth latent image. (l): True depth map.

cost. i.e., SC is made zero at edge pixels. At other pixels SC is defined as $SC(\hat{k_1}, \hat{k_2}) = |\hat{k_1} - \hat{k_2}|$.

The above steps of depth and latent image estimation are alternatively minimized until convergence. This is summarized in Algorithm 2.

Algorithm 2 Alternate Minimization

1: **Input:** Estimate TSFs $w^1, w^2, .., w^{N_b}$
2: Initialize relative depth as $k = k_0$
3: **repeat**
4: Estimate f by solving Eq. (10)
5: Estimate relative depth k using f
6: **until** k and f converge
7: **Output:** Estimated latent image f and relative depth k.

5. Experimental Results

In this section, we validate our algorithm on both synthetic and real images. The results are also compared with relevant state-of-the-art methods.

5.1. Synthetic Experiments

In the first experiment, we use a clean latent image with a known depth map (Figs. 1 (h) and (l)) to generate the blurred images. The latent image and ground truth depth are taken from http://vision.middlebury.edu/stereo/data/. The synthetic data is generated by blurring the latent image using four realistic TSFs based on the ground truth depth values. Figs. 1 (a)-(d) shows the generated blurred images and extracted PSFs at three different locations (these are marked). The space-variant nature of the blur is clearly evident. Also note that each image has independent blur.

The stages in relative depth estimation at the extracted PSF locations are detailed in Fig. 2. The patches extracted from the regions marked in green colour in Figs. 1 (a)-(d) are shown in first row. We need to estimate the relative depth at these locations. The second row shows the extracted PSFs at these locations across all the four images. Third row contains the PSFs generated from the estimated TSFs using initial depth values. The error between the generated PSFs and the actual PSFs is used to update the relative depths. The generated PSFs using the TSFs estimated

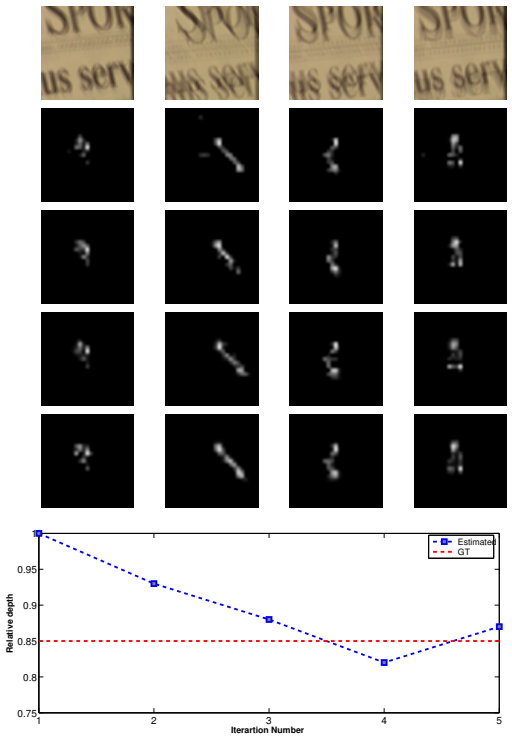


Figure 2. Iterative depth estimation at a point. First row: Blurred patches. Second row: Estimated PSFs for each patch. Third, fourth and fifth rows: PSFs generated using estimated TSF in the 1^{st} , 3^{rd} and 5^{th} iteration respectively. Last row: Plot of depth update versus iteration number.

in the 3^{rd} and 5^{th} iterations are shown in fourth and fifth rows. Fig. 2 also contains a plot of updated relative depth with respect to iterations. It can be seen that the relative depth converges to the actual relative depth value within a few iterations itself. Note that the estimated PSFs and the generated PSFs in the final iteration are quite similar. This shows that the camera poses we estimated are close to the actual camera poses which caused the blur.

The estimated camera poses can also give a rough estimate of the ego motion. Even though the temporal information about the intra-frame motion is not available, knowing the order of capture one can fit a trajectory through the centroids of the set of camera poses, that caused the blur in each image. Such a plot is shown in Fig. 3. Camera poses corresponding to a blurred image are shown in one color. The size of a point indicates the weight of that camera pose. The red points represent the centroids of the camera poses for each image. The time-stamped trail of the centroids (frame number indicated next to the centroids in the plot) reveals the ego motion of the camera. We verified that this is close to the ground truth centroids.

The last two rows of Fig. 1 depict the results of Algorithm 2. The estimated latent image (Figs. 1 (e)-(g)) and

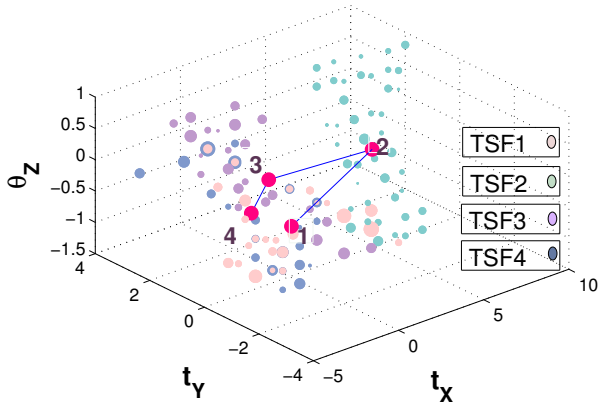


Figure 3. Ego motion: The camera poses corresponding to each of the blurred images are shown in same color. Red dot is the centroid of individual camera poses and blue line indicate the rough motion trajectory.

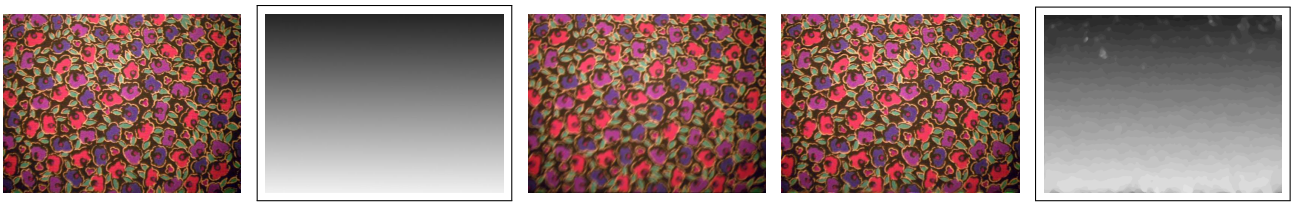
the relative depth (Figs. 1 (i)-(k)) at different stages of iteration are shown. The zoomed-in patches corresponding to each blurred image are also displayed to showcase the improvement over iterations. The depth estimates also improve greatly with iterations. The final estimate of the latent image and scene depth are quite close to the ground truth.

In our second synthetic experiment, we consider a smoothly varying depth scenario such as an inclined plane. The ground truth latent image and depth are shown in Fig. 4(a) and Fig. 4(b). Using four realistic TSFs we blur the latent image considering the continuously varying depth. One such generated image is shown in Fig. 4(c). The estimated latent image and depth are displayed in Fig. 4(d) and Fig. 4(e). Since we use discrete labels in MRF optimization by graph cut, the estimated depth contains discrete layers. But it should be noted that the continuously varying nature of the depth is very well captured by our algorithm unlike [8].

5.2. Real Experiments

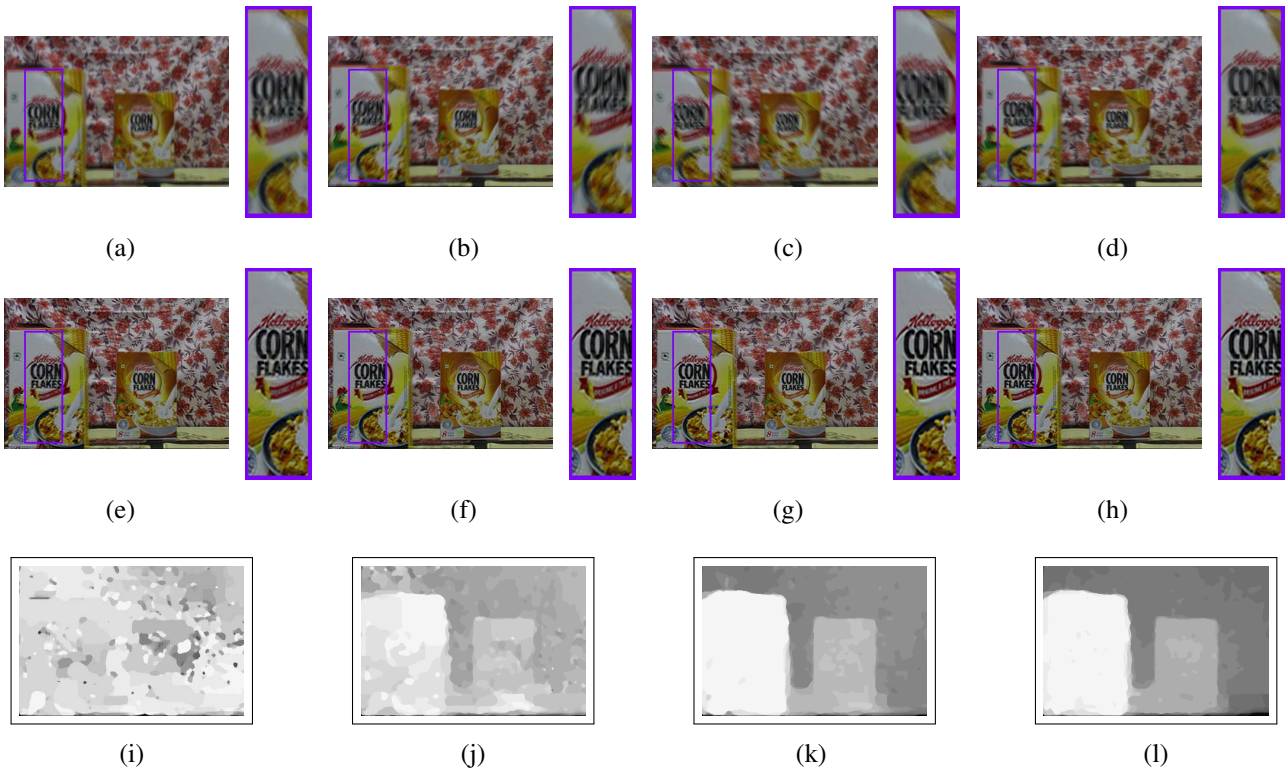
The real experiments are carried out using images captured with Moto G2 and Google Nexus4. In the first example we consider a scene with two boxes on a table. The second box is kept approximately mid-way between the first box and the background. Captured results are shown in Fig. 5 (a)-(d) . It can be seen that the images are mis-aligned due to the handshake and this effect is clearly visible in the zoomed-in patchs. Figs. 5 (e)-(h) and Figs. 5 (i)-(l) are the estimated latent image and depth over iterations. Note that the three different depth levels are clearly distinguishable in the 10^{th} iteration.

For purpose of comparison, we applied the single image non-uniform motion blur methods of [19] and [21] to individual images. Since the codes of [8] and [11] were not available, we could not perform comparison with those



(a) (b) (c) (d) (e)

Figure 4. Synthetic experiment: (a) Ground truth latent image. (b) Ground truth depth. (c) A blurred image from the set. (d) Estimated latent image. (e) Estimated depth.



(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

Figure 5. Real experiment: (a)-(d): Captured blurred images with zoomed-in locations marked. Estimated latent image and relative depth in (e) and (i): 1^{st} iteration, (f) and (j): 2^{nd} iteration, (g) and (k): 5^{th} iteration, (h) and (l): 10^{th} iteration.

methods. The best deblurred image is selected and compared with our latent image estimate. Fig. 6 contains these results. The deblurring technique of [19] is independent of depth and their result (Fig. 6(b)) is not satisfactory. The result of [21] is shown in Fig. 6(c). Note that the estimated latent image of both methods have blur residuals. We submit that this is not a fair comparison as these methods are based on single observation.

In our second real experiment, we consider top-view of a bricked pavement. The two sun shades over the windows and the pavement are at different depths. The captured blurred observations are shown in Figs. 7 (a)-(c). The final recovered latent image and depth are shown in Fig. 7 (d) and Fig. 7(e), respectively. Few errors in the depth map estimation are mainly due to lack of texture at some regions. However, note that the three main depth layers are well differentiated in the estimated depth map. We compare our deblurred results with [19] and [21] in Figs. 7 (f)-(k). It can be observed that the deblurred result of [19] has ringing artifacts while [21] fails to preserve fine details. On the other hand, our method produces a sharp and artifact-free deblurred result.

6. Conclusions

In this work, we addressed the problem of recovering the latent image, scene depth and camera motion from a set of mis-aligned and space-variantly blurred images. We proposed a method to estimate the camera motion with respect

Figure 6. Comparisons: (a) A blurred image from the set. (b) Deblurring by Whyte et.al [19] and zoomed-in patch. (c) Deblurring by Xu et.al [21] and zoomed-in patch. (d) Deblurring by our algorithm and zoomed-in patch.

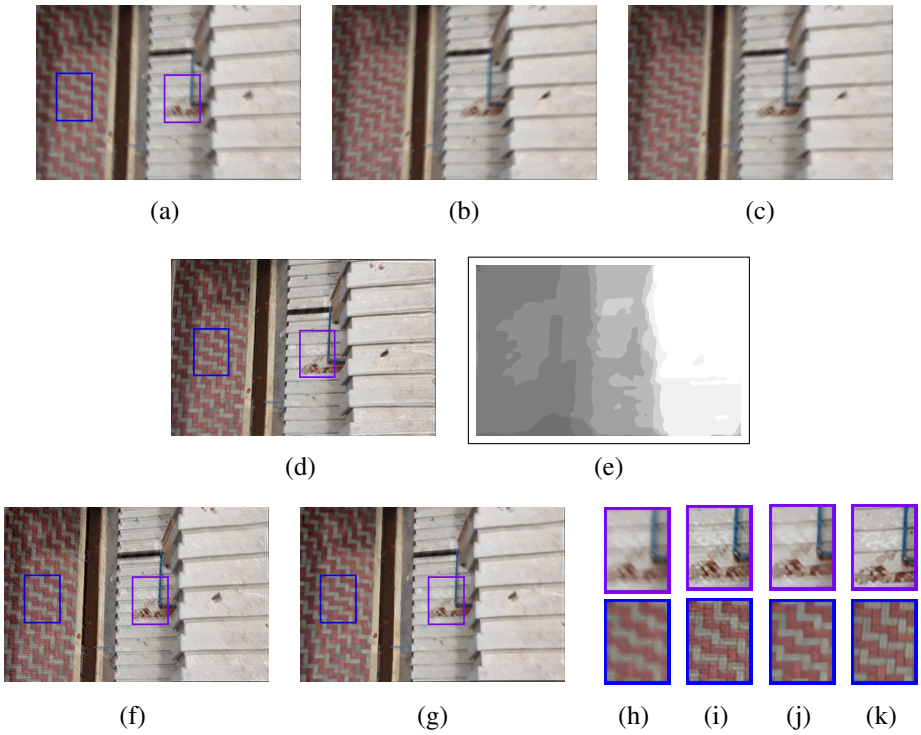


Figure 7. Real experiment: (a)-(c): Captured blurred images. (d) Estimated latent image. (e) Estimated depth. (f) Deblurring by Whyte et.al [19]. (g) Deblurring by Xu et.al [21]. (h), (i), (j) and (k): Zoomed-in patches from (a), (f), (g) and (d), respectively.

to a common pose space which enabled us to effectively model both inter-frame and intra-frame motion. Based on the estimated camera motion, the latent image and depth were recovered using an alternate minimization framework. The effectiveness of our method was demonstrated on both synthetic and real examples.

Acknowledgments

A part of this work was supported by a grant from the Asian Office of Aerospace Research and Development, AOARD/AFOSR. The support is gratefully acknowledged. The results and interpretations presented in this paper are that of the authors, and do not necessarily reflect the views or priorities of the sponsor, or the US Air Force Research Laboratory.

References

[1] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011. 3, 4

[2] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(9):1124–1137, 2004. 4

[3] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(11):1222–1239, 2001. 4

[4] S. Cho, H. Cho, Y.-W. Tai, and S. Lee. Non-uniform motion deblurring for camera shakes using image registration. In

ACM SIGGRAPH 2011 Talks, page 62. ACM, 2011. 1

[5] S. Cho, Y. Matsushita, and S. Lee. Removing non-uniform motion blur from images. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007. 2

[6] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman. Removing camera shake from a single photograph. *ACM Transactions on Graphics (TOG)*, 25(3):787–794, 2006. 1

[7] A. Gupta, N. Joshi, L. Zitnick, M. Cohen, and B. Curless. Single image deblurring using motion density functions. In *ECCV '10: Proceedings of the 10th European Conference on Computer Vision*, 2010. 1, 2

[8] Z. Hu, L. Xu, and M.-H. Yang. Joint depth estimation and camera shake removal from single blurry image. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 2893–2900. IEEE, 2014. 2, 6

[9] A. Ito, A. C. Sankaranarayanan, A. Veeraraghavan, and R. G. Baraniuk. Blurburst: Removing blur due to camera shake using multiple images. *ACM Trans. Graph., Submitted*, 3. 1

[10] V. Kolmogorov and R. Zabin. What energy functions can be minimized via graph cuts? *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(2):147–159, 2004. 4

[11] H. S. Lee and K. M. Lee. Dense 3d reconstruction from severely blurred images using a single moving camera. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 273–280. IEEE, 2013. 2, 6

[12] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Understanding and evaluating blind deconvolution algorithms. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1964–1971. IEEE, 2009. 1, 2

[13] C. Paramanand and A. Rajagopalan. Shape from sharp and motion-blurred image pair. *International journal of computer vision*, 107(3):272–292, 2014. 2, 3, 4

[14] C. Paramanand and A. N. Rajagopalan. Non-uniform motion deblurring for bilayer scenes. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1115–1122. IEEE, 2013. 1, 2, 3

[15] Q. Shan, J. Jia, and A. Agarwala. High-quality motion deblurring from a single image. In *ACM Transactions on Graphics (TOG)*, volume 27, page 73. ACM, 2008. 1

[16] F. Šroubek and J. Flusser. Multichannel blind iterative image restoration. *Image Processing, IEEE Transactions on*, 12(9):1094–1106, 2003. 1, 3

[17] F. Şroubek and J. Flusser. Multichannel blind deconvolution of spatially misaligned images. *Image Processing, IEEE Transactions on*, 14(7):874–883, 2005. 4

[18] Y.-W. Tai, P. Tan, and M. S. Brown. Richardson-lucy deblurring for scenes under a projective motion path. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(8):1603–1618, 2011. 1

[19] O. Whyte, J. Sivic, A. Zisserman, and J. Ponce. Non-uniform deblurring for shaken images. *International journal of computer vision*, 98(2):168–186, 2012. 1, 2, 6, 7, 8

[20] L. Xu and J. Jia. Two-phase kernel estimation for robust motion deblurring. In *Computer Vision–ECCV 2010*, pages 157–170. Springer, 2010. 1

[21] L. Xu, S. Zheng, and J. Jia. Unnatural l0 sparse representation for natural image deblurring. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1107–1114. IEEE, 2013. 6, 7, 8

[22] H. Zhang and L. Carin. Multi-shot imaging: Joint alignment, deblurring, and resolution-enhancement. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 2925–2932. IEEE, 2014. 1

[23] H. Zhang, D. Wipf, and Y. Zhang. Multi-image blind deblurring using a coupled adaptive sparse prior. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1051–1058. IEEE, 2013. 1, 2