Accurate Motion Deblurring using Camera Motion Tracking and Scene Depth

Hyeoungho Bae, Charless C. Fowlkes, Pai H. Chou
University of California,Irvine
hyeoungb@uci.edu, fowlkes@ics.uci.edu, phchou@uci.edu

Abstract

In this paper, we propose an estimation algorithm for spatially-variant blur due to camera motion. To estimate the most accurate latent image, we integrated depth sensor (Microsoft Kinect) and IMU sensor with the camera. The joint analysis of the blurry image, IMU data and the depth data provide better recovery of the real camera motion during the course of the exposure. The reconstructed camera trajectory along with the depth map is then used to synthesize a spatially-variant blur kernel to estimate the final latent (non-blurry) image. The results show that our algorithm effectively compensates the motion blur from the original image while taking scene geometry into account.

1. Introduction

Motion deblurring is one of the most interesting subject in the computer vision field. Motion deblurring can be classified into two different categories: blind and non-blind deblurring. Non-blind deblurring attempts to remove the effect of blurring when the motion or blur kernel is known. Blind deblurring estimates both the motion and the latent image simultaneously. Estimating two unknowns from a single blurry photograph is an ill-posed problem, which requires strong prior information to estimate accurate results.

A simple model for blur is to assume that the a blurry image B is generated by convolution of a true image I with a spatially-invariant kernel k and corrupted with some noise.

$$B = I * k + n \tag{1}$$

One common approach in solving the eqation(1) where the blur kernel is spatially-invariant is to use some prior to regularize the joint estimate of the latent image (I) and blur kernel (k) that explain the blurry image (B) [3, 4, 6, 8, 10–12, 14]. However, spatially-invariant assumption is not valid in most of the cases. Translational motion induces different pixel movement for points which are different distances from the camera. Camera rotation can even result in pixels moving in different directions in the same image. To model the phenomenon, one can use spatially-variant blur kernel estimation [2, 5, 7, 9, 13, 15]. In this case, the blur kernel is unique for each image location. To estimate spatially varying kernels, it is natural to introduce a prior at the level of the camera motion [9], incorporate scene depth information [15] and utilize measurements of camera motion to constrain the kernel more effectively [5, 7, 13].

In this paper, we suggest a novel motion deblurring system and algorithm to estimate spatially-variant blur kernel effectively. Our system estimates the camera motion during the exposure time using the data from the IMU (Inertia Measurement Unit) and scene geometry from a structured-illumination depth sensor (Microsoft Kinect). The reconstructed camera motion is used to build the individual blur kernels for each pixel in the image area. Given this data provided by additional sensors, we are able to reconstruct spatially varying blur kernels for large motions with very a very general camera motion model.

Figure 1. The system picture: The system is composed of DSLR camera (Canon EOS450d with 18-35mm lens), IMU (Analog devices, ADIS16350), Depth sensor (Microsoft Kinect), and Control board (Beagleboard-xm)

1.1. Related Works

The closest work to ours is that of Joshi et al., who suggested an IMU based spatially-variant blur kernel estimation algorithm [7]. They integrated an IMU with a camera like our system. However, they assumed the depth is constant over the scene. Most of the cases, the depth is not constant over the image area. Also, they used a naïve assumption on the direction of the gravity, which assumes the direction of the gravity is same as the direction of the averaged force (or acceleration) over the camera integration time. They defined a bound for the accelerometer drift (a

few milimeters) and minimized the energy function to find accelerometer data to get the end point. However, we used longer exposure time (0.8 - 1.7sec) and the drift error from the raw data is larger than their (more than a few centimeters). In that case, finding accelerometer data, end points, and the latent image that minimizes their equation is quite difficult. Gupta et al. developed a spatially-variant blur kernel estimation algorithm based on the work of Shan et al. [5, 12]. They divide the image area to get small patches and assume the blur kernel is spatially-invariant over the patch area. Shan et al.'s algorithm is used to estimate the blur kernel of the area. Based on the kernels, they reconstruct the camera motion that can explain the blur kernels. However, they also assumed the depth is constant over the scene. Even though RANSAC algorithm is used for filtering outlier among the estimated blur kernels, their algorithm is dependent of the performance of Shan's algorithm. Whyte et al. suggested a non-uniform motion deblurring algorithm [13]. They reconstructed the camera motion to estimate the spatially-variant blur kernel. However, due to the limited information available from only the blurry image, they restricted themselves to estimating rotational motion of the camera. Krishnan et al. suggested a limited version of spatially-variant blur kernel estimation algorithm [9]. Their algorithm can estimated 2D (or 'in-plane') and 3D rotation (no translation). Like other spatially-invariant blur kernel, their algorithm is based on the prior knowledge of the scene (l_1/l_2 regularization). Due to the simplicity of such priors, accurate motion deblurring of real-world image is limited with their approach. Xu et al. suggested a depth-aware motion deblurring algorithm [15]. They used a stereo camera to build a disparity map and applied spatially-invariant blur kernel estimation to areas with the same depth. Even thuogh they used depth information for their deblurring algorithm, their approach can only handle translational motion. Their approach has limit like Krishnan's approach since their algorithm is dependent on the prior assumption like their previous spatially-invariant motion deblurring algorithm [14]. Cho et al. used multiple sets of images to reconstruct the camera motion and estimated blur kernels [1]. Their approach needs multiple sets of images which are expensive to acquire in low-light settings and their algorithm depends on the accuracy of registration of multiple sets of images.

2. System architecture

2.1. System Overview

Our system is composed of a DSLR Camera, an IMU (Inertial Measurement Unit, Analog Devices ADIS16350), a depth sensor (Microsoft Kinect) and a control board. The camera is Canon eos450d with 18-55mm zoom lens. For our experiment, we fixed the focal distance to 18mm, which is the closest available focal length to that of the depth sensor. The IMU has a tri-axis accelerometer and a tri-axis gyroscope. The accelerometer can measure up to $10g$ and the bandwidth of the IMU is $350Hz$. The control board is Beagleboard-xm, which has OMAP3530 as the CPU. THe IMU is connected to the SPI interface and the camera shutter I/O is connected to the GPIO port. To interface 3.3V devices, an externsion board is used. A linux operating system (ubuntu 10.3 rev.) is used for the board.

2.2. Basic Operation

Our algorithm is implemented using Matlab in a laptop. The matlab script controls Kinect (USB I/F) and the control board (Serial I/F). The depth map is taken while an image is taken from the DSLR camera. The control board takes care of synchronization of the IMU data acquisition and the camera shutter control. The sampling period of the IMU for whole group(3 accelerometers and 3 gyroscopes) is 2ms. The DSLR+Kinect mount shown in Fig. 1 is built using the schematic distributed by RGBDToolkit [1]. Since the resolution of the depth map is smaller than that of DLSR image, we assume that the depth of a point is that of the nearest point. On the control board, we developed device drivers for the IMU and the camera shutter. Those binaries are called by a simple linux script.

3. Algorithm

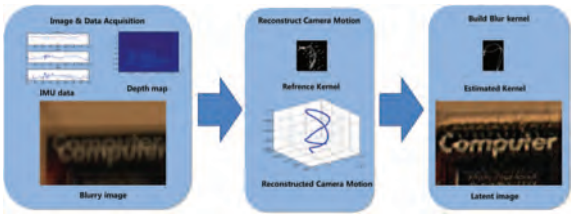


Figure 2. Algorithm overview: The depth map and the IMU data are acquired during the shutter exposure time. After compensating the gravity and the drift error, the spatially-variant blur kernel can be estimated using the depth map and the camera motion.

The motion data is gathered using two different ways in our system: 1) Direct camera motion sensing from IMU (Inertial Measurement Unit), and 2) Motion estimation from image deblurring. To reconstruct the camera motion out of those data, the depth map measured by Kinect is used.

3.1. Kinematics

We have the acceleration data, ${a_i}^t$ and the angular velocity from the gyroscope, ${\omega_i}^t$ (i means the i-th data among the data output from the IMU and t means the current frame of the camera, whereas 0 means the initial frame of the camera

[1] http://rgbdtoolkit.com/hangar.html

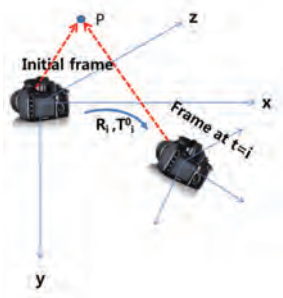


Figure 3. Basic kinematics: To estimate the blur kernel, 3D homography is used to reflect the 6-DoF camera motion to the pixel (P in the figure).

(Fig. 3). The displacement and the rotation of the IMU can be expressed using the below equations:

$$T_i^0 = T_{i-1}^0 + v_{i-1}^0 \Delta t + \frac{1}{2} a_i^0 \Delta t^2, \tag{2}$$

where $a_i^0 = R_{i-1}^T a_i^t$

$$R_i = R\left(\theta_{i-1}^0 + R_{i-1}^T \omega_i^t \Delta t\right), \tag{3}$$

where $R(\theta)$ means the 3×3 rotation matrix.

Using the above two equations, the relative displacement of the point, P in Fig. 3 at time i can be calculated like this:

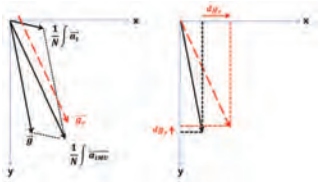
$$P_i = {R_{i-1}}^T P_0 - {T_i}^0, \tag{4}$$

where P_0 is the pixel coordinate in the initial frame at $t = 0$. Once we know the relative position of the pixel at any time during the camera exposure,then we can construct the trajectory of the pixel or blur kernel using simple homography relation.

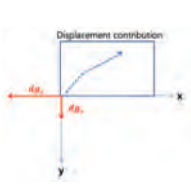
3.2. Histogram Analysis for Gravity and Drift Compensation

IMU is useful to track the motion of an object during a relatively short period time (less than a minutes). To estimate the relative position of an object under tracking, one should double integrate the acceleration from the device, which amplifies the noise included in the data (generally called drift error). Another problem of IMU is gravity. Generally, it is not feasible to distinguish the gravity from the inertia force that we want to measure unless we can detect a stationary moment, which means no-external acceleration except for the gravity. Even though we can detect such moment, due to the drift-error, it is difficult to define the direction of the gravity during the motion.

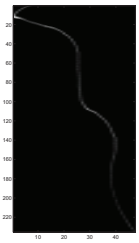
In [7], Joshi et al. used an IMU to estimate the camera motion. They assumed that the direction of the gravity is the direction of the averaged acceleration during the camera exposure time, which is not realistic assumption (See Fig. 4(c)). To compensate the drift error, they made an assumption on the bound of the accelerometer drift during the camera exposure time. The longest exposure time is 0.5sec



(a) Naïve Graviity assumption



(b) Contribution of the gravity assumption

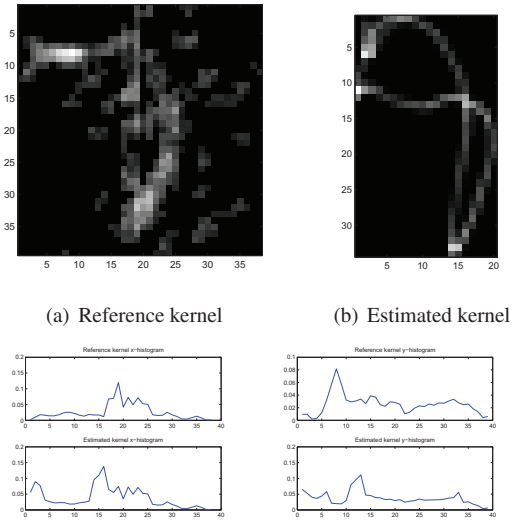


(c) Estimated kernel

Figure 4. The effect of the naïve Gravity assumption: Due to various reason, the direction of the gravity is not aligned with the simple averaged acceleration direction ($\frac{1}{N}\int \vec{a}_{imu}$).

and the order of drift is a few milimeters. Under those assumptions, they built an energy function to find a latent image, the end point, and the depth (a single depth over the scene) that minimize it. In our case, the exposure time is not bounded. During our experiment the exposure time range from 0.8sec to 1.7sec. The drift of the accelerometer during the camera exposure time is more than a few centimeters. Thus, we have at least 10^6 times larger space to explore (we don't assume that the depth is constant) even though we assume that the sampling frequency of the accelerometer is same. Instead, we suggest much simpler way of compensating the gravity and the drift error simultaneously.

Like Joshi et al., we assume that the displacement contribution due to the accelerometer drift is linear in time. Therefore, the displacement due to invalid gravity estimation and the drift can be regarded as single variable per axis ($\mathrm{d}x$ and $\mathrm{d}y$ in Fig. 4(b)). The drift error of the gyroscope is assumed to be negligible during the camera exposure time. To compensate the effect of gravity and the drift error, we referenced a blur kernel estimation from the image area around the point (Fig. 5(a)) that we want to track using a spatially-invariant blur kernel estimation algorithm. This approach may seem similar to [5]. However, we are bounded by the IMU data to construct the camera motion and the compensation factors (dx, dy, dz) are constant over the time, which means the influence of the outlier to our



(a) Reference kernel (b) Estimated kernel

(c) X-axis histrogram (d) Y-axis histrogram

Figure 5. Reference kernel based gravity compensation method: We use a reference kernel estimated from the patch-mosaic algorithm in (a). The histogram analysis is used to find the similarity between two kernels.

estimation results is much lower than that of [5].

$$[dx, dy, dz] = \arg\min_{dx,dy,dz} Corr(hist(k_{ref}) - hist(k_{imu})), \tag{5}$$

where dx, dy, and dz represent the gravity and drift components included in the accelerometer data. $Corr(A, B)$ means the function calculates the correlation between two different histograms A and B to show the similarity. (Fig. 5(c) and 5(d)). The equation (5) can be used for multiple image points to increase the accuracy of the compensation factors.

3.3. Latent Image Estimation

Once the camera motion is reconstructed from the previous step, we can build the spatially-variant blur kernel for the entire image area using the compensated IMU data and the depth map. The latent image can be estimated by minimizing the following equation:

$$I = \arg\min_{I} ||B - KI||^2 + \lambda ||I||^2, \tag{6}$$

where I, and B is the column vectorized $m \times n$ latent and the original blurry image, respectively. K is $mn \times mn$ spatially-variant blur kernel.

4. Results

To evaluate the preformance of our deblurring algorithm, we deblurred a real world blurry image took with our system (Fig. 6. In the original blurry image, three different areas of different depth and coordinate are selected (dotted rectangle box in Fig. 6(a)). The estimated blur kernels using our algorithm are shown beside each box. Our estimated latent images for each area is shown in the second column (Fig. 6(c), Fig. 6(h), and Fig. 6(m)). To compare the performance of our algorithm without the depth map, we showed the latent image estimated from the blur kernel with constant depth assumption (the depth was the average depth of the scene): Constant depth column. (Fig. 6(d), Fig. 6(i), and Fig. 6(n). More detailed comparison are shown in Fig. 8.

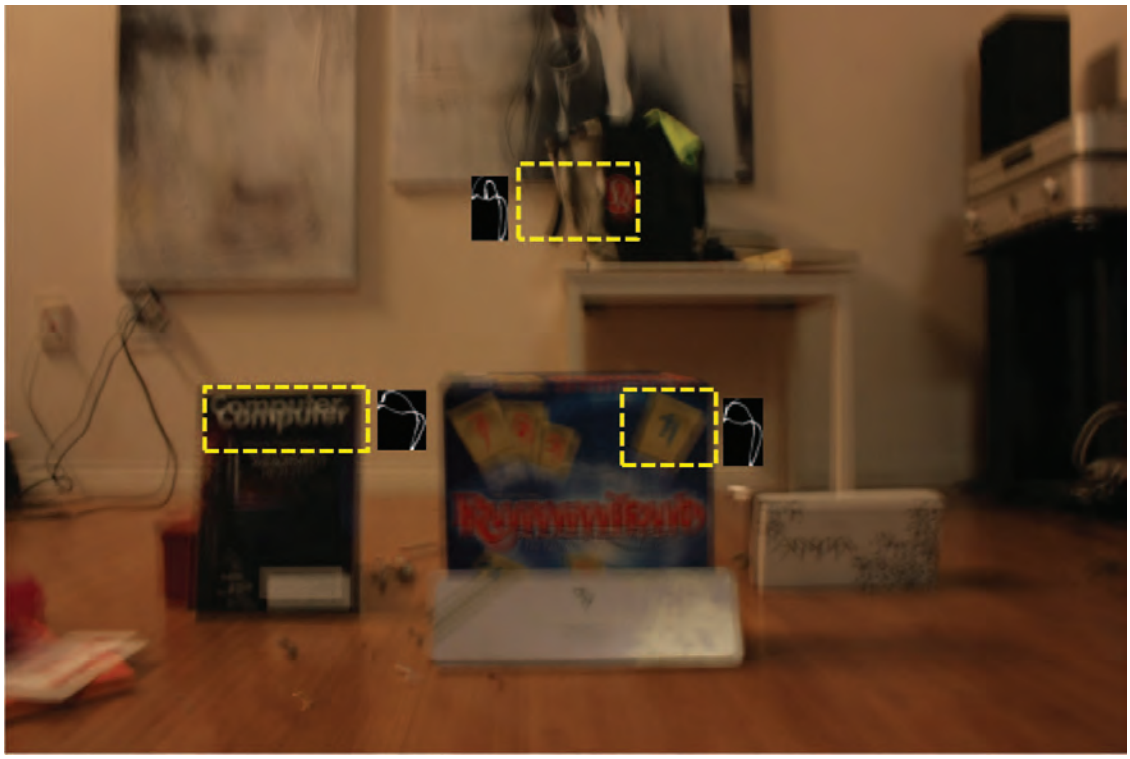
Our algorithm shows the most accurate estimation among the comparison group. The size and the shape of the blur kernel is different from the location and the depth. Our algorithm successfully reconstruct the camera motion and estimate the blur kernels from the data. Spatially-invariant assumption is definitely not valid for this case. The results from from Xu et al.'s algorithm and Fergus et al.'s algorithm ([4, 14], respectively) shows little improvement compared to the original image. The deblurring results for different type of motion are shown in Fig. 9.

The difference of the blur kernels of Fig. 6 is described more clearly in Fig. 7. We deblurred the entire image area using each blur kernel and showed the results in the figure. In Fig. 7(a), the blur kernel of the center box (of Fig. 6(a)) is used to deblur the entire image. Each area of the result is provided in the figure. Each blur kernel is dedicated to its pixel coordinate and can't accurately deblur other region.

Since we couldn't compare our algorithm with other spatially-variant blur kernel algorithms, we compare our algorithm in two different ways: with depth map (our original configuration) and without depthmap (or constant depth assumption like [5, 7]). For the closely placed objects like Fig. 8(a) or Fig. 8(b), the influence of the depth information is more clear than relatively far placed object like Fig. 8(c). For example, the shape of the number '6' of the first row in Fig. 8(a) is more accurate than that of the second row. In Fig. 8(b), the length of the blur kernel of the second row is shorter than the first row, which causes blurry shade beneath 'computer' letters.

5. Conclusion

In this paper, we developed a system to reconstruct 6-DoF camera motion during the exposure time aided by the inertial measurement unit and the depth sensor. The analysis on the IMU data and the blurry image combined with the depth map gives accurate estimation on the camera motion. We developed a novel way of compensating the gravity and drift error contained in the accelerometer of the IMU. The results show that our algorithm can estimate the latent image accurately. With the scene depth, we can reduce the limit of DoF in camera motion, which can increase the estimation accuracy compared to other algorithms.



(a) Original blurry image

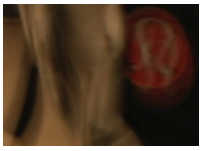
(b) Original zoom-center

(c) Our zoom-center

(d) Constant depth-center

(e) Xu zoom-center

(f) Fergus zoom-center



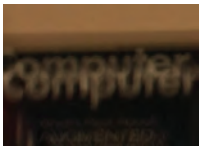
(g) Original zoom-up

(h) Our zoom-up

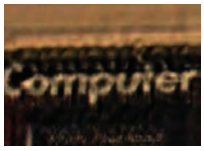
(i) Constant depth-up

(j) Xu zoom-up

(k) Fergus zoom-up



(l) Original zoom-left



(m) Our zoom-left

(n) Constant depth-left

(o) Xu zoom-left

(p) Fergus zoom-left

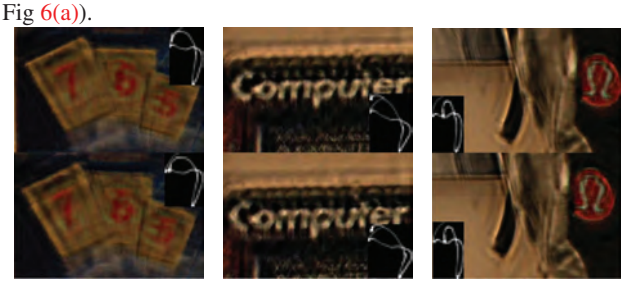
Figure 6. The estimated latent images: To evaluate the performance of our algorithm, we compared the estimated latent image with the patch-mosaic deblurring algorithm, Xu et al's algorithm [14] and Fergus et al,'s algrotihm [4]. The spatially-variant blur kernels estimated using our algorithm are shown by the selected area in (a). The latent images from our algorithm of the selected areas are provided in the second column - (c), (h), and (m).

Since our algorithm needs reference for the histogram analysis, the estimation accuracy is somewhat related to the accuracy of the reference blur kernel, which is not strongly dependent on the kernel like [5]. To address this issue, one can think about extra hardware to estimate the gravity direction like an inclinometer sensor.

Acknowledgement. This work was sponsored by the National Science Foundation grant CBET-0933694 and Air Force Office of Scientific Research grant FA9550-10-1-0538. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National

(a) Center Kernel (b) Left Kernel (c) Up Kernel

Figure 7. Spatially-variant blur kernel: Single blur kernel is used to get the result of each column. Each column contains three image patches from different image locations (square boxed area of Fig 6(a)).



(a) Depth-center (b) Depth-left (c) Depth-up

Figure 8. Blur kernel with the depth map and without the depth map: The first row of the images are the latent image from the blur kernel estimated using IMU data and the depth map. The second row are the latent image from the IMU data and constant depth for all three image patches. The estimated blur kernels are shown in each image.

Science Foundation.

References

[1] S. Cho, H. Cho, Y. Tai, and S. Lee. Registration based non-uniform motion deblurring. *Computer Graphics Forum (Special issue on Pacific Graphics 2012*, 2012. 2

[2] S. Cho, Y. Matsushita, and S. Lee. Removing non-uniform motion blur from images. *ICCV*, 2007. 1

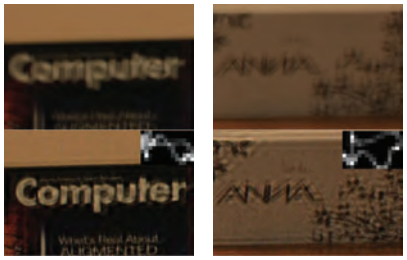
[3] T. Cho. Motion blur removal from photographs. *M.I.T Ph.D dissertation*, 2010. 1

[4] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman. Removing camera shake from a single photograph. *ACM Transactions on Graphics*, 25(3), 2006. 1, 4, 5

[5] 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, 3, 4, 5

[6] J. Jia. Single image motion deblurring using transparency. *CVPR*, 2007. 1

(a) Original blurry image



(b) Left box area (c) Right box area

Figure 9. Another deblurring result: The blur kernels for two different region in the image (a) are estimated. Top row of (b) and (c) contains the original image and bottom shows the estimated image and blur kernel.

[7] N. Joshi, S. B. Kang, C. L. Zitnick, and R. Szeliski. Image deblurring using inertial measurement sensors. *ACM Transactions on Graphics*, 29(4), 2010. 1, 3, 4

[8] N. Joshi, R. Szeliski, and D. Kriegman. Psf estimation using sharp edge prediction. *CVPR*, 2008. 1

[9] D. Krishnan, T. Tay, and R. Fergus. Blind deconvolution using a normalized sparsity measure. *CVPR*, 2011. 1, 2

[10] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Understanding and evaluating blind deconvolution algorithms. *CVPR*, 2009. 1

[11] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Efficient marginal likelihood optimization in blind deconvolution. *CVPR*, 2011. 1

[12] Q. Shan, J. Jia, and A. Agarwala. High-quality motion deblurring from a single image. *ACM Transactions on Graphics*, 27(3), 2008. 1, 2

[13] O. Whyte, J. Sivic, A. Ziseerman, and J. Ponce. Non-uniform deblurring for shaken images. *CVPR*, 2010. 1, 2

[14] L. Xu and J. Jia. Two-phase kernel estimation for robust motion deblurring. *ECCV*, pages 157–170, 2010. 1, 2, 4, 5

[15] L. Xu and J. Jia. Depth-aware motion deblurring. *ICCP*, 2012. 1, 2