Orthogonal parabolic exposures for motion deblurring

Taeg Sang Cho, *Student Member, IEEE,*
Anat Levin, Frédo Durand, and William. T. Freeman, *Fellow, IEEE*

Abstract—Relative motion between the camera and scene during exposure generates blur artifacts. Removing blur is challenging because one has to estimate the motion kernels, which can spatially vary over the image. Even if the motion is successfully identified, blur inversion can be unstable because the blur kernel attenuates high frequency image content. We present a computational camera to address these challenges. Our solution captures two images of a scene with a parabolic camera motion in two orthogonal directions. We show analytically that this strategy near-optimally preserves the image content of moving objects. This computational camera is the first camera to provide a performance guarantee for capturing 2D object motions. Taking two images of a scene also helps us estimate spatially varying object motions. We present a prototype camera and demonstrate successful motion deblurring on real world motions.

Index Terms—Computational photography, Motion deblurring, Blind deconvolution, Computational camera

1 Introduction

Relative motion between the camera and scene during exposure generates blur artifacts in photographs. Although motion blur can sometimes be desirable for artistic purposes, it often severely limits the image quality. While blur can be reduced using a shorter shutter speed,this comes with an unavoidable trade-off of increased noise. One source of motion blur is camera shake. We can mitigate the camera shake blur by using a mechanical image stabilization hardware or by placing the camera on a tripod during exposure. A second source of blur is a movement of objects in the scene. This type of blur is harder to control, therefore it is often desirable to remove such blur computationally using blur deconvolution.

Motion blur removal, often called motion deblurring or blind deconvolution, is challenging in two aspects. The first challenge is estimating blur kernels, or point-spread functions (PSF), from blurred images, which entails estimating the relative motion between the camera and scene. Blur estimation can be especially difficult if the scene is dynamic or if the camera undergoes a rotational motion, which would induce spatially variant blur. The second challenge is removing the blur to recover a blur-free image. Motion blur averages neighboring pixels, attenuating high spatial frequency information. Consequently, recovering a blur-free image is an ill-posed problem which needs to be addressed by deblurring systems or algorithms.

This paper provides a solution that addresses both challenges for a restricted class of a deblurring problem. We assume that the camera is placed on a tripod and the scene consists of objects moving at a constant speed in an arbitrary direction parallel to the image plane. This setup corresponds to a scenario in which the exposures are relatively short so that an arbitrary object motion can be approximated as a constant speed motion. Spatially variant blur induced by different object motions is therefore piecewise constant (in contrast to in-plane rotational camera motions that induce continuous spatially variant blur.) Our solution takes two successive images using a moving sensor: one moving in a horizontal parabolic displacement path and another moving in a vertical parabolic displacement path [7]. We recover one sharp image from the two input images by estimating the spatially variant blur (Sec. 4.2) and by deconvolving the input images using a multi-image deconvolution algorithm (Sec. 4.1). We empirically show that the image reconstruction error from blur kernel estimation is negligible (Sec. 4.2), addressing the first challenge of deblurring. We also analytically prove that our image capture strategy near-optimally minimizes the information loss for 2D constant-speed motions in arbitrary directions (Sec. 3), addressing the second challenge.

- *T. S. Cho, F. Durand, and W. T. Freeman are with Computer Science and Artificial Intelligence Lab (CSAIL), Massachusetts Institute of Technology, Cambridge, MA 02139.*
- *A. Levin is with the Weizmann Institute of Science, Israel.*

2 Related work

Some previous methods handle spatially variant blur by restricting the type of spatially variant blur [8], [9], [14], [15], [17], [24]: Levin [15] considers a piecewise constant spatially variant blur; Shan *et al.* [24] assume that the relative motion between the camera and the scene is purely rotational; Whyte *et al.* [27] and Gupta *et al.* [12] estimate a spatially variant blur by modeling camera shake as a rigid body motion. To aide spatially variant blur estimation, additional hardware could be used to record the relative movement between the camera and the scene during exposure, from which one can estimate the spatially variant blur [4], [26]. Levin *et al.* [17] introduce a new camera that makes the blur invariant to 1D subject motions. Users could assist spatially varying blur estimation by specifying blurred edges that should be sharp [13] or by specifying regions with different amount of blur [9]. Taking two images also helps estimate spatially variant blur [6], [25].

Most of aforementioned methods do not address information loss due to blur. Typical motion blur kernels correspond to box filters in the motion direction, therefore blurs attenuate high spatial frequencies and make the blur inversion ill-posed. One technique addressing this issue is a flutter shutter camera [21]. By opening and closing the shutter multiple times during exposure, one can significantly reduce the high frequency image information loss. Another method takes two images, each with different exposure lengths [28]. The short-exposure image contains high frequency information that supplements the missing information in the long-exposure, blurred image. Agrawal *et al.* [3] take multiple shots of a moving object, each with different exposures, and deconvolve the moving object using all the shots. The multi-shot strategy is beneficial because the information lost in one of the shots is captured by another. However, their strategy does not offer guarantees on the worst-case performance. Levin *et al.* [17] propose a parabolic motion camera to minimize the information loss for 1D constant velocity motions, but the solution is invalid if a 2D motion is present. Agrawal and Raskar [1] analyze the performance of a flutter-shutter camera and a parabolic camera and conclude that a flutter shutter camera performs better for handling a 2D constant velocity motion blur. Agrawal and Xu [2] introduce a new code for a flutter shutter camera with a better trade-off between blur estimation and information capture.

3 Sensor motion design and analysis

Consider an object moving at a constant velocity and let $s_{x,y} = [s_x, s_y]$ be its 2D velocity vector. Suppose we capture J images $B^1, ..B^J$ of this object using J translating cameras. Locally, the blur is a convolution:

$$B^j = k^j_{s_{x,y}} \otimes I + n^j \tag{1}$$

where I is an ideal sharp image, n^j imaging noise, and $k^j_{s_{x,y}}$ a blur kernel (point spread function, PSF). $k^j_{s_{x,y}}$ is a function of the relative motion between the sensor and the scene. The convolution is a multiplication in the frequency domain:

$$\hat{B}^j(\omega_{x,y}) = \hat{k}^j_{s_{x,y}}(\omega_{x,y})\hat{I}(\omega_{x,y}) + \hat{n}^j(\omega_{x,y}) \tag{2}$$

where $\omega_{x,y} = [\omega_x, \omega_y]$ is a 2D spatial frequency, and the $\hat{}$ indicates the Fourier transform of the corresponding signal.

To deblur images successfully, we need to increase the spectral content of blur kernels $\|\hat{k}^j_{s_{x,y}}(\omega_{x,y})\|^2$. Qualitatively, a deblurring algorithm divides the Fourier transform $\hat{B}^j$ of the image by that of the blur kernel $\hat{k}^j_{s_{x,y}}$ at every spatial frequency. If $\|\hat{k}^j_{s_{x,y}}(\omega_{x,y})\|^2$ is small for all cameras, the deblurring algorithm amplifies noise and degrades the quality of restored image. We show in Sec. 4.1 that the reconstruction performance of the Wiener filter deconvolution method is inversely related to the summed spectra:

$$\|\tilde{\hat{k}}_{s_{x,y}}(\omega_{x,y})\|^2 = \sum_j \|\hat{k}^j_{s_{x,y}}(\omega_{x,y})\|^2 \tag{3}$$

Therefore, we should maximize the joint spectrum $\|\tilde{\hat{k}}_{s_{x,y}}(\omega_{x,y})\|^2$ of an imaging device for every $\omega_{x,y}$ and for every $s_{x,y}$. This goal is formally stated as follows:

Given a time budget T, find a set of J camera motions that maximizes the minimum of the summed power spectrum $\|\tilde{\hat{k}}_{s_{x,y}}(\omega_{x,y})\|^2$ over every spatial frequency $\omega_{x,y}$ and every motion vector $\|s_{x,y}\| < S_{obj}$.

We introduce the *first* solution that provides the worst-case spectral power guarantee for 2D constant velocity motions. To prove our claim, we start with a brief review of space time motion blur analysis. We show that a set of PSFs for all 2D constant velocity motions $\|s_{x,y}\| < S_{obj}$ occupies the complementary volume of an inverted double cone in the Fourier domain, and that the camera motion design can be formulated as maximizing the spectral content in this volume. We show analytically that the best worst-case spectral coverage of any camera motions is bounded and that our design approaches the bound up to a constant multiplicative factor.

3.1 Motion blur in the space-time volume

We represent light received by the sensor as a 3D space-time volume $L(x,y,t)$. That is, $L(x,y,t)$ denotes the light ray hitting the x,y coordinate of a static detector at a time instance t. A static camera forms an image by integrating these light rays over a finite exposure time T:

$$B(x,y) = \int_{-\frac{T}{2}}^{\frac{T}{2}} L(x,y,t)dt \tag{4}$$

Assume the camera is translating during exposure on the x–y plane, and let $\mathbf{f}$ be its displacement path:

$$\left\{\mathbf{f} : [x,y,t] = [f_x(t), f_y(t), t]\right\} \tag{5}$$

Then the rays hitting the detector are spatially shifted:

$$B(x,y) = \int_{-\frac{T}{2}}^{\frac{T}{2}} L\left(x + f_x(t), y + f_y(t), t\right) dt + n \tag{6}$$

where n is imaging noise. For example, for a static camera the integration curve is a vertical straight line $f_x(t) = f_y(t) = 0$ (Fig. 1(a)). The integration curve of a camera moving at a constant velocity is a slanted line $f_x(t) = s_x t,\ f_y(t) = s_y t$ (Fig. 1(c)). For horizontal parabolic motion, $f_x(t) = at^2,\ f_y(t) = 0$ and for a vertical parabola $f_x(t) = 0,\ f_y(t) = at^2$ (Fig. 1(d-e)). We can represent the integration curve $\mathbf{f}$ as a 3D integration kernel ϕ:

$$\phi(x,y,t) = \delta(x - f_x(t)) \cdot \delta(y - f_y(t)) \tag{7}$$

where δ is a Dirac delta function.

If an object motion is locally constant, we can express the

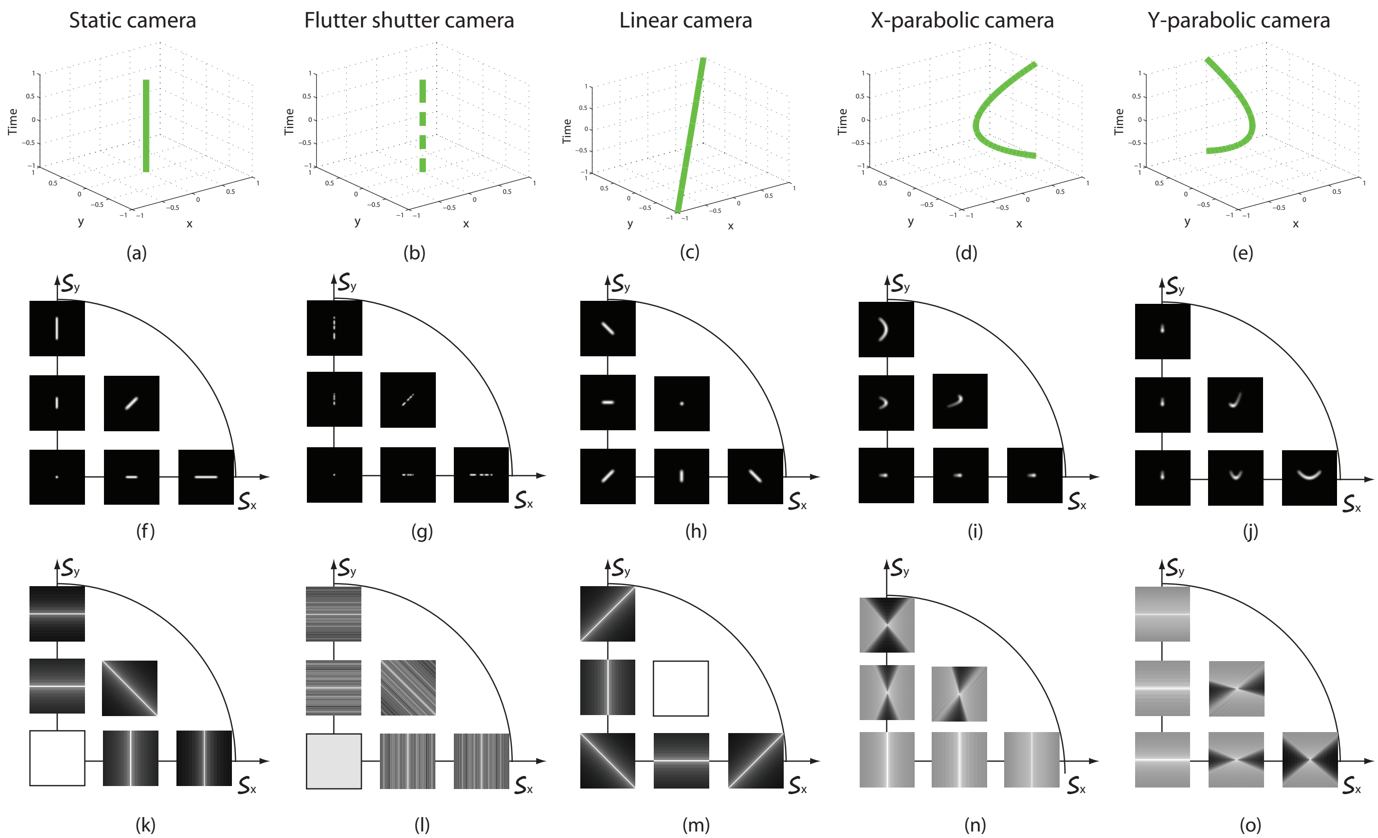


Fig. 1: *The integration curves ϕ (a-e), the point spread functions $k_{s_{x,y}}$ (f-j) and their log-power spectra (k-o) for a few cameras. In (f-o), the outer axes correspond to x,y directional speed. In (f-j), the inner axes correspond to x–y coordinates, and in the spectra plots (k-o), the inner axes correspond to ω_x–ω_y coordinates. All spectra plots are normalized to the same scale.*

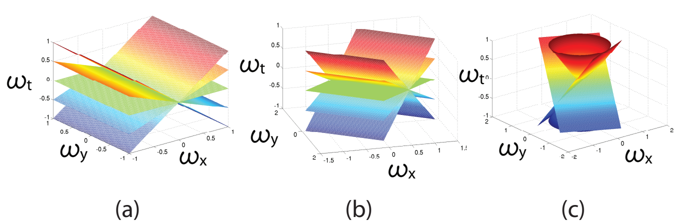


Fig. 2: *(a) The union of horizontal motion PSF slices at all velocities $\hat{k}_s$ forms a 3D double wedge. (b) A union of diagonal motion PSF slices forms a rotated 3D double wedge. (c) The spectra of all 2D constant velocity PSF slices comprise a wedge of revolution.*

integrated image as a convolution of a sharp image at one time instance (e.g. $L(x,y,0)$) with a point spread function $k_{s_{x,y}}$. The PSF $k_{s_{x,y}}$ of a constant velocity motion $s_{x,y} = [s_x, s_y]$ is a sheared projection of the 3D integration kernel ϕ:

$$k_{s_{x,y}}(x,y) = \int_t \phi(x - s_x t, y - s_y t, t)dt \tag{8}$$

Some PSFs of different integration kernels are shown in the second row of Fig. 1.

The Fourier transform $\hat{k}_{s_{x,y}}$ of the PSF $k_{s_{x,y}}$ is a slice from the Fourier transform $\hat{\phi}$ of the integration kernel ϕ [17], [19]:

$$\hat{k}_{s_{x,y}}(\omega_x, \omega_y) = \hat{\phi}(\omega_x, \omega_y, s_x\omega_x + s_y\omega_y) \tag{9}$$

$\hat{k}_{s_{x,y}}$ for several object motions for different integration kernels ϕ are shown in the bottom row of Fig. 1. Fig. 2(a) shows Fourier slices corresponding to horizontal object motions at varying velocities, the case considered in [17]. Slices occupy a 3D double wedge. When the motion direction changes (e.g. $s_x = s_y$ in Fig. 2(b)), slices occupy a rotated 3D double wedge. In general, 2D Fourier slices corresponding to all motion directions $\|s_{x,y}\| < S_{obj}$ lie in the complementary volume of an inverted double cone (Fig. 2(c)). We refer to this volume as *a wedge of revolution*, defined as a set:

$$C \equiv \{(\omega_x, \omega_y, \omega_t) | \omega_t < S_{obj}\|\omega_{x,y}\|\} \tag{10}$$

To see this, note that the Fourier transform of a PSF is a slice from $\hat{\phi}$ at $\omega_t = s_x\omega_x + s_y\omega_y$, and if $\|s_{x,y}\| \leq S_{obj}$, $s_x\omega_x + s_y\omega_y \leq S_{obj}\|\omega_{x,y}\|$.

3.1.1 Bounding spectral content

Suppose we capture J images of a scene and let $\|\tilde{\hat{\phi}}\|^2$ be the joint power spectrum $\|\tilde{\hat{\phi}}(\omega_x, \omega_y, \omega_t)\|^2 = \sum_j^J \|\hat{\phi}^j(\omega_x, \omega_y, \omega_t)\|^2$. As mentioned earlier, our goal is to design a set of camera motions that maximizes the joint kernel spectrum $\|\tilde{\hat{k}}_{s_{x,y}}\|^2$ (Eq. 3) for all object motions $\|s_{x,y}\| < S_{obj}$. Since PSFs of all bounded 2D linear motions occupy the wedge of revolution (Eq. 10), designing PSFs with high spectral power for all $s_{x,y} < S_{obj}$ is equivalent to maximizing the spectral content of $\|\tilde{\hat{\phi}}\|^2$ within the wedge of revolution.

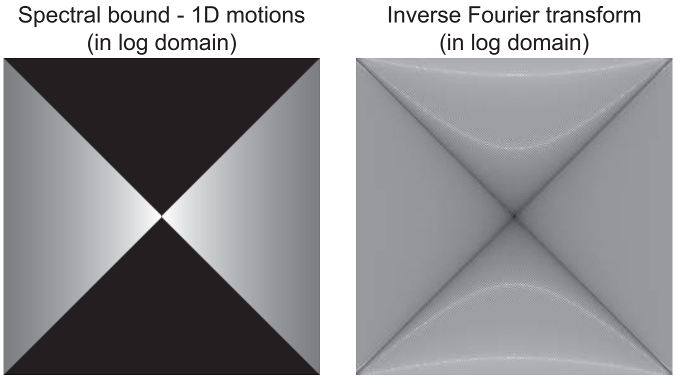


Fig. 3: *We explicitly invert the spectral bound for 1D motions to illustrate that the explicit inversion of the spectral bound in Eq. 12 does not result in a physically realizable motion of the form in Eq. 7. Both the spectrum and the motion (i.e. the inverse Fourier transform) are shown in the log-domain.*

We can derive an upper bound on the worst-case spectral content of any camera motions. The amount of photon energy collected by a camera within a fixed exposure time T is bounded. Therefore, by Parseval's theorem, the norm of every ω_{x_0,y_0} slice of $\hat{\tilde{\phi}}$ (i.e. $\hat{\tilde{\phi}}(\omega_{x_0},\omega_{y_0},\omega_t)$) is bounded [17]:

$$\int \|\hat{\tilde{\phi}}(\omega_{x_0},\omega_{y_0},\omega_t)\|^2 d\omega_t \leq T \tag{11}$$

Every ω_{x_0,y_0}-slice intersects the wedge of revolution for a segment of length $2S_{obj}\|\omega_{x_0,y_0}\|$. To maximize the worst-case spectral power, the optimal camera would spread the captured energy uniformly in this intersection. Therefore, we can derive an upper bound on the worst-case spectral power by dividing the captured energy by the segment length:

$$\min_{\omega_t} \|\hat{\tilde{\phi}}(\omega_{x_0},\omega_{y_0},\omega_t)\|^2 \leq \frac{T}{2S_{obj}\|\omega_{x_0,y_0}\|} \ . \tag{12}$$

Since the PSFs spectra $\hat{k}^j_{s_{x,y}}$ are slices through $\hat{\phi}^j$, this bound also applies for the PSFs' spectral power:

$$\min_{s_{x,y}} \|\hat{\tilde{k}}_{s_{x,y}}(\omega_{x_0},\omega_{y_0})\|^2 \leq \frac{T}{2S_{obj}\|\omega_{x_0,y_0}\|} \ . \tag{13}$$

The optimal bound Eq. 12 applies to any types of integration kernel ϕ regardless of the number of shots taken during the time budget T.

3.2 Orthogonal parabolic motions

We seek a motion path whose spectrum covers the wedge of revolution and approaches the bound in Eq. 12. We also seek to cover the spectrum with the fewest images, because as we take more images within the time budget, the delay between subsequent shots reduces the effective time budget, degrading the spectral performance.

We could compute the optimal camera motion by inverting the Fourier transform of the bound in Eq. 12. However, the inverse Fourier transform of this bound is not a physically valid motion of the form in Eq. 7. To illustrate this, we invert the bound for 1D motions [17] in Fig. 3[1]. We can see that the corresponding optimal motion in the spatial domain is not a realizable motion: the inverse Fourier transform is dense in the spatial domain and contains negative pixel values. If we invert the bound for 2D motions in Eq. 12, we observe the same phenomenon in 3D: the optimal path is not a realizable motion.

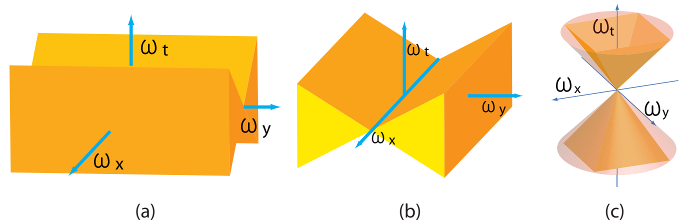


Fig. 4: *(a) The spectrum $\hat{\phi}_1$ captured by a x-parabolic camera. (b) The spectrum $\hat{\phi}_2$ captured by a y-parabolic camera. (c) The sum of spectra captured by the two orthogonal parabolic cameras approximates the wedge of revolution.*

Our solution captures two images of a scene with two orthogonal parabolic motions. We show analytically that the orthogonal parabolic motions capture the wedge of revolution with the worst-case spectral power greater than $2^{-1.5}$ of the upper bound.

3.2.1 Camera motion

Let ϕ_1, ϕ_2 be the 3D integration kernels of x and y parabolic camera motions. The kernels are defined by the integration curves f_1, f_2:

$$\begin{aligned} f_1(t) &= [a_x(t+T/4)^2, 0, t], \quad t=[-T/2 \ldots 0] \\ f_2(t) &= [0, a_y(t-T/4)^2, t], \quad t=[0 \ldots T/2] \end{aligned} \tag{14}$$

At time t, the derivative of the x-parabolic camera motion $f_1(t)$ is $2a_x(t-T/4)$, therefore the camera essentially tracks an object moving with velocity $2a_x(t-T/4)$ along x axis. During exposure, the x-parabolic camera tracks *once* every moving object with velocity within the range $[-2a_xT/4 \ldots 2a_xT/4]$. Similarly, the y-parabolic camera covers the velocity range $[-2a_yT/4 \ldots 2a_yT/4]$. For the reason that will be clarified below, we set

$$a_x = a_y = \frac{2\sqrt{2}S_{obj}}{T} \tag{15}$$

The maximal velocity of the sensor becomes $S_{sens} = \sqrt{2}S_{obj}$. That is, the velocity range covered by these parabolas is $[-S_{sens} \ldots S_{sens}]$.

Fig. 1(i-j) show PSFs of different object motions captured by the orthogonal parabolic camera. PSFs take the form of a truncated and sheared parabola that depends on the object speed.

1. The bound for 1D motions in the Fourier space is the slice of the wedge of revolution on the $\omega_x - \omega_t$ plane.

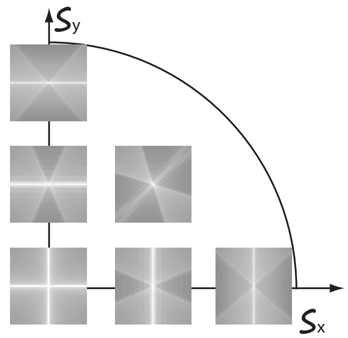


Fig. 5: *The summed spectrum coverage of the two orthogonal parabolic motions for different object velocities $s_{x,y}$. While each parabolic motion has zeros in a range of spatial frequencies (see Fig. 1(n-o)), their summed spectrum does not have zeros in any spatial frequencies. The log-spectrum plots in this figure are normalized to the same scale as that of the log-spectrum plots in Fig. 1(k-o).*

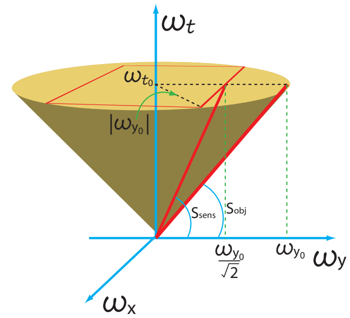


Fig. 6: *The joint spectrum of orthogonal parabolic motions subsumes the wedge of revolution if $S_{sens} \geq \sqrt{2}S_{obj}$.*

3.2.2 Optimality

As mentioned earlier, to make the blur easily invertible, we want to maximize the spectral power of the camera motion paths within the wedge of revolution (Eq. 10). We show that the orthogonal parabolic motions capture the wedge of revolution with the worst-case spectral power greater than $2^{-1.5}$ of the optimal bound in Eq. 13.

We first derive the joint spectral coverage $\|\hat{\tilde{\phi}}\|^2$ of the two orthogonal parabolic motions. Levin *et al.* [17] show that a parabolic motion's spectrum is approximately a double wedge. Since a x-parabolic motion ϕ_1 is a Dirac delta along the y axis, the 3D kernel spectrum $\|\hat{\phi}_1\|^2$ spreads energy in a 3D double wedge and is constant along the ω_y axis (Fig. 4(a)). The y-parabolic motion spreads energy in the orthogonal 3D double wedge (Fig. 4(b)). Mathematically speaking,

$$\begin{aligned}\|\hat{\phi}_1(\omega_x,\omega_y,\omega_t)\|^2 &\approx \frac{T}{4S_{sens}\|\omega_x\|}H(S_{sens}\|\omega_x\|-\|\omega_t\|)\\ \|\hat{\phi}_2(\omega_x,\omega_y,\omega_t)\|^2 &\approx \frac{T}{4S_{sens}\|\omega_y\|}H(S_{sens}\|\omega_y\|-\|\omega_t\|)\end{aligned} \tag{16}$$

where $H(\cdot)$ is a Heaviside step function.

The 2D PSF spectra are slices from the 3D double wedge $\|\hat{\phi}_j\|^2$. Fig. 1 (n-o) show the log-spectrum of PSFs $\hat{k}_s^j$ for parabolic exposures as we sweep the object velocity. For x-directional motions ($s_y = 0$), the x-parabolic camera covers all spatial frequencies without zeros. This agrees with the 1D optimality argument in Levin *et al.* [17]. However, as y-directional motion increases, the x-parabolic camera fails to capture a double wedge of frequencies near the ω_y axis. In other words, the x-parabolic camera misses spectral contents in the presence of a y-directional motion, and the blur inversion is unstable. The y-parabolic camera, however, covers the frequencies missed by the x-parabolic camera, therefore the *sum* of these two spectra (Fig. 5) does not have any zeros in any spatial frequencies. Therefore, by taking two orthogonal parabolic exposures, we can reliably invert the blur for all 2D object motions.

Fig. 4(c) visualizes the joint spectrum covered by the orthogonal parabolic motions, suggesting that the sum of orthogonal 3D wedges is an approximation of the wedge of revolution. In fact, the sum of double wedges subsumes the wedge of revolution if the maximal sensor speed S_{sens} is set to $\sqrt{2}S_{obj}$.

Claim 1: Let S_{sens} be the maximum sensor speed of the parabolic camera, and S_{obj} be the maximum object speed. If $S_{sens} \geq \sqrt{2}S_{obj}$, the joint power spectrum $\|\hat{\tilde{\phi}}\|^2$ of an orthogonal parabolic camera subsumes the wedge of revolution. When $S_{sens} = \sqrt{2}S_{obj}$, the worst-case spectral power of the orthogonal parabolic camera, at any frequency, is at least $\frac{1}{2\sqrt{2}}$ of the optimal bound.

Proof: The power spectrum of each parabolic camera is given in Eq. 16. The joint power spectrum of the orthogonal parabolic camera is non-zero in the set $\{(\omega_x,\omega_y,\omega_t)|\omega_t \leq S_{sens}\max(\|\omega_x\|,\|\omega_y\|)\}$. If $(\omega_x,\omega_y,\omega_t)$ lies in the wedge of revolution, then $\omega_t \leq S_{obj}\|\omega_{x,y}\|$. Since $\|\omega_{x,y}\|^2 \leq 2\max(\|\omega_x\|^2,\|\omega_y\|^2)$,

$$\begin{aligned}\omega_t &\leq S_{obj}\|\omega_{x,y}\|\\ &\leq \sqrt{2}S_{obj}\max(\|\omega_x\|,\|\omega_y\|)\\ &\leq S_{sens}\max(\|\omega_x\|,\|\omega_y\|)\end{aligned} \tag{17}$$

In other words, the joint spectrum of the orthogonal parabolic cameras subsumes the wedge of revolution. This is illustrated in Fig. 6.

The spectral power of the joint spectrum at $(\omega_x,\omega_y,\omega_t)$ is at least the minimum of the 3D wedge spectra:

$$\min\left(\frac{T}{4S_{sens}\|\omega_x\|}+\frac{T}{4S_{sens}\|\omega_y\|}\right) \tag{18}$$

Since $\|\omega_{x,y}\| \geq max(\|\omega_x\|,\|\omega_y\|)$,

$$\begin{aligned}\min\left(\frac{T}{4S_{sens}\|\omega_x\|}+\frac{T}{4S_{sens}\|\omega_y\|}\right) &\geq \frac{T}{4S_{sens}\|\omega_{x,y}\|}\\ &= \frac{T}{4\sqrt{2}S_{obj}\|\omega_{x,y}\|}\end{aligned} \tag{19}$$

Therefore, the worst-case spectral power of the orthogonal parabolic camera is at least $2^{-1.5}$ of the upper bound. ∎

Fig. 5 shows the log spectrum of the orthogonal parabolic cameras. Zeros present in one camera are compensated by its orthogonal counterpart for all object motions. At each velocity $s_{x,y}$, information for some spatial frequencies is better preserved than others, but Claim 1 guarantees that at each frequency, the spectral content is at least $2^{-1.5}$ of the optimal bound.

3.2.3 Discussions

The orthogonal parabolic camera deblurs diagonally moving objects better than objects moving along the camera motion axis because x-parabolic and y-parabolic shots both capture information from diagonally moving objects. Note that if we know before the image capture that object motions are primarily x-directional, one could increase the exposure length of the x parabolic shot to improve the deblurring performance in expectation.

The spectral bound in Eq. 19 assumes that the image information at each spatial frequency is independent. Therefore, our bound holds only if the restoration method treats each spatial frequency as independent. One such restoration method is the Wiener filter (introduced in Eq. 30) that imposes a Gaussian prior on image gradients. In a strict sense, the use of a non-linear image reconstruction algorithm would require a different analysis method, which takes into account correlations between different spatial frequencies. However, our framework still provides a concrete construction for comparing different camera designs, which we present below.

3.3 Discussion of other cameras

We compare the performance of the orthogonal parabolic camera to those of other designs available in literature.

3.3.1 A static camera

The integration kernel of a static camera is $\phi^{static}(t) = [0,0,t]$, for $t \in [-T/2 ... T/2]$ (Fig. 1(a)). Since the integration curve does not vary along the x or y axis, the power spectrum is constant along ω_x and ω_y:

$$\|\hat{\phi}^{static}(\omega_x, \omega_y, \omega_t)\|^2 = T^2 sinc^2(\omega_t T) \tag{20}$$

The Fourier transform of the PSF is a slice of the 3D spectrum $\hat{\phi}$, and is a sinc whose width depends on the object velocity $\|\hat{k}^{static}_{s_{x,y}}\|^2 = T^2 sinc^2((s_x\omega_x + s_y\omega_y)T)$. For fast object motions this sinc highly attenuates high frequencies. In fact, if the object motion is fast it is better to reduce the exposure time (this increases the width of the sinc) despite reducing the total amount of energy collected.

3.3.2 A flutter shutter camera

In a flutter shutter camera [21], the integration curve of a static camera is temporally modulated (Fig. 1(b)). Therefore, the spectrum of the integration curve $\phi^{flutter}$ is constant along ω_x, ω_y and is modulated along ω_t:

$$\|\hat{\phi}^{flutter}(\omega_x, \omega_y, \omega_t)\|^2 = \|\hat{m}(\omega_t)\|^2 \tag{21}$$

where $\hat{m}$ is the Fourier transform of the shutter code. This code can be designed to be more broadband than the sinc function in a static camera. However, the spectrum is constant along ω_x, ω_y. Therefore, the worst-case spectral performance is bounded as follows:

$$\min_s \|\hat{k}^{flutter}_s(\omega_x, \omega_y)\|^2 = T/(2S_{obj}\Omega) \tag{22}$$

for all (ω_x, ω_y) [17], where Ω is the spatial bandwidth of the camera. As a result, the flutter shutter poorly captures the low frequency image contents.

3.3.3 A linearly moving camera

If the camera is moving at a constant velocity (Fig. 1 third column), the integration curve is a slanted straight line $\phi^{linear}(t) = [s_x t, s_y t, t]$ (Fig. 1(c)). By linearly moving the camera, we can track the object that moves at the camera's speed, but we still suffer from a sinc fall-off for objects whose velocities are different from the camera's velocity.

3.3.4 A parabolic camera with a single exposure

Blur kernels from a single-exposure parabolic camera are invariant to 1D constant-velocity motions, and can be shown to approach the optimal bound for a set of 1D linear motions with bounded speed [17]. A single parabolic camera, however, is neither motion invariant nor optimal for 2D motions. When an object moves in a direction orthogonal to the camera movement axis (i.e. a x-parabolic camera imaging an object moving in the y direction), the spectral coverage along the orthogonal frequencies (i.e. ω_y) is poor. We have shown in Fig. 1 (n-o) several Fourier slices in which the captured spectra contain zeros.

3.3.5 A camera with parametric motions

We design other cameras with parametric motions and analyze their performance. Although we cannot analytically derive the spectral performance of each camera, we can compare each design numerically. We define four cameras: (i) a camera with a raised-cosine motion, (ii) a camera with a circular motion with a constant angular speed, (iii) a camera with a circular motion with a constant angular acceleration, and (iv) a camera with a spiral motion. Each camera moves in the defined camera path ϕ during the exposure $t \in [-T/2 ... T/2]$.

(i) A camera with a raised-cosine motion: The parametric motion for a raised-cosine motion is:

$$\{\phi : [x, y, t] = [\alpha(1 + cos(\omega t)), 0, t]\} \tag{23}$$

where $\omega = \pi/(T/2)$ and $\alpha = S_{obj}/\omega$. This camera moves in 1D and covers each velocity *twice* as opposed to once as in a parabolic camera. Fig. 7(b) shows the blur kernel spectra for

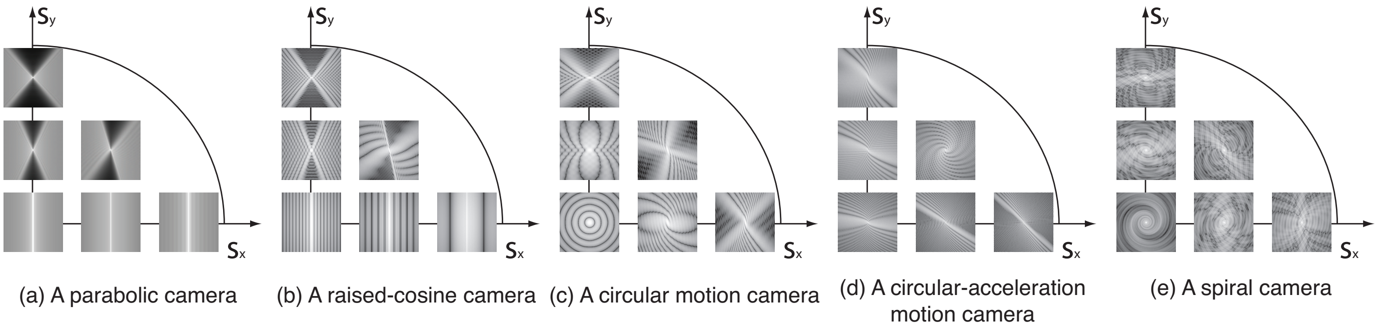


Fig. 7: *We numerically analyze the spectral performance of five different cameras: (a) a camera with a parabolic motion, (b) a camera with a raised-cosine motion, (c) a camera with a circular motion with a constant angular speed, (d) a camera with a circular motion with a constant angular acceleration, and (e) a camera with a spiral motion. This figure shows that even a two-image solution of cameras (b-e) cannot capture all frequencies without zeros for all object motions, as a two-image solution of a parabolic camera (a) does.*

different object motions. The zero pattern is quite similar to that of a parabolic camera in Fig. 7(a), but zeros also appear in frequencies that are well covered by a parabolic camera. This observation suggests that even a two-image solution of a raised-cosine camera cannot cover all frequencies without zeros, as a pair of orthogonally-moving parabolic camera does.

(ii) A camera with a circular motion with a constant angular speed: We can define the motion of this camera as:

$$\{\phi : [x,y,t] = [\alpha\cos(\omega t), \alpha\sin(\omega t), t]\} \quad (24)$$

where $\omega = \pi/(T/2)$ and $\alpha = S_{obj}/(2\omega)$. The camera sensor moves along a circular path at a predetermined speed ($S_{obj}/2$), therefore the camera essentially tracks each object motion with this particular speed in all possible orientations. However, this camera fails to capture other motions, and consequently we observe many zeros in blur kernel spectra, as shown in Fig. 7(c).

(iii) A camera with a circular motion with a constant angular acceleration: We can modify the above camera to track each object speed once, each in different orientations. The idea is to move the camera circularly but at a constant angular acceleration:

$$\left\{\phi : [x,y,t] = [\alpha\cos(\omega(t+T/2)^2), \alpha\sin(\omega(t+T/2)^2), t]\right\} \quad (25)$$

where $\omega = 2\pi/(T)^2$ and $\alpha = S_{obj}/(2\omega T)$. While this camera performs well for many object velocities, the blur spectra still contain many zeros due to phase coupling, as shown in Fig. 7(d).

(iv) A camera with a spiral motion:

$$\{\phi : [x,y,t] = [\alpha t\cos(\omega t), \alpha t\sin(\omega t), t]\} \quad (26)$$

where $\omega = k\pi/(T/2)$ and $\alpha = S_{obj}/\sqrt{(1+\omega^2T^2/4)}$. k determines the number of "spirals" during exposure. Here, we set $k = 3$. This camera tracks each speed once during exposure, but not in all directions. Fig. 7(e) shows that blur kernels for different object velocities contain substantial amount of zeros.

3.3.6 *Two shots*

Taking two images with cameras defined above can simplify the kernel estimation task, but it does not substantially enhance the spectral coverage of these cameras. Optimizing the exposure lengths of each shot [3], and in the case of a flutter shutter camera also optimizing the random codes in each shot, do not eliminate their fundamental limitations: their power spectra are constant along $\omega_{x,y}$ and hence they spend the energy budget outside the wedge of revolution. Previous two-image solutions to deblurring, such as [5], [6], [22], [28], fall into the category of taking two images with a static or a linearly moving camera. These methods can correctly find the motion kernels, but the image reconstruction quality is limited since the spectrum coverage is low.

3.3.7 *Synthetic simulation*

We compare the deblurring performance of (i) a pair of static cameras, (ii) a pair of flutter shutter cameras, (iii) a single parabolic camera and (iv) an orthogonal parabolic camera through simulations (Fig. 8). For all cameras, we fix the total exposure time T and assume that the object motion is known. The orthogonal parabolic camera is setup to deblur objects moving at speed less than S_{obj}. To give previous solutions the favor of doubt, we optimized their parameters for *each* motion independently: for a pair of static camera, we use the optimal split of the exposure time T into two shots; for a pair of flutter shutter camera, we use the optimal split of the exposure time T *and* the optimal code combinations. In a realistic scenario we cannot optimize the split of the exposure time T nor flutter-shutter codes because the object motion is not known a priori.

We render images of a moving object seen by these cameras: zero-mean Gaussian noise with standard deviation $\eta = 0.01$ is added to the rendered blurry images. We deblur rendered images using the Wiener deconvolution and compare the reconstruction performance. Fig. 8 shows deconvolution results and their peak signal-to-noise ratio (PSNR). Each row corresponds to a different object velocity. When the object is static, a pair of static camera restores visually the most pleasing deconvolution result. This is intuitively satisfying

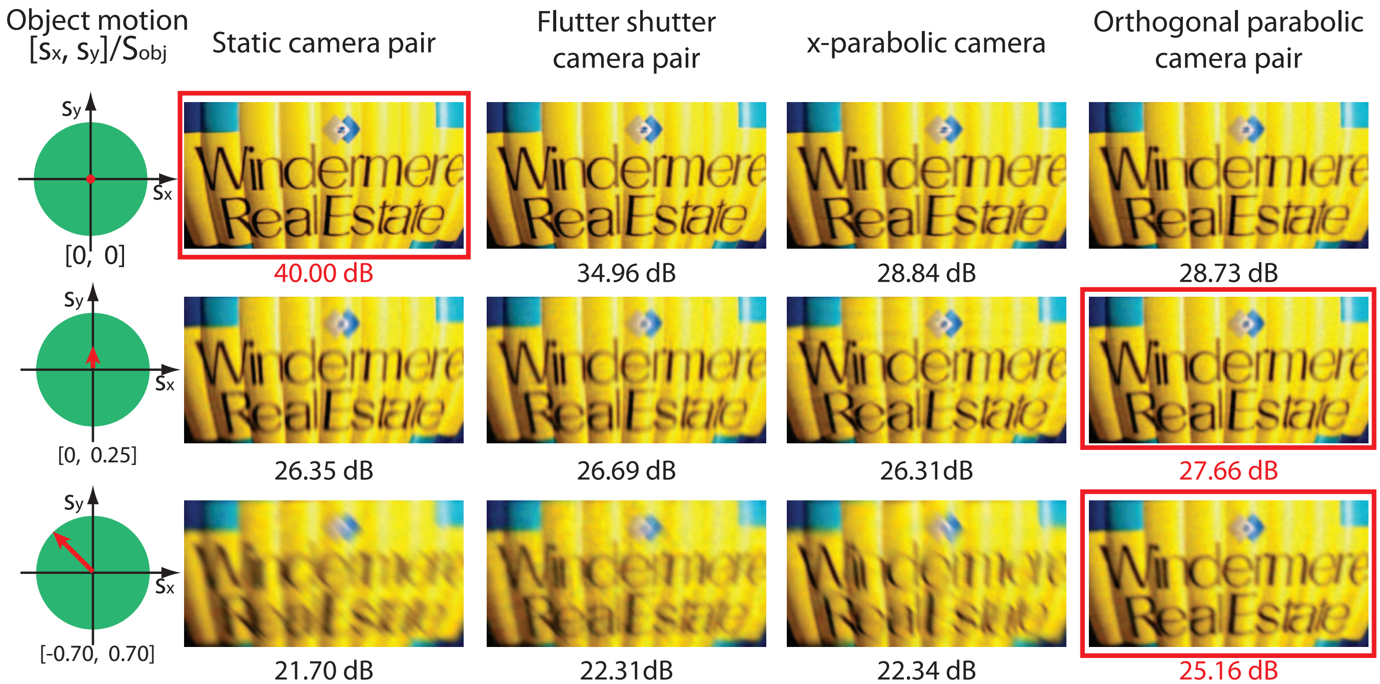


Fig. 8: *Synthetic visualizations of the reconstruction quality. We optimized the exposure lengths of each camera. First column: The object motion during the exposure. The green disc denotes the velocity range covered by the orthogonal parabolic camera, and the red arrow denotes the object velocity. Other columns show images deconvolved using the Wiener filter (Eq. 30). The orthogonal parabolic camera outperforms the other optimized solutions in deblurring the moving object.*

since the static camera is optimized for static object motions. The image quality from a flutter shutter camera is slightly worse than that of a static camera due to the loss of light. For moving objects, the orthogonal parabolic camera restores visually the most pleasing deconvolution results. While the orthogonal parabolic camera deblurs moving objects better than other cameras, its performance degrades as the object moves faster. However, the worst-case spectral performance across all velocities $s_{x,y}$ of interest is at least $2^{-1.5}$ of the optimal bound.

We put the synthetic experiment in the context of previous blur removal techniques. The performance of previous two-image motion deblurring techniques, such as [5], [6], [22], [28] can be approximated by the deconvolution result of the static camera pair in Fig. 8. Even if these solutions correctly estimate the motion, inverting the blur kernel is still hard since high frequencies are attenuated. Blind motion deblurring solutions, such as [10], [23], attempt to resolve an even harder problem, since they estimate the blur kernel from a single input image.

4 Image Reconstruction

We review a Bayesian method for image deconvolution and kernel estimation, and extend the result to accomodate two input images. We derive a closed form solution to estimate blur kernels from input images, and present an equivalent representation to estimate motion locally. Also, we experimentally show that an image reconstruction error due to kernel misclassification is small.

4.1 Non-blind deconvolution

A non-blind deconvolution algorithm recovers a blur-free image I from a blurry image B^j and an estimated blur kernel k^j. Let $\bar{B},\bar{k}$ be $\bar{B}=[B^1,B^2],\bar{k}=[k^1,k^2]$. We recover the blur-free image by maximizing the posterior probability. Using Bayes rule:

$$\begin{aligned}\tilde{I} &= \operatorname*{argmax}_I p(I|\bar{B},\bar{k}) \\ &\propto \operatorname*{argmax}_I p(I,\bar{B}|\bar{k}) \\ &= \operatorname*{argmax}_I p(I)\prod_{j=1}^{2} p(B^j|k^j,I)\end{aligned} \tag{27}$$

where we can define each term as follows:

$$\log p(B^j|k^j,I) = -\frac{1}{\eta^2}|B^j - k^j \otimes I|^2 + C_1 \tag{28}$$

$$\log p(I) = -\beta \sum_i \left(\rho(|g_{x,i}(I)|) + \rho(|g_{y,i}(I)|)\right) + C_2 \tag{29}$$

η^2 is the imaging noise variance, $\beta = 0.002$ controls the variance of the gradient profile, C_1, C_2 are constants, $g_{x,i}, g_{y,i}$ are x,y directional gradient operators at pixel i, and $\rho(z) = z^\alpha$ is a robust norm. When $\alpha = 2$, we impose a Gaussian prior on the image gradients, and when $\alpha \leq 1$, we impose a sparse prior.

Eq. 27 is essentially a joint deconvolution model, stating that we seek an image $\tilde{I}$ that fits the convolution constraints of both B^1 and B^2. In other words, the deconvolved image $\tilde{I}$ should be able to synthesize the input images B^1 and B^2 using the pair of kernels that reconstructed $\tilde{I}$. Although not presented in Bayesian terms, Rav-Acha and Peleg [22] essentially deblur two input images by maximizing the likelihood term (Eq. 28), and Chen *et al.* [5] augment it with the prior term Eq. 29. Using a sparse prior leads to visually pleasing results with crisp edges, but it is worth considering a Gaussian prior because we can derive closed form solutions.

We can efficiently solve Eq. 27 using the Wiener filter (i.e. a Gaussian image prior) [11]:

$$\hat{\tilde{I}}(\omega_{x,y}) = \frac{\hat{\bar{k}}^*(\omega_{x,y})\hat{\bar{B}}(\omega_{x,y})}{\frac{1}{\eta^2}\|\hat{\bar{k}}(\omega_{x,y})\|^2+\sigma^{-2}(\omega_{x,y})}. \tag{30}$$

where * is a complex conjugate operator, $\sigma^2(\omega_{x,y})$ is the variance of the image prior in the frequency domain: $\sigma^{-2}(\omega_{x,y}) = \beta(\|\hat{G}_x\|^2+\|\hat{G}_y\|^2)$ where $\hat{G}_x, \hat{G}_y$ are the Fourier transform of derivative filters. We use the Wiener filter to restore images for kernel estimation, but use a sparse deconvolution to restore the final blur-free image.

We can explicitly compute the expected reconstruction error using a Gaussian image prior by taking expectation over the space of natural images and over image noise:

$$\mathbf{E}_{I,\mathbf{n}}\left[\|\tilde{I}-I\|^2\right] = \sum_{\omega_x}\sum_{\omega_y}\frac{\eta^2}{\sum_j\|\hat{k}^j_{s_{x,y}}(\omega_x,\omega_y)\|^2+\eta^2\sigma^{-2}(\omega_x,\omega_y)} \tag{31}$$

Eq. 31 highlights that the image reconstruction error decreases monotonically as the summed power spectrum $\sum_j\|\hat{k}^j_{s_{x,y}}(\omega_x,\omega_y)\|^2$ increases. This justifies our PSF design goal in Eq. 3.

4.2 Kernel estimation

A critical step in motion deblurring is estimating the correct blur kernel $\bar{k}$. For that we seek

$$\bar{k} = \underset{k}{\operatorname{argmax}}\, p(\bar{k}|\bar{B}) = \underset{k}{\operatorname{argmax}}\, p(\bar{B}|\bar{k})p(\bar{k}) \tag{32}$$

where $p(\bar{k})$ is a prior on blur kernels (which we assume uniform). We derive the likelihood $p(\bar{B}|\bar{k})$ by marginalizing over all latent images I:

$$p(\bar{B}|\bar{k}) = \int p(\bar{B},I|\bar{k})dI \tag{33}$$

where $p(\bar{B},I|\bar{k})$ is defined in Eq. 28,29. If the prior $p(I)$ is Gaussian, $p(\bar{B}|\bar{k})$ is also Gaussian. If $p(I)\sim\mathbb{N}(0,\sigma^2)$, we can evaluate $p(\bar{B}|\bar{k})$ explicitly in the Fourier domain:

$$\log p(\hat{\bar{B}}|\hat{\bar{k}}) = C_3 - \frac{1}{2N}\sum_{\omega_{x,y}}^{N}\left(\frac{\|\hat{B}^1\hat{k}^{2*}-\hat{B}^{2*}\hat{k}^1\|^2+\eta^2\sigma^{-2}\left(\|\hat{B}^1\|^2+\|\hat{B}^2\|^2\right)}{\|\hat{B}^1\|^2+\|\hat{B}^2\|^2+\eta^2\sigma^{-2}}\right) \tag{34}$$

We have omitted the dependence on $\omega_{x,y}$ for clarity. When there is only one observed image (i.e. $\hat{k}^2=0, \hat{B}^2=0$), Eq. 34 reduces to a zero-frequency test which favors kernels with similar zero patterns as that of the blurry image $\hat{B}^1$ [16].

Fig. 9: *An image is synthetically blurred, and is deconvolved using the correct blur kernel pair and an incorrect blur kernel pair. An incorrect blur kernel pair has a spatial shift incompatible with input images, leading to ghosting and ringing in the restored image, whereas the correct kernel pair restores a sharp, artifact free image.*

When there are two observed images, the difference term $\|\hat{B}^1\hat{k}^{2*}-\hat{B}^{2*}\hat{k}^1\|^2$ supplements the zero frequency test: this term favors a pair of kernels that satisfies the commutative property of convolution. This phase term drastically improves the reliability of the kernel estimation measure. While Rav-Acha and Peleg [22], Chen *et al.* [5] and Agrawal *et al.* [3] introduce kernel estimation methods that explicitly instantiate the commutative property of convolution, what we introduce here is a Bayesian kernel estimation method that balances the contribution of the commutative property of convolution and the image prior.

We can rewrite $\log p(\hat{\bar{B}}|\hat{\bar{k}})$ in an equivalent representation that is more attractive for computational reasons. This involves solving for the latent image $\tilde{I}$ using Eq. 27, and expressing $p(\bar{B}|\bar{k})$ as follows [2]:

$$\log p(\bar{B}|\bar{k}) = \log p(\tilde{I},\bar{B}|\bar{k}) + \tilde{\Psi} + C_4 \tag{35}$$

where $\tilde{\Psi} = \sum_\omega \log \Psi_\omega$, and $\Psi_\omega = \frac{1}{\eta^2}\sum_j\|\hat{k}^j_\omega\|^2+\sigma_\omega^{-2}$ is the variance of $p(\hat{\bar{B}}_\omega|\hat{\bar{k}}_\omega)$. This variance term plays a critical role in distinguishing Eq. 35 from a standard MAP score $p(I,\bar{k}|\bar{B})$ since Eq. 35 accounts for the overall probability volume around the mode and not only the mode itself [18].

Qualitatively, $\log p(\tilde{I},\bar{B}|\bar{k})$ penalizes kernel pairs $\bar{k}$ that restore an image $\tilde{I}$ which would not fit the convolution constraints in Eq. 28. To satisfy the convolution constraints, the kernel pair $\bar{k}$ should "undo" the blur present in input images $\bar{B}$ *and* respect the spatial shift between input images (i.e. satisfy the commutative property of convolution.) Fig. 9 shows a synthetic example. We blur a sharp image with a pair of blur kernels, and deconvolve the blurred images using the correct/incorrect kernel pair. When we use an incorrect kernel pair, we observe ghosting artifacts due to an incompatible spatial shift. Therefore, this image is unable to regenerate the input images, and $\log p(\tilde{I},\bar{B}|\bar{k})$ penalizes that. On the other

2. This is a Laplace approximation of $\log p(\hat{\bar{B}}|\hat{\bar{k}})$, which is equivalent to $\log p(\bar{B}|\bar{k})$ since $\log p(\hat{\bar{B}}|\hat{\bar{k}})$ is a Gaussian distribution.

Algorithm 1 `Blur kernel estimation at pixel` i

% Variable definitions
$\bar{B} \equiv$ Blurred input images.
$S_k \equiv$ A set of blur kernel candidates.
$i \equiv$ A pixel index.
$f(\bar{B}, \bar{k}) \equiv$ Eq. 36

$C \Leftarrow 0.15$ *% penalty for single-image explanations*
$\bar{B}_i \Leftarrow 15 \times 15$ window around the pixel i in $\bar{B}$.
% Compute the log-likelihood
for every kernel candidate $\bar{k} \in S_k$ **do**
  if $k^2 = 0$ **then**
    $\text{cost}(\bar{k}) \Leftarrow f(\bar{B}_i, \bar{k}) + C$ *% A single-image explanation*
  else
    $\text{cost}(\bar{k}) \Leftarrow f(\bar{B}_i, \bar{k})$
  end if
end for
% The kernel estimate maximizes the log-likelihood
Kernel estimate at i $\Leftarrow \arg\min \text{cost}(\bar{k})$

hand, ghosting artifacts are not visible when we use the correct kernel to deblur the input images.

Most PSF estimation algorithms [10], [23] are designed to estimate blur kernels that are uniform across the image, and are not well suited to subject motion blur because these algorithms search over the full space of possible motions. In our scenario, object motions are assumed to be constant velocity. Since constant velocity motions comprise only a small subset of general motions, we can constrain the motion search space (i.e. the blur kernel search space) to constant velocity motions. This subsequently reduces the kernel estimation error. In this work, we estimate $\bar{k}$ by evaluating the log likelihood Eq. 35 on a set of PSF pairs that correspond to discretized 2D constant velocity motions, and by choosing the pair with the highest log likelihood.

4.3 Local kernel estimation

If there are multiple motions in the scene, we need to locally estimate the motion. Let $\tilde{\bar{I}}_s$ be images generated by deconvolving $\bar{B}$ with the blur kernel pair $\bar{k}_s$, and let $\tilde{\bar{B}}_s^j = \bar{k}_s^j \otimes \tilde{\bar{I}}_s$ be a re-convolved image. The log-likelihood $log(p(\bar{B}|\bar{k}_s))$ at pixel i is:

$$\log p(\bar{B}(i)|\bar{k}_s) \approx -\frac{1}{2\eta^2} \sum_{j=1}^{2} |B^j(i) - \tilde{B}_s^j(i)|^2 - \rho(g_{x,i}(\tilde{I}_s)) - \rho(g_{y,i}(\tilde{I}_s)) + \frac{1}{N}\tilde{\Psi} \quad (36)$$

where $N = 15 \times 15$ is the size of the local window centered around the pixel i.

4.3.1 Handling motion boundaries

Because we take two images sequentially, there are motion boundaries that are visible in one image but not in the other. In such regions the observation model (Eq. 28) is inconsistent and the joint deconvolution leads to visual artifacts. Therefore, we use an image deblurred using only one of the two input images to fill in motion boundaries. We can automatically detect where to use a single-image explanation by also considering kernel candidates that consist of a single image observation (i.e. $B^2 = 0, k^2 = 0$). We add an additional fixed penalty C (set to 0.15 for all experiments, determined through cross validation) to those kernel candidates; otherwise, the log-likelihood (Eq. 36) always favors a single image solution. Algorithm 1 provides a pseudocode for blur kernel estimation at pixel i.

Algorithm 2 `Multi-scale blur estimation`

% Variable definitions
$k_j \equiv$ the kernel estimate at j^{th} scale. j indexes the scale 1 to 3, from coarse to fine.
$\bar{B}^j \equiv$ Input image pyramids at j^{th} scale.
$S_k^1 \equiv 2 \times 4500/4^2$ kernel candidates at the coarsest scale.

Generate a 3-level image pyramid down-sampled in octaves.
% Estimate the blur at the coarsest scale
for every pixel i **do**
  $k_1(i) \Leftarrow \text{EstimateBlurPixel}(\bar{B}^1, S_k^1, i)$
end for
% Regularize the estimate using MRF
$k_1 \Leftarrow \text{MRFRegularize}(k_1)$
% Loop over scales
for $j = 2:3$ **do**
  for every pixel i **do**
    % Velocity candidate reduction
    $S_k^j(i) \Leftarrow 9$ velocity neighbors of $k_{j-1}(i)$
    $k_j(i) \Leftarrow \text{EstimateBlurPixel}(\bar{B}^j, S_k^j(i), i)$
  end for
  $k_j \Leftarrow \text{MRFRegularize}(k_j)$
end for

4.3.2 Multi-scale blur kernel estimation

The quality of restored images depends on how finely we sample the space of 2D constant velocity motions. With our current camera setup, we discretize the space into 4500 samples. We quantize the space such that a step in the velocity space results in roughly a one-pixel blur at the maximum object velocity. Searching over 4500 velocity samples to find the correct kernel pair at the full image resolution is computationally expensive. We resort to a coarse-to-fine strategy to mitigate the computational burden. We first down-sample the input images $\bar{B}$ by a factor of 4 in both width and height to reduce the number of pixels and also the motion search space: blur kernels from two adjacent velocity samples are essentially identical at a coarser resolution. At the coarsest scale, we search through $2 \times 4500/(4^2)$ velocity samples (single-image explanations incur the factor of 2). We then propagate the estimated motion to a finer resolution to refine the estimates.

At each spatial scale, we regularize the log-likelihood in Eq. 36 using a Markov random field (MRF). Algorithm 2 provides a

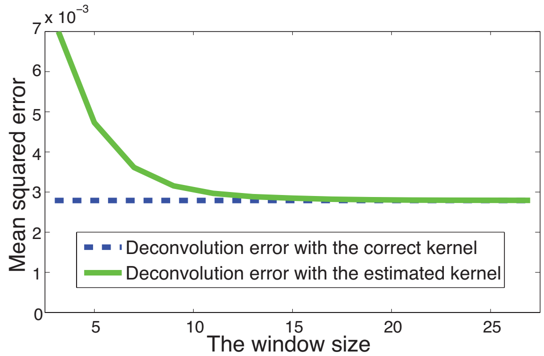
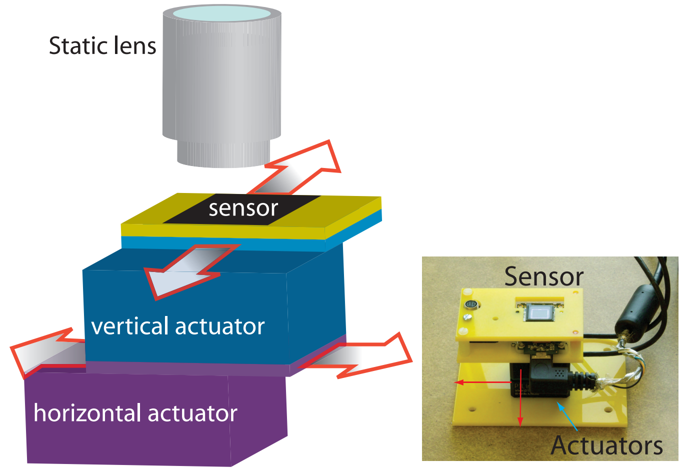


Fig. 10: *This figure evaluates the amount of deconvolution error contributed by the local kernel estimation algorithm. When the local window is larger than 15×15 pixels, the deconvolution error from kernel estimation negligible.*



(a) (b)

Fig. 11: *(a) A diagram of our prototype. (b) A photograph of the actuators and the camera sensor.*

pseudocode for our multi-scale kernel estimation strategy. We use the regularized kernel map to reconstruct the blur free image $\tilde{I}$. First, we deconvolve input images $\bar{B}$ using all blur kernels $k_{s_{x,y}}$ that appear in the estimated kernel map. Given the set of deconvolved images $\tilde{I}_{s_{x,y}}$, we reconstruct the blur free image from $\tilde{I}_{s_{x,y}}$ by selecting, at each pixel, the pixel value from the image deblurred using the estimated blur kernel. We blend different motion layers using a Poisson blending method [20] to reduce artifacts at abutting motion layers.

4.3.3 Quantifying the kernel estimation error

Fig. 10 quantifies the image reconstruction error introduced by kernel estimation. We blur a sharp natural image using a blur kernel pair, and we deblur the rendered images using the correct kernel pair. Then we compute the base-line mean-squared error (MSE). The mean-squared error is not zero because we lose information through the blurry observation process, thus the restored image is not exactly the same as the original image. We also deblur the rendered images using kernels locally estimated by maximizing the log-likelihood in Eq. 36, and compute its MSE. For this experiment, we compute the MSE as we increase the window size, shown as a green curve in Fig. 10. On the same plot, we show the base-line MSE (a dotted blue curve). The base-line MSE is *independent* of the kernel estimation error, therefore the difference between the green curve and the dotted blue curve is the deconvolution error from kernel misidentification. We observe that the additional error from kernel estimation is negligible when the window size is greater than 15×15. This result suggests that it is reasonable to focus on finding a camera motion that maximizes the spectral power of blur kernels.

5 Experiments

5.1 Prototype camera

We built a prototype camera consisting of a sensor, two motion stages and their controllers. We mounted a light-weight camera sensor (Point Grey Research Flea 2 Camera) on two motion stages (Physik Instrumente M-663 pair), where each can move the camera sensor along orthogonal axes (See Fig. 11(a)). In each image capture, one of the motion stages undergoes parabolic motion, approximated by 19 segments of constant velocity due to control constraints. In practice, we could replace the motion stages with an image stabilization hardware. The camera lens is affixed to the camera lid, and does not move during exposure. The total exposure time for taking two images is 500ms: 200ms for each image, with a delay of 100ms between exposures. 100ms delay is incurred by switching the control from one motion stage to another, and can be reduced by using an improved hardware.

We rendered PSFs of our imaging system for different object speed using a parameterized actuator motion model, and used them for deconvolution. We validated the accuracy of rendered PSFs by physically calibrating blur kernels at several object velocities and by comparing them to rendered kernels. For calibration, we place a high-frequency calibration pattern on a motion rail, take a sharp image of the static calibration pattern with a static camera, and take an image of the moving pattern with a camera undergoing a parabolic motion. We solve for the kernel k that minimizes $\|B - k \otimes I\|^2$, where I is the sharp image of the static calibration pattern, and B is the image of the moving pattern taken with a parabolic camera.

5.2 Results

Fig. 12 illustrates the deblurring pipeline. First, we capture two images successively while the sensor undergoes parabolic motions in two orthogonal directions. From the two images, we locally estimate the motion and restore the blur-free image using blur kernels that correspond to the estimated motion. Automatically detected motion boundaries are shown by black bounding boxes. Our kernel estimation algorithm sometimes misclassifies motions in un-textured regions, but this does not lead to visual artifacts. For reference we show an image taken

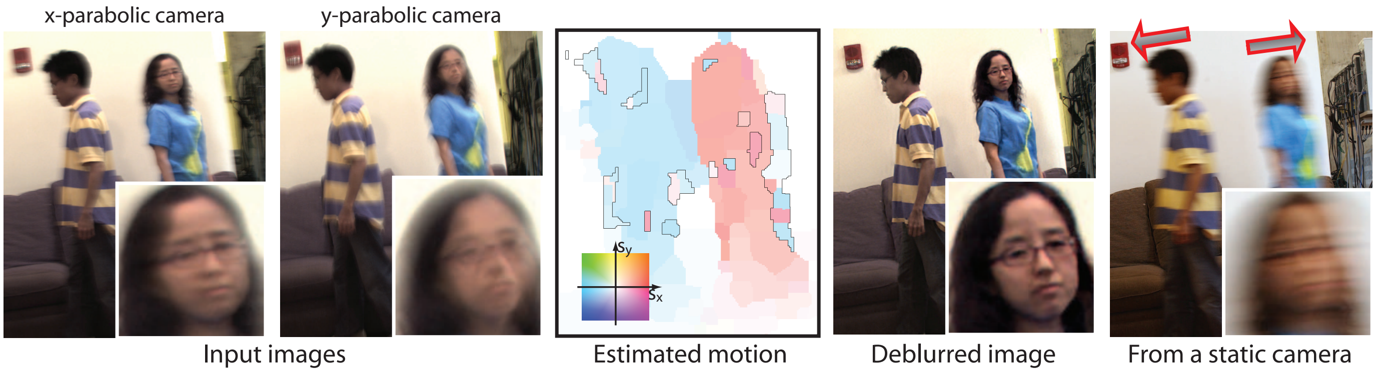


Fig. 12: *This figure shows the pipeline of our system. We take two images using the orthogonal parabolic camera, and we locally estimate motion. The estimated motion is shown with the color coding scheme in the inset, and the detected motion boundaries are represented with black bounding boxes. We deblur the captured image pair using the estimated motion map. For reference, we also show the image taken with a synchronized static camera with a 500ms exposure.*

with a static camera with 500ms exposure, synchronized to the first shot of the orthogonal parabolic camera. This reference image reveals the object motion during exposure.

In Fig. 13, we compare the deconvolution performance of a two-shot static camera and an orthogonal parabolic camera. A toy train is moving at a constant velocity, assumed known for this comparison. For the static camera, we optimize the split of the exposure for this known train motion: 40ms for the first shot, and 360ms for the second shot. Using the static camera, we can reliably reconstruct the static part of the scene at the expense of degraded renditions of moving parts. On the other hand, our camera enables reliable reconstructions of both static and moving parts, although static regions are slightly more degraded compared to static regions restored using the static camera. An orthogonal parabolic camera spreads the energy budget over all velocities of interest, whereas a static camera concentrates the energy budget for the static motion.

We present more deblurring results on human motions in Fig. 14, using parabolic exposure to capture motions in non-horizontal directions [3]. Images from the static camera (500ms exposure) reveal the motions during exposure, shown by red arrows. We can observe some artifacts at motion boundaries at which the joint convolution model does not hold. In general, however, the reconstructions are visually pleasing. In the third column of Fig. 14, we show how an orthogonal parabolic camera handles a perspective motion. While a perspective motion does not conform to our assumption on object motions, our system still recovers a reasonably sharp image.

Our image reconstruction algorithm treats an occluded region as a motion boundary. When a moving object is seen only in one of the two images due to occlusion, as in Fig. 15, an image deblurred using only one of the input images is used to fill in the occluded region.

3. The camera body is tilted for this purpose.

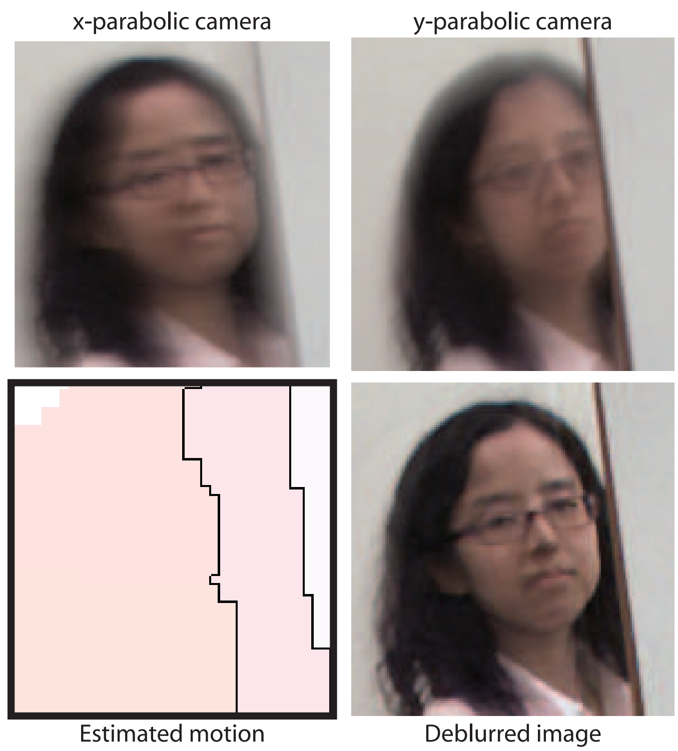


Fig. 15: *Our image reconstruction algorithm handles occlusion boundaries in a manner similar to motion boundaries. In the occluded region, an image deblurred using only one of the two input images is used.*

5.3 Discussion

Kernel estimation takes 30 min - 1 hour on a single, serial machine: the running time depends on the size of the image. A by-product of kernel estimation is a blur free image deblurred using the Wiener filter. The running time of the sparse deconvolution algorithm is roughly 6 hours.

We assume that objects move at a constant velocity within the exposure time, which is a limitation shared by most previous work that deals with object motion [15], [17]. Camera shake,

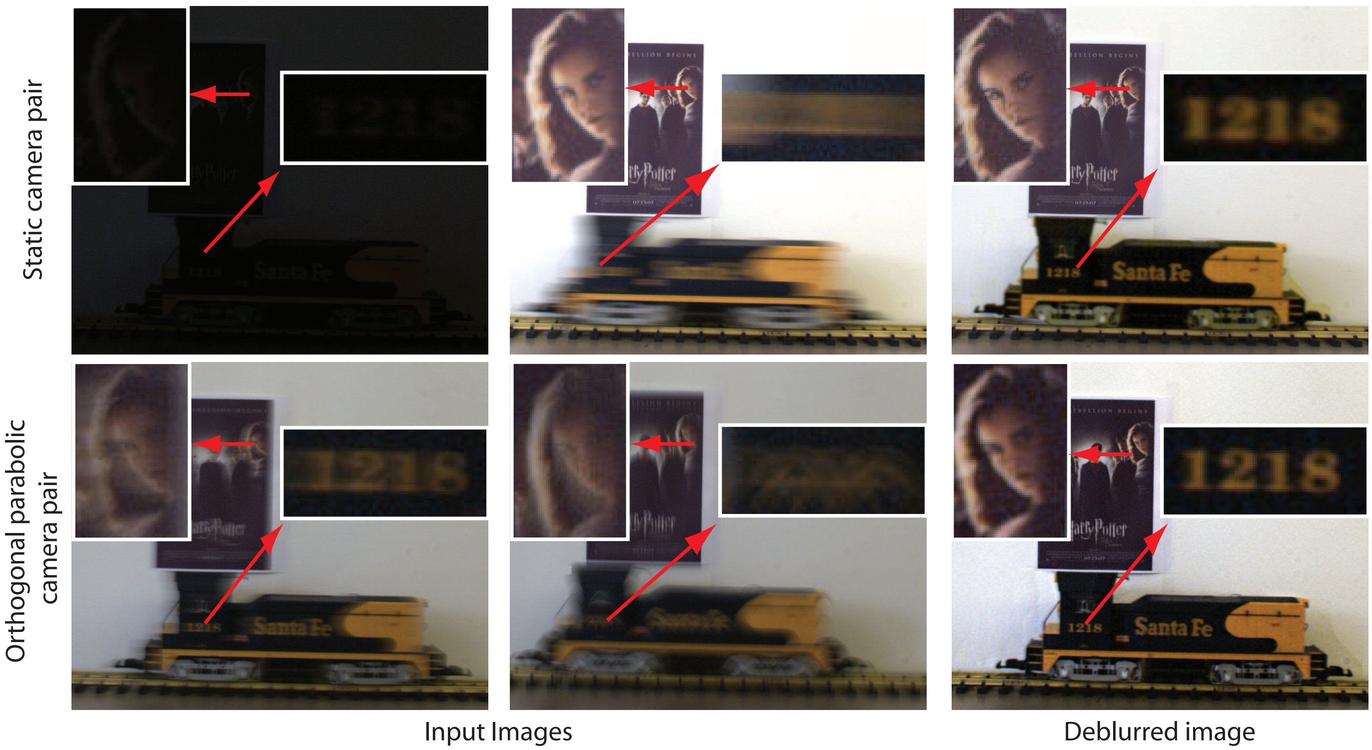


Fig. 13: *We compare the deblurring performance of a two-shot static camera and an orthogonal parabolic camera. We optimize the split of the exposure for the static camera, assuming that we know the object motion: 40ms for the first shot and 360ms for the second shot. The blur kernel is estimated manually to compare just the amount of information captured by these cameras. The static camera reconstructs static objects well, but at the expense of a degraded rendition of the moving object, whereas the orthogonal parabolic camera restores a reasonable rendition of both the static and moving parts.*

which typically exhibits complex kernels, needs to be handled separately. Our camera design captures image information almost optimally, but it does not provide guarantees for kernel estimation performance. While taking two images certainly helps kernel estimation, designing a sensor motion that optimizes both kernel estimation and information capture is an open problem. Our image reconstruction takes into account occlusions by allowing some pixels to be reconstructed from a single images, but a full treatment of occlusion for deconvolution remains an open challenge.

6 Conclusion

This paper presented a two-exposure solution to removing constant velocity object motion blur. We showed that the union of PSFs corresponding to 2D linear motions occupy a wedge of revolution in the Fourier space, and that the spectral content of the orthogonal parabolic camera approaches the optimal bound up to a multiplicative constant within the wedge of revolution.

References

[1] A. Agrawal and R. Raskar. Optimal single image capture for motion deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[2] A. Agrawal and Y. Xu. Coded exposure deblurring: Optimized codes for psf estimation and invertibility. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[3] A. Agrawal, Y. Xu, and R. Raskar. Invertible motion blur in video. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2009.

[4] M. Ben-Ezra and S. K. Nayar. Motion-based motion deblurring. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:689 – 698, 2004.

[5] J. Chen, L. Yuan, C.-K. Tang, and L. Quan. Robust dual motion deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[6] S. Cho, Y. Matsushita, and S. Lee. Removing non-uniform motion blur from images. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2007.

[7] T. S. Cho, A. Levin, F. Durand, and W. T. Freeman. Motion blur removal with orthogonal parabolic exposures. In *Proceedings of International Conference on Computational Photography (ICCP)*, 2010.

[8] S. Dai and Y. Wu. Motion from blur. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[9] S. Dai and Y. Wu. Removing partial blur in a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[10] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman. Removing camera shake from a single image. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2006.

[11] R. C. Gonzalez and R. E. Woods. *Digital Image Processing (3rd Edition)*. Prentice Hall, 2007.

[12] A. Gupta, N. Joshi, C. L. Zitnick, M. Cohen, and B. Curless. Single image deblurring using motion density functions. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2010.

Fig. 14: *We show the deblurring performance of the orthogonal parabolic camera. Images from a static camera with 500ms exposure are shown for reference. Arrows on reference images show the direction and magnitude of motion.*

[13] J. Jia. Single image motion deblurring using transparency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[14] N. Joshi, R. Szeliski, and D. J. Kriegman. PSF estimation using sharp edge prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[15] A. Levin. Blind motion deblurring using image statistics. In *Advances in Neural Information Processing Systems (NIPS)*, 2006.

[16] A. Levin, R. Fergus, F. Durand, and W. T. Freeman. Image and depth from a conventional camera with a coded aperture. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2007.

[17] A. Levin, P. Sand, T. S. Cho, F. Durand, and W. T. Freeman. Motion-invariant photography. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2008.

[18] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Understanding and evaluating blind deconvolution algorithms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[19] R. Ng. Fourier slice photography. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2005.

[20] P. Perez, M. Gangnet, and A. Blake. Poisson image editing. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2003.

[21] R. Raskar, A. Agrawal, and J. Tumblin. Coded exposure photography: Motion deblurring using fluttered shutter. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2006.

[22] A. Rav-Acha and S. Peleg. Two motion-blurred images are better than one. *Pattern Recognition Letters*, 26:311 – 317, 2005.

[23] Q. Shan, L. J. Jia, and A. Agarwala. High-quality motion deblurring from a single image. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2008.

[24] Q. Shan, W. Xiong, and J. Jia. Rotational motion deblurring of a rigid object from a single image. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2007.

[25] M. Sorel and F. Sroubek. Space-variant deblurring using one blurred and one underexposed image. In *Proceedings of the IEEE Conference on Image Processing (ICIP)*, 2009.

[26] Y.-W. Tai, H. Du, M. S. Brown, and S. Lin. Image/video deblurring using a hybrid camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[27] O. Whyte, J. Sivic, A. Zisserman, and J. Ponce. Non-uniform deblurring for shaken images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[28] L. Yuan, J. Sun, L. Quan, and H.-Y. Shum. Image deblurring with blurred/noisy image pairs. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2007.