Single Image Deblurring Using Motion Density Functions

Ankit Gupta[1], Neel Joshi[2], C. Lawrence Zitnick[2], Michael Cohen[2], and Brian Curless[1]

[1] University of Washington
[2] Microsoft Research

Abstract. We present a novel single image deblurring method to estimate spatially non-uniform blur that results from camera shake. We use existing spatially invariant deconvolution methods in a local and robust way to compute initial estimates of the latent image. The camera motion is represented as a *Motion Density Function* (MDF) which records the fraction of time spent in each discretized portion of the space of all possible camera poses. Spatially varying blur kernels are derived directly from the MDF. We show that 6D camera motion is well approximated by 3 degrees of motion (in-plane translation and rotation) and analyze the scope of this approximation. We present results on both synthetic and captured data. Our system out-performs current approaches which make the assumption of spatially invariant blur.

1 Introduction

Image blur due to camera shake is a common problem in consumer-level photography. It arises when a long exposure is required and the camera is not held still. As the camera moves, the image formation process integrates a stream of photographs of the scene taken from slightly different viewpoints.

Removing blur due to camera shake is currently a very active area of research. Given only a single photograph, this blur removal is known as blind deconvolution, i.e., simultaneously recovering both the blur kernel and the deblurred, latent image. Commonly, it is assumed that the blur kernel is spatially invariant, reducing the set of camera motions that may be modeled.

An open problem is to model more general camera motions, which are quite common and can cause spatially varying blur. We focus on generalizing the camera motion to include both 2D translation and in-plane rotation. Thus, starting from a single image, we seek to recover the latent image, and the spatially varying blur kernels that arise from this more general camera motion.

We develop a novel formulation of the camera shake deblurring problem by generalizing spatially invariant (2D) kernels. Although a full model of motion would require 6 degrees of freedom, we show that for typical scenarios, 6D general motion can be reasonably approximated with a 3-dimensional motion (only in-plane rotation and translation). The problem is still substantially more under-constrained than the standard in-plane translation-only case.

K. Daniilidis, P. Maragos, N. Paragios (Eds.): ECCV 2010, Part I, LNCS 6311, pp. 171–184, 2010.
© Springer-Verlag Berlin Heidelberg 2010

Rather than directly recovering the spatially varying blur kernels at each image point, we observe that camera motion is a 1D curve through camera pose space. We model the time spent in each pose over the exposure as a density function in a higher dimensional camera motion space; we call this a *Motion Density Function* (MDF). The MDF can be used to generate the kernel at any location in the image without knowing the temporal ordering of the motion curve. Our system takes as input (1) a blurred image, (2) its EXIF tags specifying sensor resolution and approximate focal length, and (3) an estimate of the maximum blur kernel size, and recovers both the latent image and the MDF using a non-linear optimization scheme similar to a more traditional spatially invariant blind-deconvolution method. Altogether, we demonstrate an automatic method for single image deblurring under a range of spatially-varying, camera motion blurs.

The paper is organized as follows. In Section 2, we survey related work. In Sections 3 and 4, we propose and analyze our optimization formulation and then discuss our solution of this formulation in Section 5. In Section 6, we show the results of our approach and finally conclude with a discussion of limitations and future work in Section 7.

2 Related Work

Image deblurring has received a lot of attention in the computer vision community. Deblurring is the combination of two tightly coupled sub-problems: PSF estimation and non-blind image deconvolution. These problems have been addressed both independently and jointly [1]. Both are longstanding problems in computer graphics, computer vision, and image processing.

Image blur arises from multiple causes. Image blur due to camera motion has recently received increased attention, as it is a very common problem in consumer-level photography. In most recent work, image blur is modeled as the convolution of an unobserved latent image with a single, spatially invariant blur kernel [1,2,3,4,5,6,7,8,9,10,11].

Software-based methods use image priors and kernel priors to constrain an optimization for the blur kernel and the latent image [2,3,4,5,6,12,13,14].

Fergus et al. [4] recover a blur kernel by using a natural image prior on image gradients in a variational Bayes framework. Shan et al. [2] incorporate spatial parameters to enforce natural image statistics using a local ringing suppression step. Jia et al. [13] use transparency maps to get cues for object motion to recover blur kernels by performing blind-deconvolution on the alpha matte, with a prior on the alpha-matte. Joshi et al. [14] predict a sharp image that is consistent with an observed blurred image. They then solve for the 2D kernel that maps the blurred image to the predicted image.

Levin et al. [15] give a nice overview of several of these existing deblurring techniques. Common to all of them is that they assume spatial invariance for the blur. Levin et al. show that spatial invariance is often violated, as it is only valid in limited cases of camera motion. Their experiments show that in practice in-plane camera rotation (i.e., roll), which leads to spatially varying blur kernels, is quite common.

There is relatively little work on handling spatially-varying blur. Tai et al. [16] developed a hybrid camera which captured a high frame rate video and a blurred image. Optical flow vectors from the video are used to guide the computation of spatially-varying blur kernels which are in turn used for deblurring. This method is limited by the requirement of a hybrid camera and faces problems in regions where optical flow computation fails. Tai et al.[17] use a coded exposure to produce a stroboscopic motion image and estimate motion homographies for the discrete motion steps with some user interaction, which are then used for deblurring. Their method requires close user interaction and relies on non-overlapping texture information in the blurred regions. Dai et al. [18] propose a method to estimate spatially varying blur kernels based on values of the alpha map. The method relies strongly on the pre-computation of a good alpha matte and assumes the scene to be a foreground object moving across a background. Shan et al. [19] propose a technique to handle rotational motion blur. They require user interaction for rotation cues and also rely on constraints from the alpha matte.

One approach to model the spatial variation of blur kernels is to run a blind deconvolution method at each pixel. Joshi et al. [14] do this in a limited sense, where they run their method for non-overlapping windows in an image and use this to remove spatially varying defocus blur and chromatic aberration; however, they do not address camera motion blur, nor do they try to recover a global model of the blur. Levin et al. [12] take a similar approach for object motion blur, where an image is segmented into several areas of different motion blur and then each area is deblurred independently. Hirsch et al.[20] also propose a multi-frame patch-based deblurring approach but do not impose any global camera motion constraints on the spatially-varying blur.

Unfortunately, these approaches have several limitations. First, running blind deconvolution for each pixel, window, or segment can be slow. Furthermore, it is unclear how best to handle boundaries between areas with different blur kernels, which can lead to artifacts. Second, deblurring techniques often use natural image priors, which is inherently a global constraint, and may not apply to all local areas in an image, thus leading to unreliable blur kernels and artifacts in the deblurred result.

In comparison, we do not try to recover the spatially varying blur kernels directly, but rather recover the camera motion (specifically the MDF) from which the blur kernels can be derived. In a concurrent work, Whyte et al.[21] describe a similar framework where they recover 3-dimensional rotational camera motion (roll, pitch, and yaw) to explain the spatially-varying blur. In contrast, we recover a different set of 3D camera motions (roll and x,y-translations). Our results show that these two approaches are similar for sufficiently long focal lengths due to the rotation-translation ambiguity in that focal length range. However at shorter focal lengths, each system will result in different types of artifacts depending on the errors in approximating the actual underlying camera motion. Thus, the two papers taken together form a nicely complementary set of results. We present a more detailed analysis of this rotation-translation ambiguity in Section 4.

3 A Unified Camera Shake Blur Model

In this section, we develop a unified model relating the camera motion, the latent image and the blurred image for a scene with constant depth.

3.1 Image Blur Model

Let l be the latent image of a constant depth scene and b be the recorded blurred image. The blurred image can be written as a convolution of the latent image with a kernel k and the addition of some noise n. The convolution model does not account for variations in depth and view-dependent illumination changes and we do not handle them here:

$$b = k \otimes l + n, \tag{1}$$

For simplicity, we assume Gaussian noise, $n \sim \mathcal{N}(0, \sigma^2)$.

This convolution model can also be written as a matrix-vector product:

$$B = \mathcal{K}L + N, \tag{2}$$

where L, B, and N denote the column-vector forms of l, b, and n respectively. $\mathcal{K}$ is an image filtering matrix that applies the convolution – each row of $\mathcal{K}$ is the blur kernel placed at each pixel location and unraveled into a row vector. For this reason, we also refer to $\mathcal{K}$ as the blur matrix. With spatially invariant blur each row has the same values that are just shifted in location. This matrix-vector form becomes particularly useful for formulating spatially varying blur – as each row contains a different blur kernel for each pixel [22], as we will discuss in the next section.

3.2 Blur Matrix as Motion Response

We assume the camera initially lies at the world origin with its axes aligned with the world axes. A camera motion is a sequence of camera poses where each pose can be characterized by 6 parameters - 3 rotations and 3 translations. Any camera motion can be represented as a 1D continuous path through this 6-dimensional space, which we call *camera pose space*. In a discretized version of this space, the camera spends a fraction of the exposure time at each pose; we call this proportion the *density* at that pose. Taken all together, these densities form a Motion Density Function from which a blur kernel can be directly determined for any point on the image. The MDF for all the camera poses forms a column vector over the discrete positions in the camera pose space. We denote the MDF by A where each element a_j denotes the density at the camera pose j.

The observed blurred image B is an integration over the images seen by the camera over all the poses in its path. In the discrete motion space, B is a summation over the images seen by the camera in all possible poses, each weighted by

the proportion of time spent by the camera in that pose, which in our notation is the pose's *density*. We write this mathematically as:

$$B = \sum_j a_j (K_j L) + N, \tag{3}$$

where K_j is a matrix that warps L, the latent image or the un-blurred image seen by the camera in the original pose, to the image seen in pose j. N is the noise model introduced in Section 3.1. Given a particular 6D pose (indexed by j) of a camera, we denote the corresponding homography that warps a fronto-parallel scene at depth d as P_j:

$$P_j = C(R_j + \frac{1}{d} t_j [0\ 0\ 1]) C^{-1}, \tag{4}$$

where R_j and t_j are the rotation matrix and translation vector for pose j and C is the matrix of camera intrinsics, which we form from the information in the image EXIF tags. For now we assume the depth d is known. K_j is an image warping matrix where each row contains the weights used to compute the values of pixels in the warped image by applying the inverse homography. We use bilinear interpolation for the warps and thus there are at most four non-zero values per row of K_j. For clarity, we note that K_j is a square matrix where each dimension is the width times the height of the image l.

Rearranging the linear operations in Equation 3 and comparing it with Equation 2, allows us to write the blur matrix $\mathcal{K}$ as:

$$\mathcal{K} = \sum_j a_j K_j. \tag{5}$$

Thus the K_j's form a basis set whose elements can be linearly combined using the MDF to get the corresponding blur matrix for any camera path. By definition, the blur matrix also gives us the blur kernels for each pixel location in the image. We call this basis set the *Motion Response Basis* (MRB). We note that the MRB can represent any basis in the more traditional sense, e.g., each basis matrix could actually correspond to an aggregate blur matrix itself where it captures some region of support in the 6D space.

In this work, we choose a particular basis that we found meaningful for modeling typically occurring camera motion blurs. Specifically, we choose to reduce motion in the 6D space to a 3D subspace: rotation around the z axis (*roll*) and x and y translation (modeling x translation and *yaw* and y translation and *pitch* together, respectively and neglecting the affect on z translation). We then compute the basis by point-sampling this 3D space. We discuss the validity of using a 3D space and details about creating basis sets in Section 4.

3.3 Optimization Formulation

Equation 3 relates the MDF to the latent image and the blurred image. In the process of deblurring, each basis matrix K_j is pre-computed and we solve for

the variables L and A. To do this we pose the problem in a Bayesian framework and seek to recover the latent image and MDF that is most likely given the observation and priors on the image and MDF.

We compute the maximum a posteriori (MAP) estimate, which we formulate as the minimization of the following energy function (in the interest of space, we have left out the intermediate derivation steps from the Bayesian formulation):

$$E = ||[\textstyle\sum_j a_j K_j]L - B||^2 + prior(L) + prior(A), \tag{6}$$

$$prior(L) = \phi(|\partial_x L|) + \phi(|\partial_y L|), \tag{7}$$

$$prior(A) = \lambda_1 ||A||^\gamma + \lambda_2 ||\nabla A||^2. \tag{8}$$

$prior(L)$ is the global image prior with the same parameter settings as used by Shan et al. [2]. ϕ assigns a linear penalty to small gradients and quadratic penalties to large gradients and approximates the heavy-tailed gradient distribution priors for natural images [23].

$prior(A)$ models priors that are important to recovering an accurate MDF. Specifically in 6D, the camera motion is a 1D path that captures the trajectory that the camera takes during the exposure window. This holds in the 3D space as well. Ideally, one would enforce a path prior directly on the MDF; however, this is a computationally challenging constraint to optimize. Thus we enforce two other computationally more tractable constraints.

The first component of $prior(A)$ is a sparsity prior on the MDF values. We note that while blur kernels in the 2D image space may seem quite dense, in the higher dimensional MDF space, a 1D path represents an extremely sparse population of the space. The second component of $prior(A)$ is a smoothness prior on the MDF, which also incorporates the concept of the MDF representing a path, as it enforces continuity in the space and captures the conditional probability that a particular pose is more likely if a nearby pose is likely.

We also note that we can choose to use the whole blurred image for the optimization or some selected parts by masking out rows in L, B, and the corresponding matrices.

4 Forming the Motion Response Basis

As introduced in Section 3, the Motion Response Basis (MRB) is the set of image warping matrices K_j's that correspond to a warp operation relating the image in the original camera pose to that in camera pose j. We can pre-compute the MRB; however, the size of the basis set is critical for the computational feasibility of the system. We now discuss the issues involved in this computation.

4.1 Dependence on Scene Depth

As discussed in Section 3.2, it is necessary to know the scene depth d to compute the homographies P_j and corresponding basis matrices K_j. Unfortunately, recovering the depth of a scene from a single image is an under-constrained problem.

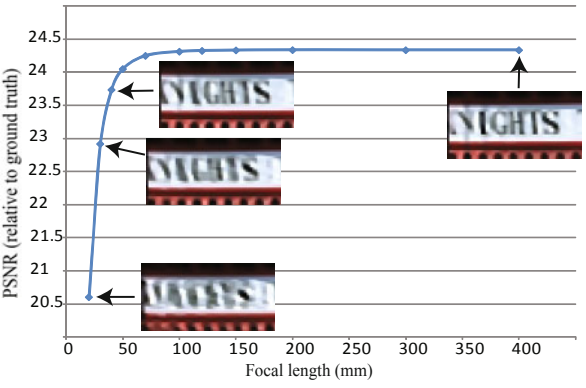

Fig. 1. PSNR of the deblurred result (by recovering *yaw* using x translation) with respect to the ground truth image. Cropped parts from deblurred images for some plot samples are also shown for qualitative comparison.

Fortunately, given our assumption of a constant depth or fronto-parallel scene, we observe that we do not need to know the exact depth and rather can consider $\frac{1}{d}t_j$ as a single 3-dimensional variable, which allows us to remove the dependence on depth and instead only concern ourselves with the image parallax. Given this, the number of degrees of freedom of our system does not change, depth is not needed as a separate variable. Computing a basis that captures the parallax reduces to sampling the total resulting image plane translation, appropriately. We discuss how to choose the sampling resolution in Subsection 4.3.

4.2 Computational Reduction in d.o.f for the Camera Motion

We observe that instead of using 6 degrees of freedom for the camera motion, we can use only 3 degrees of freedom - *roll* (rotation about z-axis) and x and y translations. This reduction makes the problem computationally more feasible since the size of the MRB is dependent on the number of degrees of freedom. Given the projective camera model, it is known that small camera rotations about the x (*pitch*) and y (*yaw*) axes can be approximated by camera translations when perspective affects are minimal (i.e., longer focal lengths). Joshi et al [24] show that in most cases the camera shake motion lies in this operating range. To validate this approximation, we performed an experiment with a ground truth image blurred using a synthetic camera motion involving *yaw*. We then solve for an MDF limited to only x translations. Figure 1 shows the PSNR values comparing the resulting deconvolved images to the ground truth as we vary the focal length. We varied the amount of *yaw* in relation to focal length to keep the blur kernels approximately the same size ($\sim$ 11 pixels wide) so that deconvolution errors due to differences in blur kernel size do not affect the analysis. The PSNR improvement levels out quickly, which means that the recovered translations

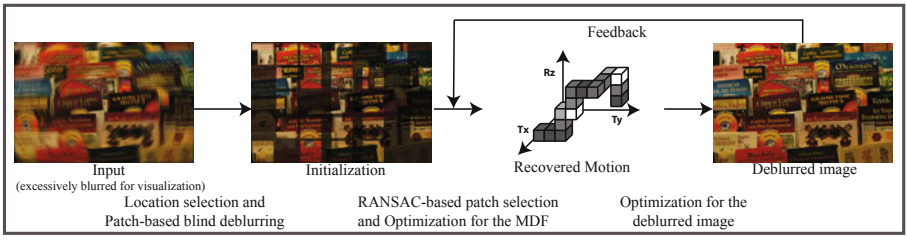

Fig. 2. Our System Pipeline

start to accurately approximate the *yaw* as the focal length increases to a value that covers most standard camera settings. A similar argument also holds for *pitch* to be approximated by y translations. We do a similar analysis for the effect of z translations of the camera and found that their contribution is also negligible under typical camera shake motion. We provide more analysis including similar figures for *pitch* and z translations on the project webpage [25].

As a result, the full 6D space of camera motion can be accurately approximated using only samples of the 3D space of *roll*, x and y translations across a wide range of focal lengths. We note that 6D motions can still be theoretically solved using our framework, but the high dimensionality makes the solution computationally prohibitive.

4.3 Sampling Range and Resolution of the Camera Motion Space

The number of matrices K_j is the number of motion poses that we sample. The number of samples along each d.o.f. affects the size of the MRB and hence we want to keep the sampling as coarse as possible. We hypothesize that the sampling needs to only be dense enough that the neighboring voxels in the discretized motion space project to within a pixel width at any image location. The range of the motion can be chosen to cover the estimate of the kernel size that is initially specified by the user. Hence we automatically choose the sampling range and resolution along the 3 degrees of freedom and pre-compute the K_j's.

5 Our System

The proposed optimization in Equation 6 is non-linear in the variables L and A. We solve this using an alternating, iterative EM-style procedure which takes an initialization for L and then optimizes for A and L successively. Figure 2 shows the steps in our system pipeline, and we explain each of the steps now.

5.1 Generating an Initial Estimate for the Latent Image

We first select uniformly distributed patches on the blurred image which we independently deblur for generating an initial estimate for the latent image L.

The patch sizes are proportional to the estimated maximum blur kernel size in the image, which is an input parameter for our system. Since kernel estimation techniques require a good distribution of edge orientations [14], we filter out the patches having a low average value of the Harris corner metric. The Harris corner metric measures the presence of gradients in orthogonal directions in an image region and hence is a good estimate for the success of kernel estimation. We empirically choose this threshold value to be 0.1. We denote these selected patches as p_i's. We then use the blind deconvolution approach proposed by Shan et al [2] to deblur each of these patches independently. We denote the corresponding deblurred patches and blur kernels as d_i's and k_i's, respectively. These deblurred patches are the initial estimates for the latent image in corresponding regions.

5.2 Ransac-Based Optimization for the MDF

Assuming we know L, we can solve for A by minimizing the following function which is a reduced form of Equation 6.

$$E = ||\sum_j a_j(K_j L) - B||^2 + \lambda_1||A||^\gamma + \lambda_2||\nabla A||^2 \tag{9}$$

Here we only use the parts of the image regions of L and B which are covered by patches p_i's. This optimization is performed using an iterative re-weighted least squares (IRLS). We use the values of $\gamma = 0.8$, $\lambda_1 = 0.1$ and $\lambda_2 = 0.5$ in all our experiments. In practice, we see that using five iterations of IRLS works well.

We have found that using all the deblurred patches from the initialization phase does not give good results. This is because blind blur kernel estimation on a patch can vary in performance based on the quality and quantity of texture information and image noise. Ideally, we would want to select the best deblurred patches in the image for fitting the MDF. Unfortunately, this is a hard problem to solve in itself. There are numerous metrics that have been used in the literature for this classification – penalizing based on a heavy-tailed gradient distribution ([23]) and slope of the power spectrum ([26]); however, we have not found these to work well.

Instead, we use a RANSAC-based scheme to robustly choose a set of "good" patches from the initial set of deblurred patches. Each RANSAC iteration randomly chooses 40% of the patches and fits an MDF to them by minimizing Equation 9. We classify each of the patches, p_i's, as inliers or outliers by how well the MDF describes the corresponding blur kernel. We consider this a contribution of using an MDF – the process of fitting a lower-dimensional MDF to blurred/deblurred patch pairs allows us to measure the quality of deblurring in local image patches, which is otherwise difficult.

Specifically, to compute the inlier metric, let k_i' be the recovered kernel using the MDF, the residual error is given as, $||d_i * k_i' - b_i||_2$. A patch is an inlier if its residual error is less than 1.2 times the median of all the errors in the patch set. From all the RANSAC iterations, we select the set of patches which gives the minimum average residual error on the inliers. Finally, we fit an MDF using all the inlier patches.

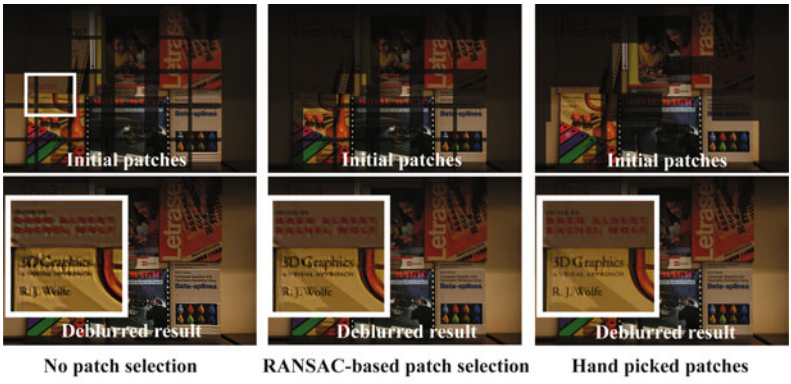

Fig. 3. Deblurred results using different schemes for choosing good deblurred patches for initialization. Handpicking patches works better than RANSAC-based selection which works better than no selection at all.

To test the performance of our RANSAC approach, we ran our complete deblurring pipeline on three cases – (A) using all the initially deblurred image patches, (B) automatically choose inliers from the initially deblurred patches using RANSAC, and (C) handpicking patches to be deblurred. Figure 3 shows the deblurring results for these three cases, and we see that using handpicked patches works better than RANSAC which in turn works better than doing no patch selection. We use the RANSAC-based scheme in all our experiments since it is robust, automatic, and gives reasonable results.

5.3 Optimization for the Latent Image

Assuming we know the MDF A, we can solve for L by minimizing the following function which is another reduced form of Equation 6. This is essentially a non-blind image deconvolution:

$$E = ||\sum_j (a_j K_j)L - B||^2 + \phi(|\partial_x L|) + \phi(|\partial_y L|). \tag{10}$$

We solve this optimization as described by Shan et al. [23] in their paper. We feed the solution back into the RANSAC-based MDF optimization and repeat the overall procedure until the latent image converges. We have observed that 2-3 iterations are enough for convergence of the recovered latent image in all our experiments.

6 Experiments and Results

We run our deblurring experiments on a quad dual-core 3.0GHz PC with 64GB RAM. Running our system on a 768X512 sized image takes about 1 hour and

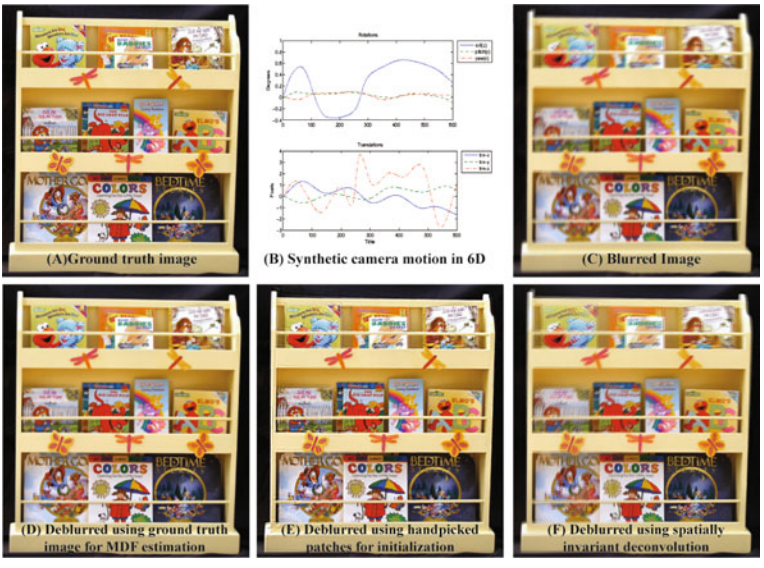

Fig. 4. Model validation. (A) Ground truth image, (B) Synthetic camera motion, (C) Blurred image, (D) Deblurred output using ground truth initialization, (E) Deblurred output using handpicked locally blind deblurred image regions for initialization, (F) Result with a globally spatially invariant deblurring system. (Please zoom in to compare.)

takes up around 8 GBs of RAM. As there are no existing single image automatic deblurring systems for a general (spatially-varying) camera motion blur, we perform all our comparisons with the recent blind deconvolution method of Shan et al [2], who have code available online.

6.1 Model Validation Using Synthetic Data

Figure 4 shows the visual validation of our model formulation and illustrates sensitivity to the initialization values. We take a sharp image (A) and use a specified 6D camera motion (B) to blur it (C). We then optimize for the 3D MDF, (z-rotation and x, y-translations) using the original sharp image as the initialization (D) and using some handpicked locally blind deblurred image regions for initialization (E). (F) shows the corresponding result for a blind spatially invariant deblurring system. We see that (D) is very close to the ground truth which means that if we start with the ideal initialization, we can recover a very accurate 3D approximation of the true 6D MDF. We do see some artifacts (upper left corner) in (E) which shows that our system is sensitive to the initial latent image. But we still out-perform the result in (F), which assumes only a translational motion of the camera or in other words, a spatially invariant blur kernel. We show more such comparison sets with synthetic data on the project webpage [25].

Fig. 5. Deblurring result. (A) Blurred image, (B) Deblurred image using spatially invariant approach, (C) Deblurred result using our system. Recovered blur kernels at few locations are shown in yellow boxes.

Fig. 6. Deblurring result. (A) Blurred image, (B) Deblurred image using spatially invariant approach, (C) Deblurred result using our system. Recovered blur kernels at few locations are shown in yellow boxes.

6.2 Results and Comparisons on Real-World Data

Figure 5 shows one of our results for real-world blurred images of scenes captured using a Canon EOS-1D camera. It shows the original blurred image (A), the deblurred result using spatially invariant deconvolution (B), our deblurred result (C), and the inset comparisons between (B) and (C). Our approach shows a significant improvement over the spatially invariant approach in all the cases. Our current implementation does not handle depth variance in the scene. Figure 6 is a difficult example as it has large depth variation, yet our deblurring method performs better than the spatially invariant approach and gives a reasonable looking deblurred result. This shows that our system can handle depth variation until the point where it starts to cause a large amount of spatial variation in the blur kernels. We also provide more results and intermediate step images for each of these results on the project webpage [25].

7 Discussion and Future Work

We presented a unified model of camera shake blur and a framework to recover the camera motion and latent image from a single blurred image. One limitation of our work is that it depends on imperfect spatially invariant deblurring estimates for initialization. Two things could improve this: (a) using other blur estimation methods for initialization and (b) a better metric to judge the accuracy of a particular kernel estimate, which is still a very open and interesting problem.

Another interesting area of work is to explore other motion response bases. Instead of using a uniform sampling with delta functions, a more sophisticated basis with larger, more complex support regions may be more appropriate for modeling common camera blurs.

Acknowledgements

We would like to thank the reviewers for their insightful comments. We would also like to thank Qi Shan for useful discussions regarding blind/non-blind image deblurring methods. This work was supported by funding from the University of Washington Animation Research Labs, Microsoft, Adobe, and Pixar.

References

1. Richardson, W.H.: Bayesian-based iterative method of image restoration. Journal of the Optical Society of America (1917-1983) (1972)
2. Shan, Q., Jia, J., Agarwala, A.: High-quality motion deblurring from a single image. In: ACM SIGGRAPH 2008 Papers (2008)
3. Likas, A.C., Galatsanos, N.P.: A variational approach for bayesian blind image deconvolution (2004)

4. Fergus, R., Singh, B., Hertzmann, A., Roweis, S.T., Freeman, W.T.: Removing camera shake from a single photograph. In: ACM SIGGRAPH 2006 Papers (2006)
5. Yuan, L., Sun, J., Quan, L., Shum, H.Y.: Image deblurring with blurred/noisy image pairs. In: ACM SIGGRAPH 2007 Papers (2007)
6. Joshi, N., Zitnick, L., Szeliski, R., Kriegman, D.: Image deblurring and denoising using color priors. In: Proceedings of IEEE CVPR '09 (2009)
7. Ben-Ezra, M., Nayar, S.K.: Motion-based motion deblurring. IEEE Trans. Pattern Analysis Machine Intelligence (2004)
8. Levin, A., Sand, P., Cho, T.S., Durand, F., Freeman, W.T.: Motion-invariant photography. In: ACM SIGGRAPH 2008 Papers (2008)
9. Agarwal, A., Xu, Y.: Coded exposure deblurring: Optimized codes for psf estimation and invertibility. In: Proceedings of IEEE CVPR '09 (2009)
10. Agarwal, A., Xu, Y., Raskar, R.: Invertible motion blur in video. In: ACM SIGGRAPH 2009 Papers (2009)
11. Raskar, R., Agrawal, A., Tumblin, J.: Coded exposure photography: motion deblurring using fluttered shutter. In: ACM SIGGRAPH 2006 Papers (2006)
12. Levin, A.: Blind motion deblurring using image statistics. In: Advances in Neural Information Processing Systems (2007)
13. Jia, J.: Single image motion deblurring using transparency. In: Proceedings of IEEE CVPR '07, pp. 1–8 (2007)
14. Joshi, N., Szeliski, R., Kriegman, D.J.: Psf estimation using sharp edge prediction. In: Proceedings of IEEE CVPR '08 (2008)
15. Levin, A., Weiss, Y., Durand, F.: Understanding and evaluating blind deconvolution algorithms. In: Proceedings of IEEE CVPR '09 (2009)
16. Tai, Y.W., Du, H., Brown, M., Lin, S.: Image/video deblurring using a hybrid camera. In: Proceedings of IEEE CVPR '08 (2008)
17. Tai, Y.W., Kong, N., Lin, S., Shin, S.Y.: Coded exposure imaging for projective motion deblurring. In: Proceedings of IEEE CVPR '10 (2010)
18. Dai, S., Wu, Y.: Motion from blur. In: Proceedings of IEEE CVPR '08 (2008)
19. Shan, Q., Xiong, W., Jia, J.: Rotational motion deblurring of a rigid object from a single image. In: Proceedings of IEEE ICCV '07 (2007)
20. Hirsch, M., Sra, S., Schölkopf, B., Harmeling, S.: Efficient filter flow for space-variant multiframe blind deconvolution. In: Proceedings of IEEE CVPR '10 (2010)
21. Whyte, O., Sivic, J., Zisserman, A., Ponce, J.: Non-uniform deblurring for shaken images. In: Proceedings of IEEE CVPR '10 (2010)
22. Seitz, S., Winder, S.: Filter flow. In: Proceedings of IEEE ICCV '09 (2009)
23. Weiss, Y., Freeman, W.T.: What makes a good model of natural images? In: Proceedings of IEEE CVPR '07 (2007)
24. Joshi, N., Kang, S.B., Zitnick, C.L., Szeliski, R.: Image deblurring using inertial measurement sensors. In: ACM SIGGRAPH 2007 Papers (2010)
25. Gupta, A.: Project webpage: Single image deblurring using motion density functions (2010), `http://www.grail.cs.washington.edu/projects/mdf_deblurring`
26. Reinhard, E., Shirley, P., Ashikhmin, M., Troscianko, T.: Second order image statistics in computer graphics. In: Proceedings of the ACM Symposium on Applied Perception in Graphics and Visualization (2004)