

# Fast Removal of Non-uniform Camera Shake

Michael Hirsch, Christian J. Schuler, Stefan Harmeling and Bernhard Schölkopf  
Max Planck Institute for Intelligent Systems, Tübingen, Germany

{mhirsch, cschuler, harmeling, bs}@tuebingen.mpg.de

[http://webdav.is.mpg.de/pixel/fast\\_removal\\_of\\_camera\\_shake](http://webdav.is.mpg.de/pixel/fast_removal_of_camera_shake)

## Abstract

*Camera shake leads to non-uniform image blurs. State-of-the-art methods for removing camera shake model the blur as a linear combination of homographically transformed versions of the true image. While this is conceptually interesting, the resulting algorithms are computationally demanding. In this paper we develop a forward model based on the efficient filter flow framework, incorporating the particularities of camera shake, and show how an efficient algorithm for blur removal can be obtained. Comprehensive comparisons on a number of real-world blurry images show that our approach is not only substantially faster, but it also leads to better deblurring results.*

## 1. Introduction

Camera motion during longer exposure times, e.g., in low light situations, is a common problem in handheld photography. It causes image blur that destroys details in the captured photo. Single image blind deconvolution or motion deblurring aims at restoring the sharp latent image from its blurred picture without knowing the camera motion that took place during the exposure. Blind deconvolution has several challenging aspects: modeling the image formation process, formulating tractable priors based on natural image statistics, and devising efficient methods for optimization.

Early attempts to remove real camera shake model the blur as a space-invariant convolution [4, 19] and recent approaches [2, 25] to the case of uniform blur yield impressive results both in speed and quality. However, this model is only sufficient if the camera shake is inside the sensor plane without any rotations. If the camera tilts or rotates the blur becomes non-uniform, i.e. different locations in the image are blurred differently.

In general one can view an image that has been blurred by camera shake as the result of integrating all intermediate images the camera “sees” along the trajectory of the camera shake. These intermediate images are differently projected copies (i.e. homographies) of the true sharp scene. This insight recently led to the so-called Projective Motion Path

Blur models (PMPB models), that have been proposed by several authors to model non-uniform blur due to camera shake [21, 24, 5]. Such models have the benefit of ruling out sets of blur kernels that do not correspond to a valid camera motion. However, the currently available approaches suffer from high computational cost, because during the optimization many homographies of the intermediate estimated images have to be computed.

A different approach to model non-uniform blur was proposed recently as *Efficient Filter Flow* (EFF) in the context of imaging through air turbulence [7]. By a position-dependent combination of a set of localized blur kernels, the EFF framework is able to express smoothly varying blur while still being linear in its parameters. Making use of the FFT, an EFF transformation can be computed almost as efficiently as an ordinary convolution, while being much more expressive. However, the EFF framework does not impose any global camera motion constraint on the non-uniform blur. This renders kernel estimation for single images a delicate task, especially in image regions with little structural information, and heuristics need to be used to propagate information about blur kernels across such regions [6].

In this paper, we combine the ideas of these two recent developments, i.e. we combine the structural constraints of the PMPB models and the efficiency of the EFF framework to obtain a fast single image blind deconvolution algorithm that is able to handle non-uniform blur caused by camera shake and provides comparable or better results than existing methods. At the same time, it is computationally more efficient.

The paper is outlined as follows: Section 2 discusses related work, in 3 we show how to combine the PMPB and EFF framework and deduce a fast forward model for camera shake. In 4 we propose an efficient deblurring algorithm and do a comprehensive comparison in 5. We conclude our paper with a discussion of current limitations in 6.

## 2. Related work

The problem of removing blur caused by space-invariant convolution, i.e. uniform blur has been studied extensively

for a long time. Early works include e.g., Richardson [18] and Lucy [13], which date back to the early 70s. See Kundur and Hatzinakos [10] for an overview of related methods.

For blind deconvolution of single photographs, Fergus et al. [4] combined the variational approach of Miskin and MacKay [14] with natural image statistics. Shan et al. [19], Cho and Lee [2] and Xu et al. [25] refined that approach using carefully chosen regularization and fast optimization techniques, see also Levin et al. [12] for an overview of these approaches.

However, all these methods assume a uniform blur model based on space-invariant convolution, which is a severe limitation since real camera shake not only translates the sensor but often also tilts and rotates it, which generates non-stationary (i.e. space-variant) blur. This motivated work on non-uniform blur models which we already discussed in the introduction [22, 24, 5, 6]. A further work on space-variant blurs in the context of astronomical imaging is Bardsley et al. [1].

Other generalizations of the uniform blur model consider object motion instead of camera motion: Levin [11] is able to deblur objects that move linearly, such as a bus that drives from left to right. Shan et al. [20] focus on blurs in the image due to rotating objects, such as propellers.

Hardware approaches to obtain sharper images are based on manipulating the way images are taken, exemplarily we mention: Yuan et al. [26] reconstruct a single sharp image from a pair of blurred and noisy images. Raskar et al. [17] encodes the movement of objects by “fluttering” the shutter. Joshi et al. [8] exploit motion sensor information to recover the true trajectory of the camera during the shake.

### 3. Fast forward model for camera shake

Let  $g$  be the blurry photograph for which we would like to recover a sharp version  $f$ . EFF approximates a non-stationary blur as the sum of  $R$  differently blurred patches,

$$g = \sum_{r=1}^R a^{(r)} * \left( w^{(r)} \odot f \right), \quad (1)$$

where the weighting image  $w^{(r)}$  has the same size as the sharp image  $f$ , so the  $r$ -th patch can be written as  $w^{(r)} \odot f$  with  $\odot$  being the pixel-wise (Hadamard) product. By  $a^{(r)}*$  we denote the stationary blur (aka convolution) with the  $r$ -th blur kernel  $a^{(r)}$ . Since  $w^{(r)} \geq 0$  is chosen to be only non-zero in a small region of  $f$ , the convolution of each patch can be implemented efficiently with short fast Fourier transforms (FFTs). If the patches are chosen with sufficient overlap and the  $a^{(r)}$  are distinct, the blur expressed by the EFF will be different at each image location, varying gradually from pixel to pixel.

To restrict the possible blurs of EFF to camera shake, we create a basis for the blur kernels  $a^{(r)}$  using homogra-

phies. However, instead of applying the homographies to the sharp image  $f$  (as done in [24, 5]), we apply all possible homographies only once to a grid of single-pixel dots. The possible camera shakes can then be generated by linearly combining different homographies of the point grid, see Fig. 1. Note that those homographies can be precomputed without knowledge of the blurry image  $g$ .

For concreteness, let  $p$  be the image of delta peaks, where the peaks are exactly located at the centers of the patches, and  $p$  has the same size as the sharp image  $f$ . Note that the center of a patch is determined by the center of the support of the corresponding weight images  $w^{(r)}$ . Let  $\theta$  index a set of homographies and denote a particular homography by  $H_\theta$ . Then we can generate different views  $p_\theta = H_\theta(p)$  of the point grid  $p$  by applying a homography  $H_\theta$ , where  $p_\theta$  is again an image of the same size as  $f$ . Appropriately chopping these views  $p_\theta$  into local blur kernels  $b_\theta^{(r)}$ , one for each patch, we obtain a basis for the local blur kernels  $a^{(r)}$ , which we can now restrict to linear combinations of the basis blur kernels,

$$a^{(r)} = \sum_{\theta} \mu_{\theta} b_{\theta}^{(r)}, \quad (2)$$

where  $\mu_{\theta}$  determines the relevance of the corresponding homography for the overall blur, i.e.  $\mu_{\theta}$  does not depend on the patch index  $r$ . Note that the construction of the blur kernel basis ensures that the resulting overall blur parametrized by the  $a^{(r)}$  corresponds to a possible camera shake. Also note that such linear combinations correspond not to a single homography, but linearly combine several homographies, which form a possible camera shake. Plugging Eq. (2) into Eq. (1) we obtain our fast forward model for camera shake:

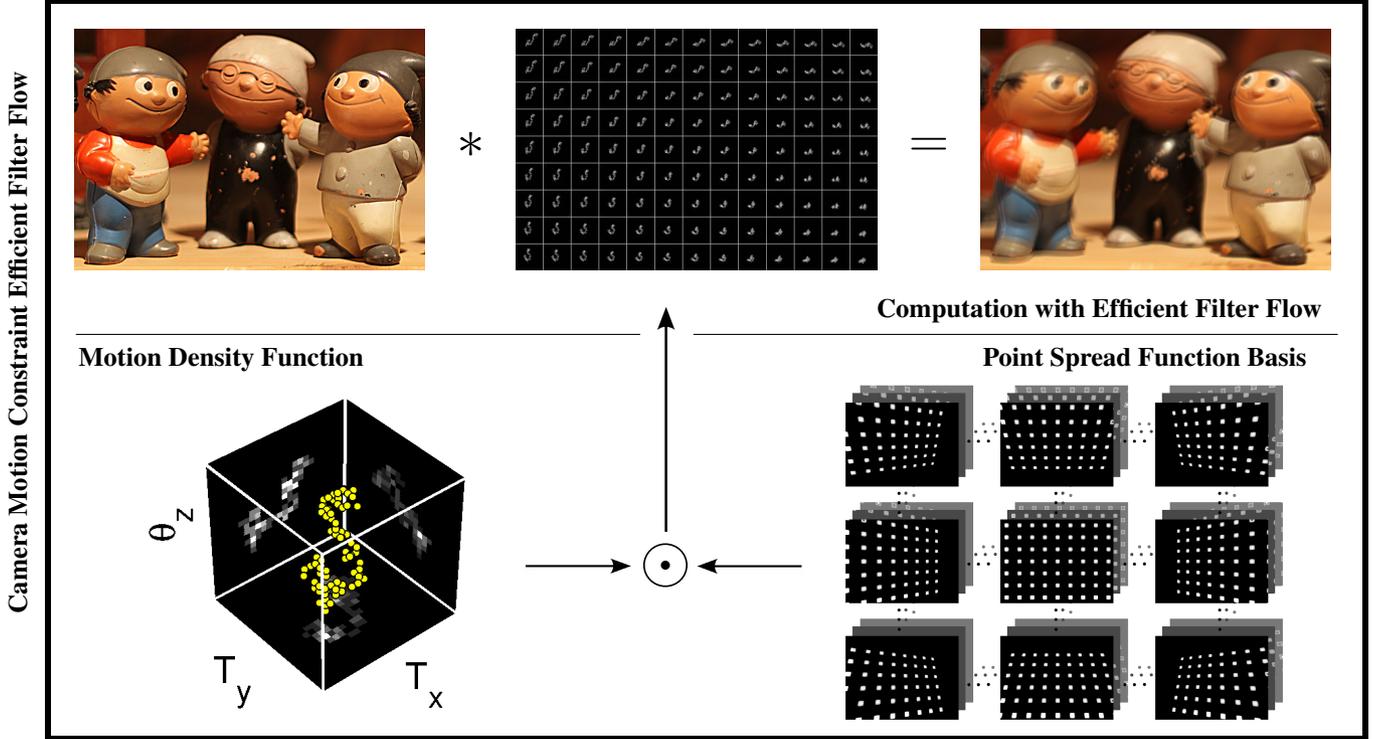
$$g = \mu \diamond f := \sum_r \left( \sum_{\theta} \mu_{\theta} b_{\theta}^{(r)} \right) * \left( w^{(r)} \odot f \right), \quad (3)$$

Here we introduced the notation  $\mu \diamond f$  to denote our proposed fast forward model for camera shake. Note that this model parametrizes the camera shake with the parameters  $\mu_{\theta}$ , which we summarize in the vector  $\mu$ . These parameters appear only linearly, which is crucial for efficient optimization. Likewise, the sharp image  $f$  appears only linearly for fixed  $\mu$ . Thus there exist matrices  $M$  and  $A$ , such that our forward model for vector-valued images can be expressed as matrix-vector-multiplications (MVMs),

$$g = \mu \diamond f = Mf = A\mu. \quad (4)$$

For convenience, we will henceforth use this notation; all formulas for vector-valued images can be straightforwardly generalized to matrix-valued images.

To efficiently evaluate the forward model, or in other words to compute the MVM with  $M$  or  $X$ , we first calculate all  $a^{(r)}$  using the blur parameters  $\mu$  and blur kernel



**Figure 1:** The values of the Motion Density Function (bottom left, plotted with `plot_nonuni_kernel.m` from Oliver Whyte; exemplarily, rotation around the optical axis (roll) and in-plane translations are depicted only) correspond to the time spent in each camera pose. Linearly combined with the blur kernel basis (bottom right), it yields a non-uniform PSF (top middle) which parametrises the EFF transformation allowing fast computation. By construction, our forward model permits only physically plausible camera motions. The blur kernel basis has to be computed only once and allows a memory saving sparse representation. The dimensionality and size of the blur kernel basis depends on the motion considered. For translational motion only, the model reduces naturally to the uniform blur model. In this case the Motion Density Function equals the invariant PSF.

basis (with Eq. (2)) and then run the fast implementation of EFF detailed in Hirsch et al. [7]. Similarly we can obtain fast implementations of the MVMs with  $M^T$  and  $A^T$ .

The homography calculations on the point grid  $p$  are pre-computed, and neither required after updating the blur parameters  $\mu_\theta$  nor after updating the estimate of the sharp image estimate. This fact is essential for our method’s fast run-time. Fig. 2 compares the run-time of our forward model in dependence of both the image and blur size for camera shake to Whyte et al. [24]. There, the computation of a forward model consists of making  $d$  homographies on an image with  $n$  pixels, which means a complexity of  $O(n \cdot d)$ . Since our model uses the EFF, the complexity is  $O(n \cdot \log q)$  with the number  $q$  of pixels in a patch [6], which depends on the image and PSF sizes. The disadvantage in  $\log q$  is easily outweighed even for a small number of homographies. Furthermore, Fig. 3 shows that our fast forward model can approximate the non-stationary blur of Whyte and Gupta almost perfectly with as little kernels as  $16 \times 12$  for an image of size  $1600 \times 1200$  pixels. We mention in passing that the blur kernel basis can be represented as sparse matrices

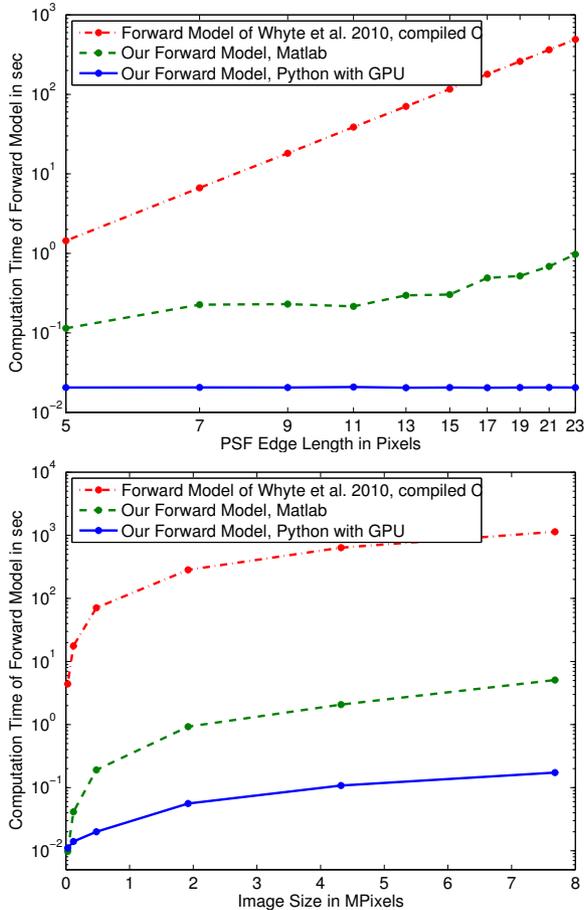
which require less memory than storing large transformation matrices as done by Gupta et al. [5].

## 4. Deconvolution of non-stationary blurs

Starting with a photograph  $g$  that has been blurred by camera shake, we recover the unknown sharp image  $f$  in two phases: (i) a blur estimation phase for non-stationary PSFs, and (ii) the sharp image recovery using a non-blind deconvolution procedure, tailored to non-stationary blurs. In the following, we will describe both phases in detail and where appropriate we explicitly include the values of hyper-parameters that determine the weighting between the terms involved. These values were fixed during all experiments.

### 4.1. Blur estimation phase

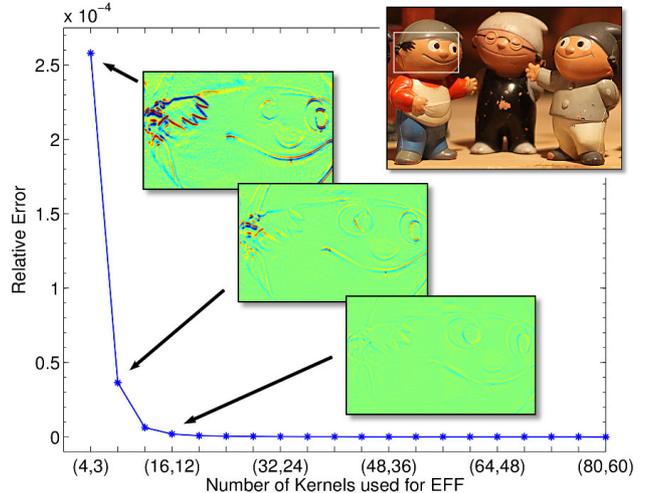
In the first phase of the algorithm, we try to recover the motion undertaken by the camera during exposure given only the blurry photo. To this end, we iterate the following three steps: (i) *prediction step* to reduce blur and enhance image quality by a combination of shock and bilateral filter-



**Figure 2:** Run-time comparison of our forward model with the blurring model of [24, 5] as a function of PSF (top) and image size (bottom). For an image of size  $1600 \times 1200$  pixels our Matlab implementation is a factor 40 faster than the compiled C code of [24]. Note that for fair comparison computation was performed on a single core machine as our Matlab implementation is able to take advantage of a multicore architecture by parallel computation while the implementation of [24] does not. A factor of 1000 can be gained by our Python implementation supporting GPU computation.

ing, (ii) *blur parameter estimation step* to find the camera motion parameters, which best explain the blurry picture from the predicted image of step (i), and (iii) *latent image estimation* via non-blind deconvolution.

To avoid local minima and robustify the blur estimation in the first phase, we employ a coarse to fine approach (over several scales). In the beginning the resolution of the recorded image  $g$  is reduced and the blur estimation phase is performed. Then the lower resolution blur estimate initializes the next scale, and so on, up to the full resolution of the recorded image. At the coarsest scale we initialize the unknown sharp image  $f$  by a downsampled version of the



**Figure 3:** The curve shows the relative error of a homographically transformed image ( $1600 \times 1200$  pixels) using the forward model of Whyte et al. and our fast forward model which approximates the homography by the camera motion constrained EFF framework. For some data points closeups of the difference images are shown. The relative error decreases the more kernels are used. With as little as  $16 \times 12$  kernels the error is negligible.

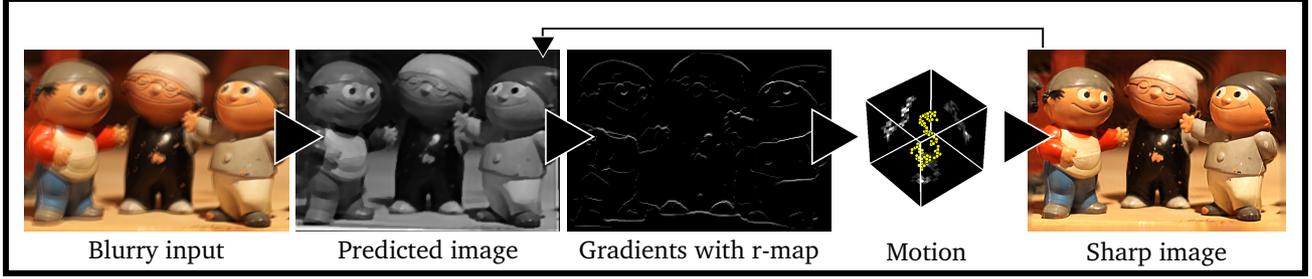
blurry image  $g$ . For a fixed scale we iterate steps (i)-(iii), which are visualized in Fig. 4 and will be detailed in the following, five times.

**(i) Prediction step:** The blur parameters are estimated by finding the best non-stationary blur which transforms the current image estimate  $f$  into the recorded blurry image  $g$ . However, in the beginning, the current estimate  $f$  might not be even close to the sharp image, and after a few iterations, it might still be slightly blurry. To accelerate the convergence of the blur parameter estimation [15, 2], the prediction step emphasizes edges in  $f$  by shock filtering [16] and lowers the importance of flat noisy regions by bilateral filtering [23].

**(ii) Blur parameter update step:** For notational simplicity we assume that  $g$ ,  $\mu$ , and  $f$  are vector-valued. The blur parameters  $\mu$  are updated by minimizing

$$\|\partial g - m_S \odot \partial(\mu \diamond \tilde{f})\|_2^2 + \frac{1}{10} \|\mu\|_2^2 + \frac{1}{2} \|\partial \mu\|_2^2, \quad (5)$$

where we write the discrete derivative of  $g$  symbolically as  $\partial g$ , i.e.  $\partial g = [1, -1]^T * g$ . For matrix-valued images we consider the horizontal and vertical derivatives. Furthermore,  $\tilde{f}$  denotes the outcome of the prediction step (i) and  $m_S$  is a weighting mask which selects only edges that are informative and facilitate kernel estimation. In particular, it neglects structures that are smaller in size than the local



**Figure 4:** Overview of the blur estimation phase. See text for details.

kernels, which could be misleading for the kernel estimation [25]. For computing  $m_S$  we employ the r-map approach of Xu et al. as detailed in [25].

The terms in Eq. (5) can be motivated as follows: The first term is proportional to the log-likelihood,  $\|\partial g - m_S \odot \partial(\mu \diamond f)\|_2^2$  if we assume additive Gaussian noise  $n$ . Considering the derivatives of  $g$  and  $\mu \diamond f$  brings several benefits: First, Shan et al. [19] have shown that such terms with image derivatives help to reduce ringing artifacts by putting weight on the edges. Secondly, it lowers the condition number of the optimization problem Eq. (5) and hence leads to faster convergence [2]. The second summand  $\|\mu\|_2^2$  penalizes the  $L_2$  norm of  $\mu$  and helps to avoid the trivial solution by suppressing high intensity values in  $\mu$ . The third term  $\|\partial\mu\|_2^2$  enforces smoothness of  $\mu$ , and thus favors connectedness in camera motion space, see also Gupta et al. [5].

**(iii) Sharp image update step:** The sharp image estimate  $f$  that is repeatedly updated during the blur estimation phase does not need to recover the true sharp image perfectly. However, it should guide the PSF estimation during the alternating updates, i.e. steps (i), (ii), and (iii). Since most computational time is spent in this first phase, the sharp image update step should be fast. This motivates to employ  $L_2$  based regularization terms for the sharp image, even though the resulting estimates might show some ringing and possibly other artifacts (which are dealt with in the prediction step). Thus we would like to minimize

$$\|g - \mu \diamond f\|_2^2 + \frac{1}{2} \|\partial f\|_2^2 \quad (6)$$

with respect to  $f$ .

Cho and Lee [2] gained large speed-ups for this step by replacing the iterative optimization in  $f$  by a pixel-wise division in Fourier space. They showed that such a non-iterative update step despite its known restoration artifacts is sufficient to guide the PSF estimation. We call such a pixel-wise division in Fourier space *Direct Deconvolution* (DD) and provide a similar update for our fast forward model for camera shake.

First, we adapt the matrix expression given in [7] to ob-

tain an explicit expression for  $M$  introduced in Sec. 3,

$$g = \underbrace{\sum_r E_r^T F^H \text{Diag}(F Z_a B^{(r)} \mu)}_M F C_r \text{Diag}(w^{(r)}) f, \quad (7)$$

where  $B^{(r)}$  is the matrix with column vectors  $b_\theta^{(r)}$  for varying  $\theta$ , i.e.  $a^{(r)} = B^{(r)} \mu = \sum_\theta \mu_\theta b_\theta^{(r)}$ , see also Eq. (2) and Fig. 1. Matrices  $C_r$  and  $E_r$  are appropriately chosen cropping matrices,  $F$  is the discrete Fourier transform matrix, and  $Z_a$  a zero-padding matrix. Furthermore, we denote by  $\text{Diag}(v)$  the diagonal matrix with vector  $v$  along its diagonal.

The basic idea for a direct update step of the image estimate is to combine the patch-wise pixel-wise divisions in Fourier space with reweighting and edge fading to minimize ringing artifacts. We use the following expression to approximately “invert” our forward model  $g = Mf$ :

$$f \approx \text{Diag}(v) \sum_r \text{Diag}(w^{(r)})^{1/2} C_r^T F^H \frac{F Z_a B^{(r)} \mu \odot (F E_r \text{Diag}(w^{(r)})^{1/2} g)}{|F Z_a B^{(r)} \mu|^2 + \frac{1}{2} |F Z_l l|^2} \quad (8)$$

where  $|z|$  for a vector  $z$  with complex entries calculates the entry-wise absolute value, and  $\bar{z}$  the entry-wise complex conjugate. The square root is taken pixel-wise. The term  $\text{Diag}(v)$  is some additional weighting which we experimentally justify in the next paragraph. The fraction has to be implemented pixel-wise. The term  $|F Z_l l|^2$  in the denominator of the fraction originates from the regularization in Eq. (6) with  $l = [-1, 2, -1]^T$  corresponding to the discrete Laplace operator.

Note that the update formula in Eq. (8) approximates the true sharp image  $f$  given the blurry photograph  $g$  and the blur parameters  $\mu$  and can be implemented efficiently by reading it from right to left. The image rightmost in Fig. 5 demonstrates how well Eq. (8), i.e. direct deconvolution, *without* the additional weighting term (i.e.  $v = 1$ ) approximates the true image, but also reveals artifacts stemming from the windowing. By applying the additional weighting term  $v$ , these artifacts can be suppressed effectively, as

Size in pixels		Processing time in seconds			
Image	Kernel	GPU			CPU
		A	B	C	C
$354 \times 265$	$21 \times 21$	135.9	0.7	136.6	724
$441 \times 611$	$15 \times 15$	169.7	0.8	170.5	1567
$1123 \times 749$	$21 \times 21$	439.8	1.3	441.1	3860

**Table 1:** Run time of our Matlab and GPU implementation for several deblurring examples. A: kernel estimation. B: final deconvolution. C: total processing time.

can be seen in the middle panel of Fig. 5. The weighting  $v$  is computed by applying Eq. (8) without the additional weighting term to a constant image of the same size as the blurred image  $g$ . The deconvolved constant image reveals the same artifacts as present in the rightmost image of Fig. 5. By taking its inverse pixel-wise, it serves as a corrective weighting term, which is able to remove most artifacts caused by the windowing and at the same time is fast to compute.

## 4.2. Sharp Image Recovery Phase

After having estimated and fixed the blur parameters  $\mu$ , we recover the final sharp image  $f$  by replacing the  $L_2$  image prior of the sharp image update step (6) by a natural image prior that is based on sparsity of the gradient images (e.g. Fergus et al. [4]), i.e. we minimize

$$\|g - \mu \diamond f\|_2^2 + \nu \|\partial f\|_\alpha^\alpha \quad (9)$$

where the  $L_\alpha$  term represents a natural image prior for some  $\alpha \leq 1$ .

To minimize Eq. (9), we adapt Krishnan and Fergus’s [9] approach for stationary non-blind deconvolution in the non-stationary case: after introducing the auxiliary variable  $v$  we alternately minimize

$$\min \|g - \mu \diamond f\|_2^2 + 2^t \|f - v\|_2^2 + \frac{1}{2000} \|v\|_{2/3}^{2/3} \quad (10)$$

in  $f$  and  $v$ . Note that the weight  $2^t$  increases from 1 to 256 during nine alternating updates in  $f$  and  $v$  for  $t = 0, 1, \dots, 8$ . Choosing  $\alpha = 2/3$  allows an analytical formula for the update in  $v$ , see [9] for details.

## 4.3. GPU implementation

The algorithm detailed above lends itself to parallelization on a Graphics Processing Unit (GPU). We reimplemented all steps of the algorithm in PyCUDA, a Python wrapper to NVIDIA’s CUDA API. To evaluate the speed-up, we compared the run time of our MATLAB implementation on a single core of an Intel Core i5 against our PyCUDA version on a NVIDIA Tesla C2050 with three gigabytes of memory, running on a 2.4Ghz Intel Xeon. Tab. 1

shows that deblurring real images of realistic sizes can be performed about ten times faster on GPUs than on usual (single-core) processors.

## 5. Results

In this section, we show results on several challenging example images taken from the literature and do a comprehensive comparison against state-of-the-art algorithms for single image blind deblurring. We compare against both algorithms assuming uniform as well as non-uniform blur.

**Comparison against Whyte et al. [24]:** The example *Notre Dame* of Fig. 6 shows a picture with real camera shake taken from [24] and compares with Fergus et al. [4] who assume stationary blur and Whyte et al. [24] who model the motion blur as PMPB caused by rotations only. The image obtained by [24] exhibits much more detail compared to [4] which suggests that the camera motion during exposure involved a significant amount of rotational motion. While Whyte et al. [24] considers rotations (roll, pitch, yaw) for describing the motion blur, we took the basis of Gupta et al. [5] comprising of translations in x- and y-direction and in-plane rotations. It equally well captures the motion blur which is verified by the good restoration quality of our approach.

**Comparison against Gupta et al. [5]:** The *Magazines* example in Fig. 6 is an image taken from [5]. To compare against a state-of-the-art deblurring method assuming uniform blur, we applied Xu et al. [25], which however fails to find a meaningful kernel. In contrast, the non-stationary PMPB model of [5] is able to capture and remove the blur such that the result reveals much more detail. Although using the same basis (in-plane rotations and translations) as [5], we are able to improve image quality even further, evident by less artifacts and clearly visible in the closeups.

**Comparison against Harmeling et al. [6]:** Similarly, our results improve also recent work of Harmeling et al. [6], see *Elephant* example in Fig. 6. Especially in regions with little edge information (e.g. top region containing sky our camera motion constrained model improves kernel estimation as it allows globally consistent blur only while [6] estimate each kernel locally.

**Comparison against Joshi et al. [8]:** The *Coke* example in Fig. 6 is an interesting example, as [8] uses data from inertial measurement sensors to determine the PSF. In contrast, we are able to estimate the blur *blindly* without exploiting the additional sensor data and recover a sharp image with comparable if not superior quality. For comparison, we also show the result of [25] whose assumption of an invariant motion blur is again too restrictive to yield a good restoration result.



**Figure 5:** Direct Deconvolution with and without corrective weighting for the blurred image shown in Fig. 1. Note the artifacts stemming from improper treatment of overlapping parts which can be minimized by appropriate corrective weighting.

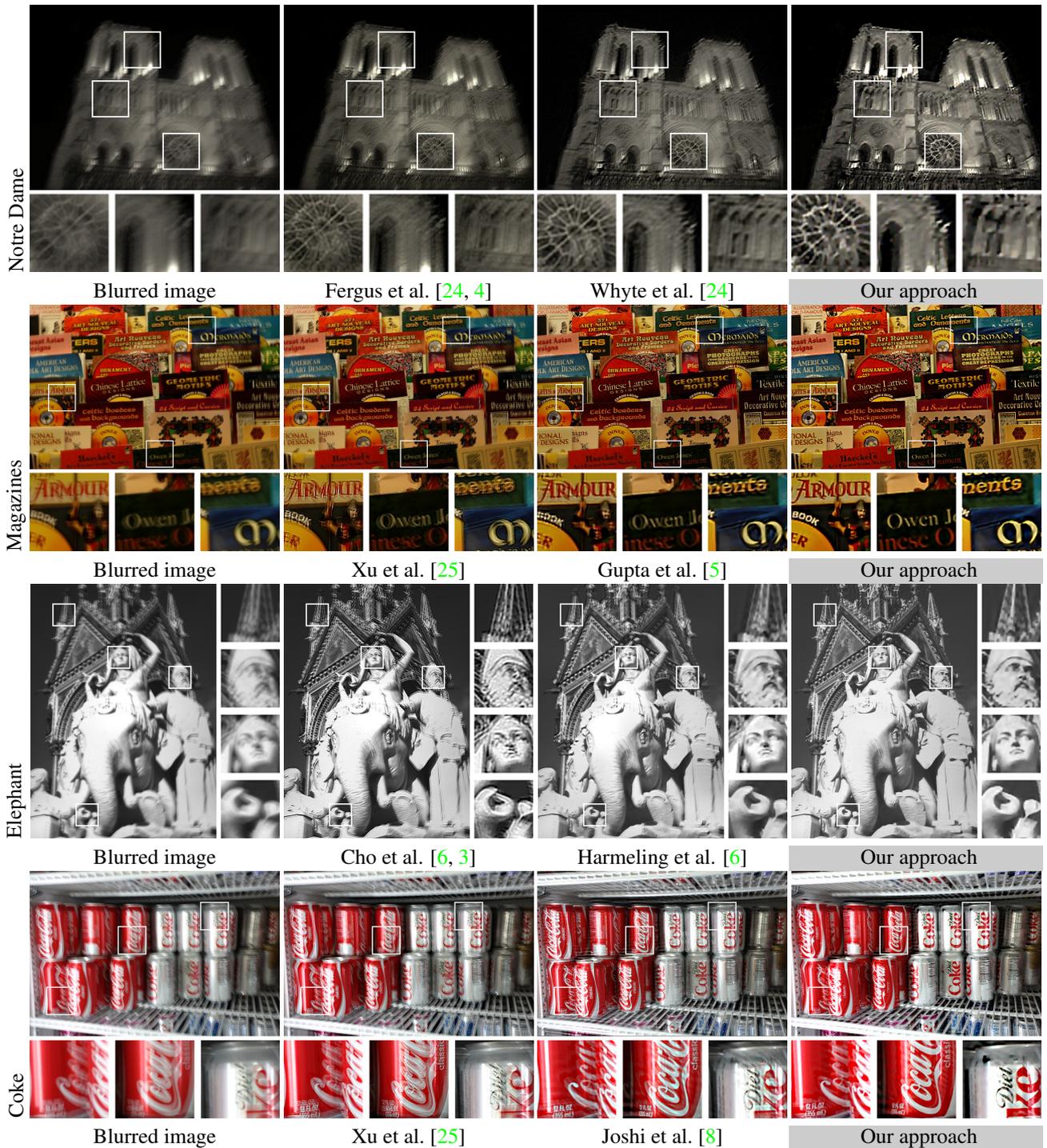
## 6. Conclusion

In this paper we proposed a single image blind deblurring algorithm for removing non-uniform motion blur due to camera shake. By combining the efficiency of the EFF and the camera motion constraints of PMPB we derive an algorithm that substantially enlarges the regime where hand held photographs can be taken.

One major limitation of our approach is that it does not deal with moving or deformable objects, or scenes with significant depth variation. A preprocessing step to separate moving objects or depth layers may be able to address this. Further limitations of our approach include the problem of pixel saturations and severe image noise, which are subject of ongoing research.

## References

- [1] J. Bardsley, S. Jeffries, J. Nagy, and B. Plemmons. A computational method for the restoration of images with an unknown, spatially-varying blur. *Optics Express*, 14(5):1767–1782, 2006. 2
- [2] S. Cho and S. Lee. Fast Motion Deblurring. *ACM Trans. Graph.*, 28(5), 2009. 1, 2, 4, 5
- [3] S. Cho, Y. Matsushita, and S. Lee. Removing non-uniform motion blur from images. In *Proc. Int. Conf. Comput. Vision*. IEEE, 2007. 8
- [4] R. Fergus, B. Singh, A. Hertzmann, S. Roweis, and W. Freeman. Removing camera shake from a single photograph. In *ACM Trans. Graph.* IEEE, 2006. 1, 2, 6, 8
- [5] A. Gupta, N. Joshi, L. Zitnick, M. Cohen, and B. Curless. Single image deblurring using motion density functions. In *Proc. 10th European Conf. Comput. Vision*. IEEE, 2010. 1, 2, 3, 4, 5, 6, 8
- [6] S. Harmeling, M. Hirsch, and B. Schölkopf. Space-variant single-image blind deconvolution for removing camera shake. In *Advances in Neural Inform. Processing Syst.* NIPS, 2010. 1, 2, 3, 6, 8
- [7] M. Hirsch, S. Sra, B. Schölkopf, and S. Harmeling. Efficient Filter Flow for Space-Variant Multiframe Blind Deconvolu-  
tion. In *Proc. Conf. Comput. Vision and Pattern Recognition*. IEEE, 2010. 1, 3, 5
- [8] N. Joshi, S. Kang, C. Zitnick, and R. Szeliski. Image deblurring using inertial measurement sensors. In *ACM Trans. Graph.* ACM, 2010. 2, 6, 8
- [9] D. Krishnan and R. Fergus. Fast image deconvolution using hyper-Laplacian priors. In *Advances in Neural Inform. Processing Syst.* NIPS, 2009. 6
- [10] D. Kundur and D. Hatzinakos. Blind image deconvolution. *Signal Processing Mag.*, 13(3):43–64, May 1996. 2
- [11] A. Levin. Blind motion deblurring using image statistics. In *Advances in Neural Inform. Processing Syst.* NIPS, 2006. 2
- [12] A. Levin, Y. Weiss, F. Durand, and W. Freeman. Understanding and evaluating blind deconvolution. In *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, 2009. 2
- [13] L. Lucy. An iterative technique for the rectification of observed distributions. *J. Astronomy*, 79:745–754, 1974. 2
- [14] J. Miskin and D. MacKay. Ensemble learning for blind image separation and deconvolution. In *Advances Independent Component Analysis*. Advances Independent Component Analysis, 2000. 2
- [15] J. Money and S. H. Kang. Total variation minimizing blind deconvolution with shock filter reference. *Image Vision Comput.*, 26(2):302–314, 2008. 4
- [16] S. Osher and L. Rudin. Feature-oriented image enhancement using shock filters. *SIAM J. Numerical Analysis*, 27(4):919–940, 1990. 4
- [17] R. Raskar, A. Agrawal, and J. Tumblin. Coded exposure photography: motion deblurring using fluttered shutter. *ACM Trans. Graph.*, page 804, 2006. 2
- [18] W. Richardson. Bayesian-based iterative method of image restoration. *J. Opt. Soc. of Am.*, 62(1), 1972. 2
- [19] Q. Shan, J. Jia, and A. Agarwala. High-quality motion deblurring from a single image. *ACM Trans. Graph.*, 2008. 1, 2, 5
- [20] Q. Shan, W. Xiong, and J. Jia. Rotational motion deblurring of a rigid object from a single image. In *Proc. Int. Conf. Comput. Vision*. IEEE, 2007. 2
- [21] Y. W. Tai, N. Kong, S. Lin, and S. Shin. Coded exposure imaging for projective motion deblurring. In *Proc. Conf. Comput. Vision and Pattern Recognition*. IEEE, 2010. 1



**Figure 6:** Comparison with state-of-the-art stationary and non-stationary deblurring algorithms on real-world data.

[22] Y. W. Tai, P. Tan, L. Gao, and M. S. Brown. Richardson-Lucy deblurring for scenes under projective motion path. Technical report, KAIST, 2009. 2

[23] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Int. Conf. Comput. Vision*, pages 839–846. IEEE, 2002. 4

[24] O. Whyte, J. Sivic, A. Zisserman, and J. Ponce. Non-uniform

deblurring for shaken images. In *Proc. Conf. Comput. Vision and Pattern Recognition*. IEEE, 2010. 1, 2, 3, 4, 6, 8

[25] L. Xu and J. Jia. Two-phase kernel estimation for robust motion deblurring. In *Proc. 10th European Conf. Comput. Vision*. IEEE, 2010. 1, 2, 5, 6, 8

[26] L. Yuan, J. Sun, L. Quan, and H.-Y. Shum. Image deblurring with blurred/noisy image pair. *ACM Trans. Graph.*, 2008. 2