Understanding and evaluating blind deconvolution algorithms

Anat Levin[1,2], Yair Weiss[1,3], Fredo Durand[1], William T. Freeman[1,4]
[1]MIT CSAIL, [2]Weizmann Institute of Science, [3]Hebrew University, [4]Adobe

Abstract

Blind deconvolution is the recovery of a sharp version of a blurred image when the blur kernel is unknown. Recent algorithms have afforded dramatic progress, yet many aspects of the problem remain challenging and hard to understand. The goal of this paper is to analyze and evaluate recent blind deconvolution algorithms both theoretically and experimentally. We explain the previously reported failure of the naive MAP approach by demonstrating that it mostly favors no-blur explanations. On the other hand we show that since the kernel size is often smaller than the image size a MAP estimation of the kernel alone can be well constrained and accurately recover the true blur.

The plethora of recent deconvolution techniques makes an experimental evaluation on ground-truth data important. We have collected blur data with ground truth and compared recent algorithms under equal settings. Additionally, our data demonstrates that the shift-invariant blur assumption made by most algorithms is often violated.

1. Introduction

Blind deconvolution is the problem of recovering a sharp version of an input blurry image when the blur kernel is unknown [13]. Mathematically, we wish to decompose a blurred image y as

$$y = k \otimes x \tag{1}$$

where x is a visually plausible sharp image, and k is a non negative blur kernel, whose support is small compared to the image size. This problem is severely ill-posed and there is an infinite set of pairs (x, k) explaining any observed y. For example, One undesirable solution that perfectly satisfies eq. 1 is the no-blur explanation: k is the delta (identity) kernel and $x = y$. The ill-posed nature of the problem implies that additional assumptions on x or k must be introduced.

Blind deconvolution is the subject of numerous papers in the signal and image processing literature, to name a few consider [1, 11, 24, 17, 19] and the survey in [13]. Despite the exhaustive research, results on real world images are rarely produced. Recent algorithms have proposed to address the ill-posedness of blind deconvolution by characterizing x using natural image statistics [18, 4, 16, 9, 10, 3, 22]. While this principle has lead to tremendous progress, the results are still far from perfect. Blind deconvolution algorithms exhibit some common building principles, and vary in others. The goal of this paper is to analyze the problem and shed new light on recent algorithms. What are the key challenges and what are the important components that make blind deconvolution possible? Additionally, which aspects of the problem should attract further research efforts?

One of the puzzling aspects of blind deconvolution is the failure of the MAP approach. Recent papers emphasize the usage of a sparse derivative prior to favor sharp images. However, a direct application of this principle has not yielded the expected results and all algorithms have required additional components, such as marginalization across all possible images [18, 4, 16], spatially-varying terms [10, 21], or solvers that vary their optimization energy over time [21]. In this paper we analyze the source of the MAP failure. We show that counter-intuitively, the most favorable solution under a sparse prior is usually a blurry image and not a sharp one. Thus, the global optimum of the MAP approach is the no-blur explanation. We discuss solutions to the problem and analyze the answers provided by existing algorithms. We show that one key property making blind deconvolution possible is the strong asymmetry between the dimensionalities of x and k. While the number of unknowns in x increases with image size, the dimensionality of k remains small. Therefore, while a simultaneous MAP estimation of both x and k fails, a MAP estimation of k alone (marginalizing over x), is well constrained and recovers an accurate kernel. We suggest that while the sparse prior is helpful, the key component making blind deconvolution possible is not the choice of prior, but the thoughtful choice of estimator. Furthermore, we show that with a proper estimation rule, blind deconvolution can be performed even with a weak Gaussian prior.

Finally, we collect motion-blurred data with ground truth. This data allows us to quantitatively compare recent blind deconvolution algorithms. Our evaluation suggest that the variational Bayes approach of [4] outperforms all existing alternatives. This data also shows that the shift invariance convolution model involved in most existing algorithms is often violated and that realistic camera shake includes in-plane rotations.

2. MAP$_{x,k}$ estimation and its limitations

In this paper y denotes an observed blurry image, which is a convolution of an unknown sharp image x with an unknown blur kernel k, plus noise n (this paper assumes i.i.d. Gaussian noise):

$$y = k \otimes x + n. \tag{2}$$

Using capital letters for the Fourier transform of a signal:

$$Y_\omega = K_\omega X_\omega + N_\omega. \tag{3}$$

The goal of blind deconvolution is to infer both k and x given a single input y. Additionally, k is non negative, and its support is often small compared to the image size.

The simplest approach is a maximum-a-posteriori ($\text{MAP}_{x,k}$[1]) estimation, seeking a pair $(\hat{x}, \hat{k})$ maximizing:

$$p(x, k|y) \propto p(y|x, k)p(x)p(k). \tag{4}$$

For simplicity of the exposition, we assume a uniform prior on k. The likelihood term $p(y|x, k)$ is the data fitting term $\log p(y|x, k) = -\lambda\|k \otimes x - y\|^2$. The prior $p(x)$ favors natural images, usually based on the observation that their gradient distribution is sparse. A common measure is

$$\log p(x) = -\sum_i |g_{x,i}(x)|^\alpha + |g_{y,i}(x)|^\alpha + C \tag{5}$$

where $g_{x,i}(x)$ and $g_{y,i}(x)$ denote the horizontal and vertical derivatives at pixel i (we use the simple $[-1\ 1]$ filter) and C is a constant normalization term. Exponent values $\alpha < 1$ lead to sparse priors and natural images usually correspond to α in the range of $[0.5, 0.8]$ [23]. Other choices include a Laplacian prior $\alpha = 1$, and a Gaussian prior $\alpha = 2$. While natural image gradients are very non-Gaussian, we examine this model because it enables an analytical treatment.

The $\text{MAP}_{x,k}$ approach seeks $(\hat{x}, \hat{k})$ minimizing

$$(\hat{x}, \hat{k}) = \arg\min_{x,k} \lambda\|k \otimes x - y\|^2 + \sum_i |g_{x,i}(x)|^\alpha + |g_{y,i}(x)|^\alpha. \tag{6}$$

Eq. (6) reveals an immediate limitation:

Claim 1 *Let x be an arbitrarily large image sampled from the prior $p(x)$, and $y = k \otimes x$. The pair (x, k) optimizing the $\text{MAP}_{x,k}$ score satisfies $|x| \to 0$ and $|k| \to \infty$.*

Proof: For every pair (x, k) we use a scalar s to define a new pair $x' = s \cdot x, k' = 1/s \cdot k$ with equal data fitting $\|k \otimes x - y\|^2 = \|k' \otimes x' - y\|^2$. While the data fitting term is constant, the prior term improves as $s \to 0$. ∎

This observation is not surprising. The most likely image under the prior in Eq. (5) is a flat image with no gradients. One attempt to fix the problem is to assume the mean intensity of the blurred and sharp images should be equal, and constrain the sum of k: $\sum_i k_i = 1$. This eliminates the zero solution, but usually the no-blur solution is still favored.

To understand this, consider the 1D signals x in Fig. 1 that were convolved with a (truncated) Gaussian kernel k^* of standard deviation 4 pixels. We compare two interpretations: 1) the true kernel: $y = k^* \otimes x$. 2) the delta kernel (no blur) $y = k^0 \otimes y$. We evaluate the $-\log p(x, k|y)$ score (Eq. (6)), while varying the α parameter in the prior.

For step edges (Fig. 1(a)) $\text{MAP}_{x,k}$ usually succeeds. The edge is sharper than its blurred version and while the Gaussian prior favors the blurry explanation, appropriate sparse priors ($\alpha < 1$) favor the correct sharp explanation.

[1] We keep estimation variables in subscript to distinguish between a MAP estimation of both x and k, to a MAP estimation of k alone.

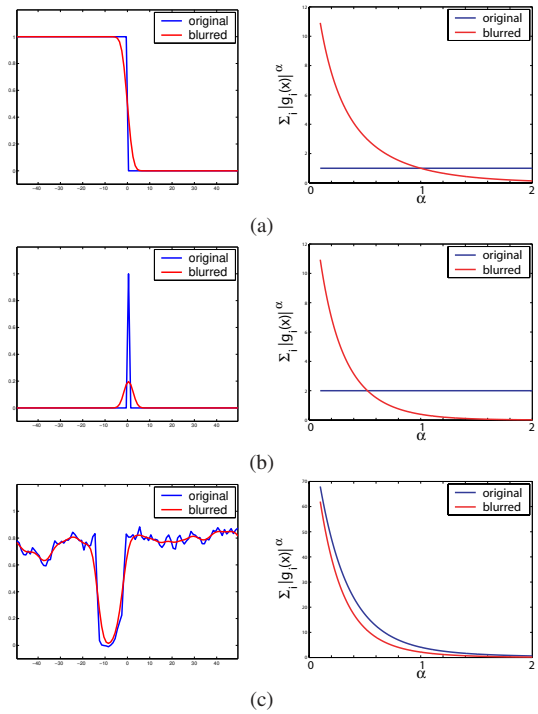


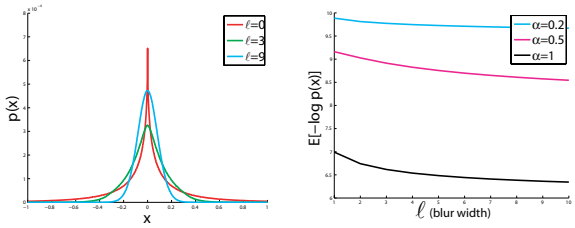
Figure 1. The $\text{MAP}_{x,k}$ score evaluated on toy 1D signals. Left: sharp and blurred signals. Right: sum of gradients $-\log p(x) = \sum_i |g_i(x)|^\alpha$ as a function of α.

15 × 15 windows 3% | 25 × 25 windows 1% | 45 × 45 windows 0%

Figure 2. $\text{MAP}_{x,k}$ failure on real image windows. Windows in which the sharp explanation is favored are marked in red. The percent of windows in which the sharp version is favored decreases with window size.

In contrast, Fig. 1(b) presents a narrow peak. Blurring reduces the peak height, and as a result, the Laplacian prior $\alpha = 1$ favors the blurry x (k is delta) because the absolute sum of gradients is lower. Examining Fig. 1(b-right) suggests that the blurred explanation is winning for smaller α values as well. The sharp explanation is favored only for low alpha values, approaching a binary penalty. However, the sparse models describing natural images are not binary, they are usually in the range $\alpha \in [0.5, 0.8]$ [23].

The last signal considered in Fig. 1(c) is a row cropped from a natural image, illustrating that natural images contain a lot of medium contrast texture and noise, corresponding to the narrow peak structure. This dominates the statistics more than step edges. As a result, blurring a natural image reduces the overall contrast and, as in Fig. 1(b), even sparse priors favor the blurry x explanation.



(a) p^l for $\alpha = 0.5$ (b) $E_{p^l}[-\log p^0(x^l)]$

Figure 3. (a) Comparison of gradient histograms for blurred and unblurred images sampled from $p^0(x)$. Blur reduces the average gradient magnitude. (b) Expected negative likelihood reduces (probability increases) with blur.

To confirm the above observation, we blurred the image in Fig. 2 with a Gaussian kernel of standard deviation 3 pixels. We compared the sum of the gradients in the blurred and sharp images using $\alpha = 0.5$. For 15×15 windows the blurred image is favored over 97% of the windows, and this phenomenon increases with window size. For 45×45 windows, the blurred version is favored at all windows. Another observation is that if the sharp explanation does win, it happens next to significant edges.

To understand this, note that blur has two opposite effects on the image likelihood: 1) it makes the signal derivatives less sparse, and that reduces the likelihood. 2) It reduces the derivatives variance and that increases its likelihood. For very specific images, like ideal step edges, the first effect dominants and blur reduces the likelihood. However, for most natural images the second effect is stronger and blur increases the likelihood. To illustrate this, let x^0 be a sequence sampled i.i.d. from $p^0(x_i^0) \propto e^{-\gamma|x_i^0|^\alpha}$, x^ℓ a sequence obtained by convolving x^0 with a width ℓ box filter (normalizing the kernel sum to 1), and p^ℓ its probability distribution. The expected negative log likelihood (effecting the $\text{MAP}_{x,k}$) of x^ℓ under the sharp distribution p^0 is: $E_{p^\ell}[-\log p^0(x^\ell)] = -\int p^\ell(x) \log p^0(x) dx$. Fig. 3(a) plots p^ℓ for $\alpha = 0.5$, and Fig. 3(b) the expected likelihood as a function of ℓ. The variance is reduced by convolution, and hence the negative log-likelihood reduces as well.

Revisiting the literature on the subject, Fergus *et al.* [4] report that their initial attempts to approach blind deconvolution with $\text{MAP}_{x,k}$ failed, resulting in either the original blurred explanation or a binary two-tone image, depending on parameter tunings.

Algorithms like [10, 9] explicitly detect edges in the image (either manually or automatically), and seek a kernel which transfers these edges into binary ones. This is motivated by the example in Fig. 2, suggesting that $\text{MAP}_{x,k}$ could do the right thing around step edges. Another algorithm which makes usage of this property is [21]. It optimizes a semi-$\text{MAP}_{x,k}$ score, but explicitly detects smooth image regions and reweights their contribution. Thus, the $\text{MAP}_{x,k}$ score is dominated by edges. We discuss this algorithm in detail in the appendix. Earlier blind deconvolution papers which exploit a $\text{MAP}_{x,k}$ approach avoid the delta solution using other assumptions which are less applicable for real world images. For example, [1] assumes x contains an object on a flat background with a known compact support.

All these examples highlight the fact that the prior alone does not favor the desired result. The source of the problem is that for all α values, the most likely event of the prior in Eq. (5) is the fully flat image. This phenomenon is robust to the exact choice of prior, and replacing the model in Eq. (5) with higher order derivatives or with more sophisticated natural image priors [20, 25] does not change the result. We also note that the problem is present even if the derivatives signal is sampled exactly from $p(x)$ and the prior is perfectly correct in the generative sense.

In the next section we suggest that, to overcome the $\text{MAP}_{x,k}$ limitation, one should reconsider the choice of estimator. We revisit a second group of blind deconvolution algorithms derived from this idea.

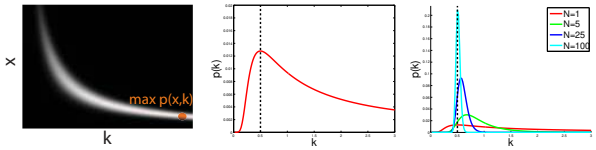
3. MAP_k estimation

The limitations of MAP estimation in the case of few measurements have been pointed out many times in estimation theory and statistical signal processing [12, 2]. Indeed, in the $\text{MAP}_{x,k}$ problem we can never collect enough measurements because the number of unknowns grows with the image size. In contrast, estimation theory tells us [12] that, given enough measurements, MAP estimators do approach the true solution. Therefore, the key to success is to exploit a special property of blind deconvolution: the strong asymmetry between the dimensionalities of the two unknowns. While the dimensionality of x increases with the image size, the support of the kernel is fixed and small relative to the image size. The image y does provide a large number of measurements for estimating k. As we prove below, for an increasing image size, a MAP_k estimation of k alone (marginalizing over x) can recover the true kernel with an increasing accuracy. This result stands in contrast to Claim 1 which stated that a $\text{MAP}_{x,k}$ estimator continues to fail *even as the number of measurements goes to infinity*. This leads to an alternative blind deconvolution strategy: use a MAP_k estimator to recover the kernel and, given the kernel, solve for x using a non blind deconvolution algorithm.

Before providing a formal proof, we attempt to gain an intuition about the difference between MAP_k and $\text{MAP}_{x,k}$ scores. A MAP_k estimator selects $\hat{k} = \arg\max_k p(k|y)$, where $p(k|y) = p(y|k)p(k)/p(y)$, and $p(y|k)$ is obtained by *marginalizing over* x, and evaluating the full volume of possible x interpretations:

$$p(y|k) = \int p(x, y|k) dx. \tag{7}$$

To see the role of marginalization, consider the scalar blind deconvolution problem illustrated in [2]. Suppose a scalar y is observed, and should be decomposed as $y = k \cdot x + n$. Assume a zero mean Gaussian prior on the noise and signal,



(a) $p(x,k|y)$ (b) $p(k|y_1)$ (c) $p(k|y_1,..y_N)$

Figure 4. A toy blind deconvolution problem with one scalar $y = kx+n$ (replotted from [2]). (a) The joint distribution $p(x,k|y)$. A maximum is obtained for $x \to 0,\ k \to \infty$. (b) The marginalized score $p(k|y)$ produce an optimum closer to the true k^*. (c) The uncertainty of $p(k|y)$ reduces given multiple observations $y_j = kx_j + n_j$.

$x \sim N(0,\sigma^2),\ \ n \sim N(0,\eta^2)$. Then

$$P(x,k|y) \propto e^{-\frac{1}{2\eta^2}|kx-y|^2 - \frac{x^2}{2\sigma^2}}. \tag{8}$$

Fig. 4(a) illustrate the 2D distribution $P(x,k|y)$. Unsurprisingly, it is maximized by $x \to 0,\ k \to \infty$. On the other hand, $p(y|k)$ is the integral over all x explanations:

$$P(y|k) \propto \int e^{-\frac{1}{2\eta^2}|kx-y|^2 - \frac{x^2}{2\sigma^2}} dx. \tag{9}$$

This integral is not maximized by $k \to \infty$. In fact, if we consider the first term only $\int e^{-\frac{1}{2\eta^2}|kx-y|^2} dx$, it clearly favors $k \to 0$ values because they allow a larger volume of possible x values. To see that, note that for every k and every $\epsilon > 0$ the size of the set of x values satisfying $|kx - y| < \epsilon$ is $2\epsilon/k$, maximized as $k \to 0$. Combining the two terms in (9) leads to an example in the middle of the range, and we show in Sec. 3.2.1 that $x \approx \sigma$, which make sense because x now behaves like a typical sample from the prior. This is the principle of genericity described in Bayesian terms by [2]. Fig. 4(b) plots $P(y|k)$, which is essentially summing the columns of Fig. 4(a).

Now consider blur in real images: for the delta kernel there is only a single solution $x = y$ satisfying $k \otimes x = y$. However, while the delta spectrum is high everywhere, the true kernel is usually a low pass, and has low spectrum values. Referring to the notation of Eq. (3), if $K_\omega = 0$, an infinite subspace of possible explanations is available as X_ω can be arbitrary (and with noise, any low $|K_\omega|$ values increase the uncertainty, even if they are not exactly 0). Hence, the true kernel gets an advantage in the $p(y|k)$ score.

We prove that for sufficiently large images, $p(k|y)$ is guaranteed to favor the true kernel.

Claim 2 *Let x be an arbitrarily large image, sampled from the prior $p(x)$, and $y = k \otimes x + n$. Then $p(k|y)$ is maximized by the true kernel k^*. Moreover, if $\arg\max_k p(y|k)$ is unique, $p(k|y)$ approaches a delta function*[2].

[2]Note that Claim 2 does not guarantee that the MAP$_k$ is unique. For example, if the kernel support is not constrained enough, multiple spatial shifts of the kernel provide equally good solutions. The problem can be easily avoided by a weak prior on k (e.g. favoring centered kernels).

Proof: We divide the image into small disjoint windows $\{y^1, ..., y^n\}$ and treat them as i.i.d. samples $y^j \sim p(y|k^*)$. We then select $k^{ML} = \arg\max_k \prod_j p(y^j|k)$. Applying the standard consistency theorem for maximum likelihood estimators [12] we know that given enough samples, the ML approaches the true parameters. That is, when $n \to \infty$

$$p(k^{ML}(\{y^1, ..., y^n\}) = k^*) \to 1. \tag{10}$$

Due to the local form of the prior $p(x)$ (Eq. (5)), taking sufficiently far away disjoint windows will ensure that $p(y|k) \approx \prod_j p(y^j|k)$. Thus, $p(y|k)$ is maximized by k^{ML}. Also, if we select a m times larger image y', $p(y'|k) = p(y|k)^m$. Thus, if $p(y|k) < \max_k p(y|k)$ then $p(y|k) \to 0$. Finally, if $p(k^*) > 0$, then k^{MAP}, k^{ML} are equal on large images since $\arg\max_k p(y|k) = \arg\max_k p(y|k)p(k)$, and thus, $k^{MAP} \to k^*$. Similarly, if $\max_k p(y|k)$ is unique, $p(k|y)$ approaches a delta function. ∎

Fig. 4(c) plots $p(y|k)$ for a scalar blind deconvolution task with N observations $y_j = kx_j + n_j$, illustrating that as N increases, the uncertainty around the solution decreases (compare with Fig. 4(b)).

3.1. The loss function perspective

As another way to understand the difference between the MAP$_{x,k}$ and MAP$_k$ estimators, we return to the definition of a Bayesian estimator. A Bayesian estimator involves a loss function $L(\hat{x} - x, \hat{k} - k)$ on both parameters, specifying the price for an estimation error. The expected loss is minimized by:

$$(\hat{x}, \hat{k}) = \arg\min \iint p(x,k|y) L(\hat{x} - x, \hat{k} - k) dx dk. \tag{11}$$

One simple choice of loss function yielding the MAP$_{x,k}$ solution is the Dirac delta loss function: $L(\hat{x} - x, \hat{k} - k) = 1 - \boldsymbol{\delta}\Big((\hat{x}, \hat{k}) - (x,k)\Big)$. The limitations of this loss have been pointed out many times [12, 2]. This "all or nothing" loss is too harsh for many signal processing applications, as it completely ignores all information around the mode. Instead, it is common to use loss functions that increase more smoothly with estimation error, such as the mean squared error (MSE) loss: $L(x,k) = |x - \hat{x}|^2 + |k - \hat{k}|^2$, or a robustified loss like the MLM [2].

Claim 3 *If $p(k|y)$ has a unique maxima, then for large images a MAP$_k$ estimator followed by a $MMSE_x$ image estimation, is equivalent to a simultaneous MMSE$_{x,k}$ estimation of both x and k*[3].

[3]If multiple solutions with equal probability exist, MMSE$_{x,k}$ and MAP$_k$ are not fully equivalent, and MMSE$_{x,k}$ leads to undesired averaging. On the other hand, MAP$_k$ avoids the problem by picking one solution.

Proof: The mean squared error is minimized by the mean, and in our case $\text{MMSE}_{x,k}$ provides

$$\begin{aligned}\hat{x} &= \iint p(x,k|y)x\,dxdk \\ &= \iint p(k|y)p(x|y,k)x\,dxdk \\ &= \int p(k|y)\mu^{(k)}dk \qquad (12)\end{aligned}$$

where $\mu^{(k)} = \int p(x|y,k)xdx$, is a "non blind" MMSE_x estimation of x given k. From Claim 2, $p(k|y)$ is a delta function and thus: $\hat{x} = \mu^{(k^{MAP})}$. □

3.2. Examples of MAP_k estimation

Claim 2 reduces to a robust blind deconvolution strategy: use MAP_k estimator to recover $k^{MAP} = \arg\max_k p(k|y)$, and then use k^{MAP} to solve for x using some non blind deconvolution algorithm. To illustrate the MAP_k approach, we start with the simple case of a Gaussian prior on $p(x)$, as it permits a derivation in closed form.

3.2.1 The Gaussian prior

The prior on X in Eq. (5) is a convolution and thus diagonal in the frequency domain. If G_x, G_y denote the Fourier transform of the derivatives g_x, g_y, then:

$$X \sim N(0, diag(\sigma_\omega^2)) \quad \sigma_\omega^2 = \beta(\|G_{x,\omega}\|^2 + \|G_{y,\omega}\|^2)^{-1}. \quad (13)$$

Note that since a derivative filter is zero at low frequencies and high at higher frequencies, this is similar to the classical $1/f^2$ power spectrum law for images. Denoting noise variance by η, we can express $p(X,Y;K) = p(Y|X;K)p(X)$ as:

$$p(X,Y;K) \propto e^{-\frac{1}{2\eta^2}\|K_\omega X_\omega - Y_\omega\|^2 - \frac{1}{2\sigma_\omega^2}\|X_\omega\|^2}. \quad (14)$$

(see the appendix for details). Conditioned on k, the mean and mode of a Gaussian are equal:

$$X_\omega^{MAP} = \left(|K_\omega|^2 + \frac{\eta^2}{\sigma_\omega^2}\right)^{-1} K_\omega^T Y_\omega. \quad (15)$$

Eq. (15) is the classic Wiener filter [7]. One can also integrate X and express $p(Y|K)$ analytically. This is also a diagonal zero mean Gaussian with

$$Y \sim N(0, diag(\phi_\omega^2)), \quad \phi_\omega^2 = \sigma_\omega^2|K_\omega|^2 + \eta^2. \quad (16)$$

Eq. (16) is maximized when $\phi_\omega^2 = |Y_\omega|^2$, and for blind deconvolution, this implies:

$$|\hat{K}_\omega|^2 = \max\left(0, \frac{|Y_\omega|^2 - \eta^2}{\sigma_\omega^2}\right). \quad (17)$$

The image estimated using $\hat{K}$ satisfies $|X_\omega|^2 \approx \sigma_\omega^2$. Therefore MAP_k does not result in a trivial $X = 0$ solution as $\text{MAP}_{x,k}$ would, but in a solution whose variance matches the prior variance σ^2, that is, a solution which looks like a typical sample from the prior $p(X)$.

Another way to interpret the MAP_k, is to note that

$$\log p(Y|K) = \log p(X^{MAP}, Y; K) - \frac{1}{2}\sum_\omega \log\left(\frac{|K_\omega|^2}{\eta^2} + \frac{1}{\sigma_\omega^2}\right) + C \quad (18)$$

Referring to Eq. (14), the second term is just the log determinant of the covariance of $p(X|Y;K)$. This second term is optimized when $K_\omega = 0$, i.e. by kernels with more blur. That is, $\log p(Y|K)$ is equal to the $\text{MAP}_{x,k}$ score of the mode plus a term favoring kernels with blur.

The discussion above suggests that the Gaussian MAP_k provides a reasonable solution to blind deconvolution. In the experiment section we evaluate this algorithm and show that, while weaker than the sparse prior, it can provide acceptable solutions. This stands in contrast to the complete failure of a $\text{MAP}_{x,k}$ approach, even with the seemingly better sparse prior. This demonstrates that a careful choice of estimator is actually more critical than the choice of prior.

Note that Eq. (17) is accurate if every frequency is estimated independently. In practice, the solution can be further constrained, because the limited spatial support of k implies that the frequency coefficients $\{K_\omega\}$ are linearly dependent. Another important issue is that Eq. (17) provides information on the kernel power spectrum alone but leaves uncertainty about the phase. Many variants of Gaussian blind deconvolution algorithms are available in the image processing literature (e.g. [11, 17]) but in most cases only symmetric kernels are considered since their phase is known to be zero. However, realistic camera shake kernels are usually not symmetric. In the appendix we describe a Gaussian blind deconvoltion algorithm which attempts to recover non symmetric kernels as well.

3.2.2 Approximation strategies with a sparse prior

The challenge with the MAP_k approach is that for a general sparse prior, $p(k|y)$ (Eq. (7)) cannot be computed in closed form. Several previous blind deconvolution algorithms can be viewed as approximation strategies for MAP_k, although the authors might not have motivated them in this way.

A simple approximation is proposed by Levin [16], for the 1D blur case. It assumes that the observed derivatives of y are independent (this is usually weaker than assuming independent derivatives of x): $\log p(y|k) = \sum_i \log p(g_{x,i}(y)|k)$. Since $p(g_{x,i}(y)|k)$ is a 1D distributions, it can be expressed as a 1D table, or a histogram h^k. The independence assumption implies that instead of summing over image pixels, one can express $p(y|k)$ by summing over histogram bins:

$$\log p(y|k) = \sum_i \log p(g_{x,i}(y)|k) = \sum_j h_j \log(h_j^k) \quad (19)$$

where h denotes the gradients histogram in the observed image and j is a bin index. In a second step, note that maximizing Eq. (19) is equivalent to minimizing the histogram dis-

tance between the observed and expected histograms h,h^k. This is because the Kullback Leibler divergence is equal to the negative log likelihood, plus a constant that does not depend on k (the negative entropy):

$$D_{KL}(h, h^k) = \sum_j h_j \log(h_j) - \sum_j h_j \log(h_j^k). \quad (20)$$

Since the KL divergence is non-negative, the likelihood is maximized when the histograms h, h^k are equal. This very simple approach is already able to avoid the delta solution but as we demonstrate in Sec. 4.1 it is not accurately identifying the exact filter width.

A stronger approximation is the variational Bayes mean-field approach taken by Fergus *et al*. [4]. The idea is to build an approximating distribution with a simpler parametric form:

$$p(x, k|y) \approx q(x, k) = \prod_i q(g_{i,x}(x))q(g_{i,y}(x)) \prod_j q(k_j). \quad (21)$$

Since q is expressed in the gradient domain this does not recover x directly. Thus, they also pick the MAP$_k$ kernel from q and then solve for x using non blind deconvolution.

A third way to approximate the MAP$_k$ is the Laplace approximation [2], which is a generalization of Eq. (18):

$$\log p(y|k) \approx \log p(x^{MAP}, y; k) - \frac{1}{2} \log |A| + C \quad (22)$$

$$A = \frac{\partial^2}{\partial x_i \partial x_j} \log p(x, y; k)|_{x=x^{MAP}}. \quad (23)$$

The Laplace approximation states that $p(y|k)$ can be expressed by the probability of the mode x^{MAP} plus the log determinant of the variance around the mode. As discussed above, higher variance is usually achieved when k contains more zero frequencies, i.e. more blur. Therefore, the Laplace approximation suggests that $p(y|k)$ is the MAP$_{x,k}$ score plus a term pulling toward kernels with more blur. Unfortunately, in the non Gaussian case the covariance matrix isn't diagonal and exact inversion is less trivial. Some earlier blind deconvolution approaches [24, 19] can be viewed as simplified forms of a blur favoring term. For example, they bias towered blurry kernels by adding a term penalizing the high frequencies of k or with an explicit prior on the kernel. Another approach was exploit by Bronstein *et al*. [3]. They note that in the absence of noise and with invertible kernels $p(k|y)$ can be exactly evaluated for sparse priors as well. This reduces to optimizing the sparsity of the image plus the log determinant of the kernel spectrum.

4. Evaluating blind deconvolution algorithms

In this section we qualitatively compare blind deconvolution strategies on the same data. We start with a synthetic 1D example and in the second part turn to real 2D motion.

4.1. 1D evaluation

As a first test, we use a set of 1000 signals of size 10×1 cropped from a natural image. These small 1D signals allow us to evaluate the marginalization integral in Eq. (7) exactly even for a sparse prior. The signals were convolved with a 5-tap box filter (cyclic convolution was used) and an i.i.d. Gaussian noise with standard deviation 0.01 was added. We explicitly search over the explanations of all box filters of size $\ell = 1, .., 7$ taps (all filters normalized to 1). The explicit search allows comparison of the score of different blind deconvolution strategies without folding in optimization errors. (In practice optimization errors do have a large effect on the successes of blind deconvolution algorithms.)

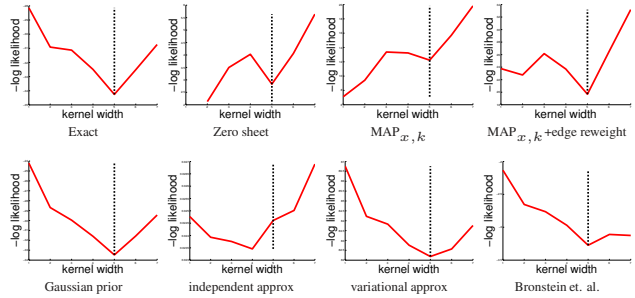


Figure 5. $\log p(y|k)$ scores using various approximation strategies on 1D image signals. Successful algorithms locate the minimum score at the true kernel width, denoted by the dashed line.

The exact $-\log p(y|k)$ score is minimized by the true box width $\ell = 5$.

We tested the zero sheet separation (e.g. [14]), an earlier image processing approach with no probabilistic formulation. This algorithm measures the Fourier magnitude of y at the zero frequencies of each box filter k. If the image was indeed convolved with that filter, low Fourier content is expected. However, this approach considers the zero frequencies alone ignoring all other information, and is known to be noise sensitive. It is also limited to kernel families from a simple parametric form and with a clear zeros structure.

Supporting the example in Sec. 2, a pure MAP$_{x,k}$ approach ($p(y|k) \approx p(x^{MAP}, y|k)$) favors no-blur ($\ell = 1$). Reweighting the derivative penalty around edges can improve the situation, but the delta solution still provides a noticeable local optimum.

The correct minimum is favored with a variational Bayes approximation [4] and with the semi Laplace approximation of [3]. The independence approximation [16] is able to overcome the delta solution, but does not localize the solution very accurately (minimum at $\ell = 4$ instead of $\ell = 5$.) Finally, the correct solution is identified even with the poor image prior provided by a Gaussian model, demonstrating that the choice of estimator (MAP$_{x,k}$ v.s. MAP$_k$), is more critical than the actual prior (Gaussian v.s. sparse).

Since claim 2 guaranties success only for large images, we attempt to evaluate how large an image should be in practice. Fig. 6 plots the uncertainty in $p(k|y)$ for multiple random samples of N 10×1 columns. The probability is tightly peaked at the right answer for as little as $N = 20$ columns. The search space in Fig. 6 is limited to the single

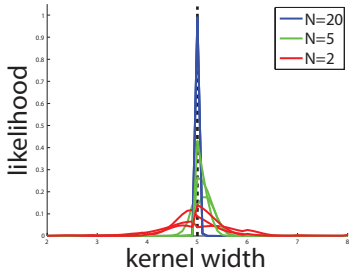
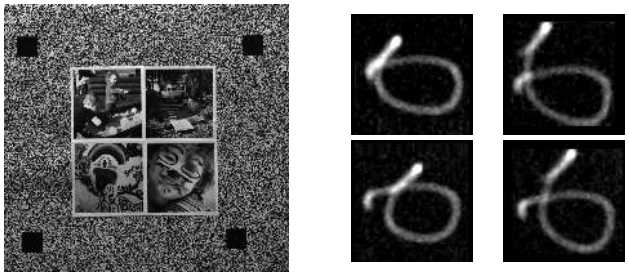


Figure 6. The uncertainty in kernel estimation decreses with more samples. For as little at $N = 20$ columns it is already tightly peaked at the true answer.



(a) (b)

Figure 7. Ground truth data acquisition. (a) Calibration image. (b) Smear of points at 4 corners, demonstrating that the spatially uniform blur model is violated.

parameter family of box filters. In real motion deblurring one searches over a larger family of kernels and a larger uncertainty is expected.

4.2. 2D evaluation

To compare blind deconvolution algorithms we have collected blurred data with ground truth. We capture a sharp version a planar scene (Fig. 7(a)) by mounting the camera on a tripod, as well as a few blurred shots. Using the sharp reference we solve for a non-negative kernel k minimizing $\|k \otimes x - y\|^2$. The scene in Fig. 7(a) includes high frequency noise patterns which helps stabilizing the constraints on k. The central area of the frame includes four real images used as input to the various blind deconvolution algorithms.

We first observed that assuming a uniform blur over the image is not realistic even for planar scenes. For example Fig. 7(b) shows traces of points at 4 corners of an image captured by a hand-held camera, with a clear variation between the corners. This suggests that an in-plane rotation (rotation around the z-axis) is a significant component of human hand shake. Yet, since a uniform assumption is made by most algorithms, we need to evaluate them on data which obeys their assumption. To capture images with spatially invariant blur we placed the camera on a tripod, locking the Z-axis rotation handle of the tripod but loosening the X and Y handles. We calibrated the blur of 8 such images and cropped $4 \quad 255 \times 255$ windows from each, leading to 32 test images displayed in Fig. 8 and available online[4].

[4] www.wisdom.weizmann.ac.il/~levina/papers/LevinEtalCVPR09Data.zip

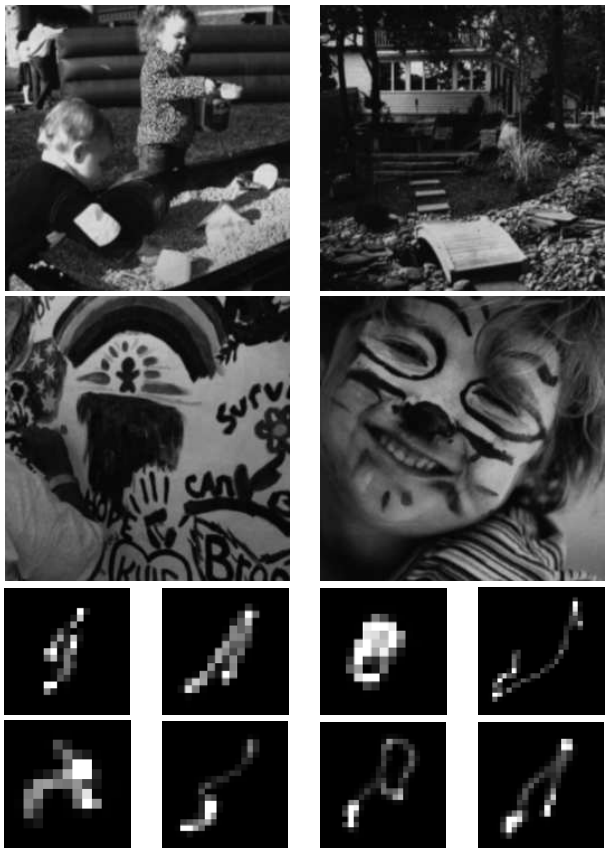


Figure 8. Ground truth data: 4 images and 8 blur kernels, resulting in 32 test images

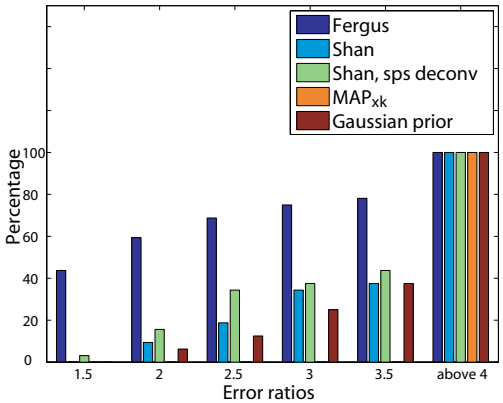


Figure 9. Evaluation results: Cumulative histogram of the deconvolution error ratio across test examples.

We used an 85mm lens and a 0.3 seconds exposure. The kernels' support varied from 10 to 25 pixels.

We can measure the SSD error between a deconvolved output and the ground truth. However, wider kernels result in larger deconvolution error even with the true kernel. To normalize this effect, we measure the ratio between deconvolution error with the estimated kernel and deconvolution

with the truth kernel. In Fig. 9 we plot the cumulative histogram of error ratios (e.g. bin $r = 3$ counts the percentage of test examples achieving error ratio below 3). Empirically, we noticed that error ratios above 2 are already visually implausible. The dataset and all deconvolution results are included at the end of this manuscript.

We have evaluated the algorithms of Fergus *et al*. [4] and Shan *et al*. [21] (each using the authors' implementation), as well as MAP_k estimation using a Gaussian prior (described in the appendix), and a simplified $\text{MAP}_{x,k}$ approach constraining $\sum k_i = 1$ (we used coordinate descent, iterating between holding x constant and solving for k, and then holding k constant and solving for x using the sparse deconvolution algorithm of [15]). The algorithms of [16, 10, 3] were not tested because the first was designed for 1D motion only, and the others focus on smaller blur kernels.

We made our best attempt to adjust the parameters of Shan *et al*. [21], but run all test images with equal parameters. Fergus *et al*. [4] used Richardson-Lucy non blind deconvolution in their code. Since this algorithm is a source for ringing artifacts, we improved the results using the kernel estimated by the authors' code with the (non blind) sparse deconvolution of [15]. Similarly, we used sparse deconvolution with the kernel estimated by Shan *et al*.

The bars in Fig. 9 and the visual results in the appendix suggest that Fergus *et al*.'s algorithm [4] significantly outperforms all other alternatives. Many of the artifacts in the results of [4] can be attributed to the Richardson-Lucy non blind deconvolution artifacts, or to non uniform blur in their test images. Our comparison also suggests that applying sparse deconvolution using the kernels outputted by Shan *et al*. [21] improves their results. As expected, the naive $\text{MAP}_{x,k}$ approach outputs small kernels approaching the delta solution.

5. Discussion

This paper analyzes the major building blocks of recent blind deconvolution algorithms. We illustrate the limitation of the simple $\text{MAP}_{x,k}$ approach, favoring the no-blur (delta kernel) explanation. One class of solutions involves explicit edge detection. A more principled strategy exploits the dimensionality asymmetry, and estimates MAP_k while marginalizing over x. While the computational aspects involved with this marginalization are more challenging, existing approximations are powerful.

We have collected motion blur data with ground truth and quantitatively compared existing algorithms. Our comparison suggests that the variational Bayes approximation [4] significantly outperforms all existing alternatives.

The conclusions from our analysis are useful for directing future blind deconvolution research. In particular, we note that modern natural image priors [20, 25] do not overcome the $\text{MAP}_{x,k}$ limitation (and in our tests did not change the observation in Sec. 2). While it is possible that blind deconvolution can benefit from future research on natural image statistics, this paper suggests that better estimators for existing priors may have more impact on future blind deconvolution algorithms. Additionally, we observed that the popular spatially uniform blur assumption is usually unrealistic. Thus, it seems that blur models which can relax this assumption [22] have a high potential to improve blind deconvolution results.

Acknowledgments: We thank the Israel Science Foundation, the Royal Dutch/Shell Group, NGA NEGI-1582-04-0004, MURI Grant N00014-06-1-0734, NSF CAREER award 0447561. Fredo Durand acknowledges a Microsoft Research New Faculty Fellowship and a Sloan Fellowship.

6. Appendix A: Blind deconvolution with a Gaussian prior

To complete section 3.2.1 of the main paper, we provide a detailed derivation of a MAP_k estimation algorithm using a Gaussian prior. The simple analytic treatment of a Gaussian prior is attractive both from a computational viewpoint and from a research viewpoint, as it affords intuition. While the algorithm is not as powerful as sparse deconvolution algorithms, it approaches the solution using second order statistics alone.

To derive the Gaussian algorithm, we rewrite the generative model explicitly for a Gaussian prior and, to simplify notation, use the frequency domain.

$\mathbf{p(Y|X;K)}$: The spatial i.i.d. Gaussian observation noise is invariant to the frequency basis change. Therefore

$$(Y_\omega|X_\omega;K_\omega) \sim N(K_\omega X_\omega, \eta^2) \tag{24}$$

where η denotes the noise variance.

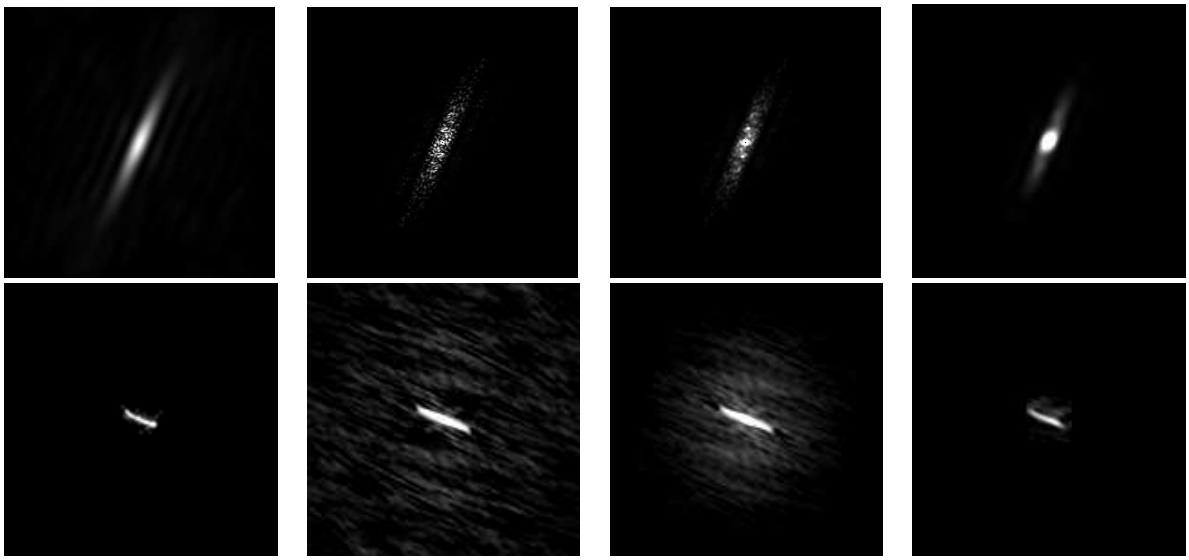
$\mathbf{p(X)}$: The prior on X uses a convolution and is diagonal in the frequency domain. If G_x, G_y denote the Fourier transform of the derivative filters g_x, g_y, the convolution and Parseval's theorems result in $\sum_i |g_{x,i}(x)|^2 + |g_{y,i}(x)|^2 = \sum_\omega |G_{x,\omega}X_\omega|^2 + |G_{y,\omega}X_\omega|^2$. Therefore X follows a zero mean Gaussian distribution with diagonal covariance:

$$X \sim N(0, diag(\sigma_\omega^2)) \quad \sigma_\omega^2 = \beta(\|G_{x,\omega}\|^2 + \|G_{y,\omega}\|^2)^{-1}. \tag{25}$$

(the scale β can be fitted based on the derivative histogram in a natural image). Note that since a derivative filter is zero at the low frequencies and high at the higher frequencies, this is very similar to the classical $1/f^2$ power spectrum law (and our algorithm produced very similar results with an explicit $1/f^2$ prior).

MAP_x estimation:

$$X^{MAP} = \arg\max p(X,Y;K) = \arg\max p(Y|X;K)p(X). \tag{26}$$



(a) Ground truth (b) Independent estimation (c)Smoothing PWS (d) Compact support constraint

Figure 10. Power spectrum estimation and the compact support constraint. Top: power spectrum, Bottom: kernel in primal domain

Therefore, solving for the MAP$_x$ (using Eqs. (24,25)) is a least square minimization:

$$X_\omega^{MAP} = \arg\min \frac{1}{\eta^2}\|K_\omega X_\omega - Y_\omega\|^2 + \frac{1}{\sigma_\omega^2}\|X_\omega\|^2 \quad (27)$$

$$X_\omega^{MAP} = \left(|K_\omega|^2 + \frac{\eta^2}{\sigma_\omega^2}\right)^{-1} K_\omega^T Y_\omega. \quad (28)$$

Eq. (28) is essentially the famous Wiener filter [7]. The prior term in Eq. (28) pulls the estimation toward zero, pulling stronger at high frequencies where the expected signal magnitude is small ($\sigma_\omega \to 0$) and noise contribution is higher. When the filter value $K_\omega = 0$, the signal value cannot be recovered and the prior leads the estimation to $X_\omega = 0$.

$\mathbf{p(Y)}$: One can also integrate X and express $p(Y|K)$ analytically. This is also a diagonal zero mean Gaussian with

$$Y \sim N(0, diag(\phi_\omega^2)), \quad \phi_\omega^2 = \sigma_\omega^2|K_\omega|^2 + \eta^2. \quad (29)$$

Given Eqs. (24-29), we can return to blind deconvolution. If we were to estimate every frequency K_ω independently, we could differentiate Eq. (29) and conclude it is maximized when $\phi_\omega^2 = |Y_\omega|^2$, which results in:

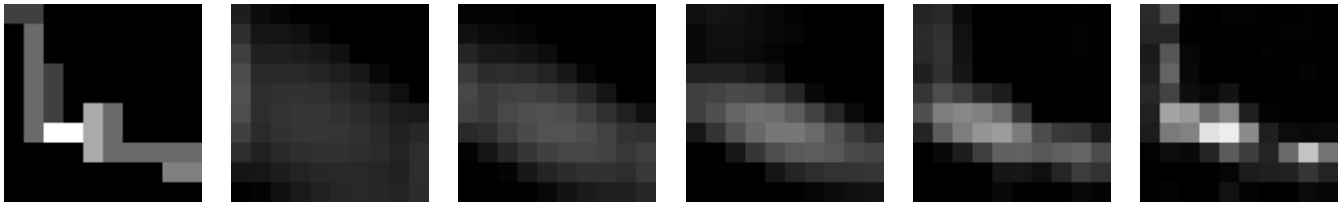
$$|K_\omega|^2 = \max(0, \frac{|Y_\omega|^2 - \eta^2}{\sigma_\omega^2}). \quad (30)$$

Eq. (30) essentially states that the optimal K leads to an X whose power spectrum equals the expected power spectrum σ^2. However, for frequencies ω in which the observed signal value is below the noise variance (i.e. $|Y_\omega|^2 < \eta^2$), the estimator acknowledges that K_ω cannot be recovered and outputs 0. Below we make usage of this point to derive a coarse-to-fine algorithm. In Fig. 10(b) we show the filter estimated using Eq. (30). The estimation nicely resembles the overall shape and power spectrum of the true filter (Fig. 10(a)) but is far too noisy to be acceptable. This noise is not surprising as every component of K was estimated from a single measurement.

The signal processing literature [12] addresses the problem of power spectrum estimation (also known as the periodigram), suggesting that the power spectrum of the observed signal Y should be smoothed before applying Eq. (30). While such smoothing operation increases the bias of the estimation, it significantly reduces its variance. Fig. 10(c) demonstrates the estimation from a smoothed power spectrum. One can note that as smoothing reduces the fluctuation in the frequency domain, the support of the filter in the primal domain becomes more compact. This leads to another important property of the problem that was ignored so far: while Eq. (30) estimate every Fourier coefficient independently, the number of free parameters to estimate in K is much smaller than the image size, since a typical filter is assumed to have only a small compact support. Fig. 10(d) presents the estimated kernel, once a compact support was enforced (according to the algorithm described below). This constraint significantly increases the stability of the estimation.

6.1. Phase estimation

While Eq. (30) defines the power spectrum of K, it leaves us with a complete ambiguity regarding its phase. In fact, for every solution K, X such that $Y_\omega = K_\omega X_\omega$ and for any phase vector θ_ω, the pair $\widetilde{K}_\omega = K_\omega e^{i\theta_\omega}, \widetilde{X}_\omega = X_\omega e^{-i\theta_\omega}$ is an equally valid solution, satisfying $Y_\omega = \widetilde{K}_\omega \widetilde{X}_\omega$. The prior on X does not help resolving this ambiguity – as the Gaussian prior in Eq. (25) depends only on the power spectrum, $p(\widetilde{X}) = p(X)$. However, while ev-



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

Figure 11. Coarse to fine kernel estimation. (a) Ground truth. (b-f) estimated kernels with decreasing η values

(a) Deconvolution with correct filter (b) Deconvolution with mirrored filter

Figure 12. Mirroring ambiguity with second order statistics

ery phase can maintain the convolution model, most random phase choices destroy the finite support of K. The question of estimating the signal phase given the power spectrum has a long history in signal processing. [8] states that for most of kernels, a finite support constraint uniquely defines the signal phase, up to (1) shift and (2) flipping (mirroring). While a shift ambiguity in deconvolution is reasonable and does not effect the visual quality of the deconvolved image, deconvolving the image with the mirrored filter leads to noticeable artifacts, as illustrated in Fig. 12. For the implementation in this paper we escape this ambiguity by noticing that while the original image x (in the spatial domain) be non negative, deconvolving y with the mirrored filter often leads to negative x values. Yet, this ambiguity highlights one of the weaknesses of second order statistics. While the second order statistics of the images in Fig. 12(a,b) are equal, it is clear that every simple sparse measure will favor Fig. 12(a). Nevertheless, we show that the second order statistics plus finite support constraint can get us surprisingly close to the true solution.

While a bounded support constraint removes most phase ambiguity, recovering the phase algorithmically is not a trivial question. A popular gradient based optimization scheme is the Gerchberg-Saxton [6, 5] algorithm. This algorithm initializes the kernel phase randomly, and then alternates between primal-frequency transformations, enforcing the finite support constraint in the primal domain and the required power spectrum in the frequency domain.

6.2. EM optimization

Applying the Gerchberg-Saxton algorithm [6, 5] to the independent power spectrum estimated from Eq. (29) provides a reasonable initialization for our algorithm. We then proceed with an EM algorithm. The E-step computed the expected mean and variance for the deblurred image X. The M-step uses the second order statistics of X to solve for k, enforcing two constraints: the finite support constraint discussed above, plus the simple requirement that the blur kernel k (in the spatial domain) is non negative.

E-step: Applying Eq. (28):

$$< X_\omega > = \left(|K_\omega|^2 + \frac{\eta^2}{\sigma_\omega^2}\right)^{-1} K_\omega^T Y_\omega \quad (31)$$

$$< X_\omega^T X_\omega > = \left(|K_\omega|^2 + \frac{\eta^2}{\sigma_\omega^2}\right) + < X_\omega >^T < X_\omega > \quad (32)$$

M-step Transform $< X >$ and $< XX >$ to the spatial domain and solve for k minimizing $< k \otimes x - y >$ subject to finite support and non negativity.

To express this minimization, suppose that k is an $l \times l$ filter. We denote by x_{w_i} the $l \times l$ window around the i'th pixel, such that $y_i = \sum_{j \in w_i} k_j x_j$. Let A be an $m \times l^2$ matrix whose rows are the windows x_{w_i}, and m is the number of windows included in the image. If x, y are known, the best filter k is the one minimizing

$$\|Ak(:) - y(:)\|^2 = k(:)^T A^T Ak(:) - 2y(:)^T Ak(:) + y(:)^T y(:) \quad s.t.\ k \geq 0. \quad (33)$$

Note that the number of unknowns in this system is equal to the kernel size l^2, which is much lower than the number of pixels in the image. In practice we do not precisely know x, but from the **E-stp** we have access to $< A^T A >$ and $< A >$.

This is a quadratic minimization subject to linear constraints, and thus a convex problem that can be solved using quadratic programming.

6.3. Coarse-to-fine

Fergus *et al*. [4] estimated the kernel in a coarse-to-fine scheme. In our case, Eq. (29) provides an easy way to implement this. We initialize the optimization with a high noise variance η. As a result all frequencies with observation below the noise variance (usually the high frequencies) are set to zero, and we mostly optimize the low frequencies of the kernel. Once the low frequency optimization starts to

converge we gradually reduce the noise variance η, allowing more and more bands of frequencies to be nailed down. The kernels estimated with varying η values are illustrated in Fig. 11.

7. Appendix B: Shan *et al.*'s algorithm

We discuss the blind deconvolution algorithm of [21] and try to understand how it is working. This algorithm attempts to optimize a semi-MAP$_{x,k}$ score, seeking a solution k, x that minimizes:

$$\lambda\|x-y\|^2 + \sum_i w_i|g_{x,i}(x)|^\alpha + w_i|g_{y,i}(x)|^\alpha. \qquad (34)$$

There are two main components that distinguish this algorithm from a naive MAP$_{x,k}$ optimization: edge reweighting and iterative update of the likelihood weight.

Edge rewighting: One main component that prevents Eq. (34) from outputting the delta solution is the usage of non uniform weights w_i on the gradient penalty. The authors explicitly detect low contrast image regions and increase their smoothness penalty.

To test this idea, we have implemented a simplified coordinate descent variant of the algorithm. We attempt to minimize the cost in Eq. (34), alternating between minimization with respect to x and minimization with respect to k (holding the other constant). We use $\alpha = 0.8$ for the sparse prior, and solve for x using iterative reweighted least squares, as in [15]. Gradients are reweighted using an edge detector. We emphasize that the goal of our implementation is to test the basic idea of a MAP$_{x,k}$ approach with edge reweighting, and not to reproduce the algorithm of [21] exactly. This algorithm involves a sophisticated number of additional details which affect the final output. Our observation is that while edge reweighting helps in avoiding the delta solution, edge rewighting alone is not always sufficient.

Iterative likelihood update: Another important component in [21] is to start the optimization with a low likelihood weight λ, and gradually increase it during subsequent iterations. To understand this, Fig. 13 shows an image deconvolved with two kernels - the true kernel and a delta kernel. We have performed the deconvolution with a set of λ values and compared the sum of gradients in the deconvolved image. Examining Fig. 13 we note that for low λ values, there is no need to explain all low contrast texture in y. These low contrast details are interpreted as noise, and the resulting latent image x is piecewise constant with step edges. Given the piecewise constant structure, the derivatives response is low. Therefore, for low λ values the true blur is indeed favored over the delta kernel. However, the situation is usually inverted when the likelihood weight is increased to a realistic level, and a delta kernel wins.

The fact that the true kernel is favored when the likelihood weight is low can help steer the algorithm toward

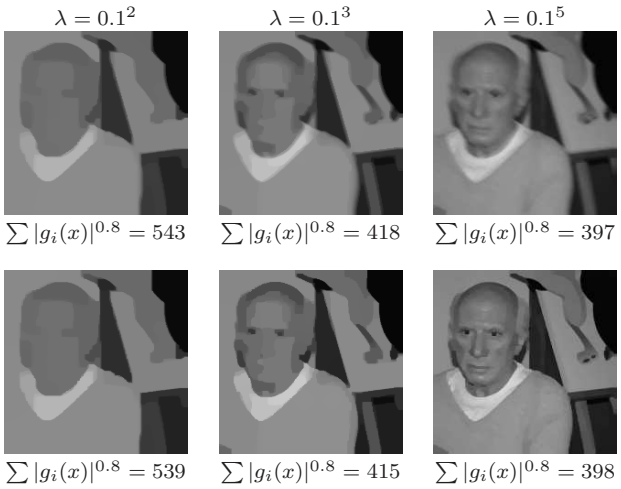
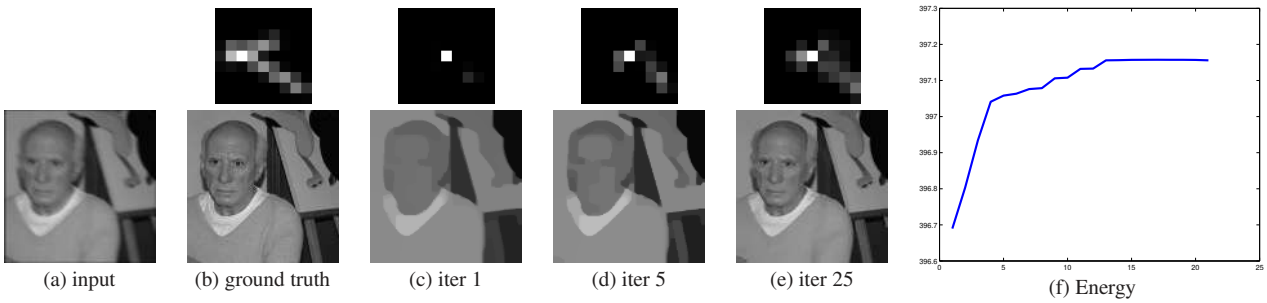


Figure 13. Non blind deconvolution using a delta kernel (top) and the true kernel (bottom), with increasing likelihood (data fitting term) weight. The estimated image is piecewise constant with low likelihood weight, while fine details are added as the weight increases. The true kernel achieves a lower score with low weight, but realistic likelihood weight is favoring the delta solution.

the desired solution. As suggested by [21], we have initialized our coordinate descent algorithm with a low λ value and gradually increased it during iterations. Since λ is initially low the algorithm is steered toward the true kernel and when λ is increased, the algorithm is already trapped in a local minimum and does not move significantly away from it. Some iterations from our coordinate descent implementation are available in Fig. 14. To evaluate this, Fig. 14(f) illustrates the likelihood changes during optimization. While λ is updated during optimization, at the end we traced back the kernels estimated in previous iterations, and evaluated their score using the final realistic (high) λ value. Fig. 14(f) plots the scores with this final λ. The interesting observation is that the score of the solution is increasing during optimization and the score of the first iteration (a delta kernel) is actually better than the final one. That is, by changing likelihood weight during optimization, the algorithm is steered toward a *local minimum* of the cost in Eq. (34), but this local minimum often happens to be the desired one.

As another way to evaluate this, we blurred the image in Figs.13,14 with a box kernel of width 13 pixels. We have computed the MAP$_{x,k}$ score for this image, varying two parameters: the kernel (running over box filters of size 1 to 15 pixels) and the likelihood weight λ. The 2D surfaces of scores is visualized in Fig. 15. Two ridges are observed, and one can also notice that while the minima with the delta solution is much lower, the ridge from the low λ values is leading toward the true kernel local minima, and not toward the delta solution.



(a) input (b) ground truth (c) iter 1 (d) iter 5 (e) iter 25 (f) Energy

Figure 14. coordinate descent Kernel optimization with an edge reweighted $\text{MAP}_{x,k}$ score. Likelihood weight is increased during optimization.

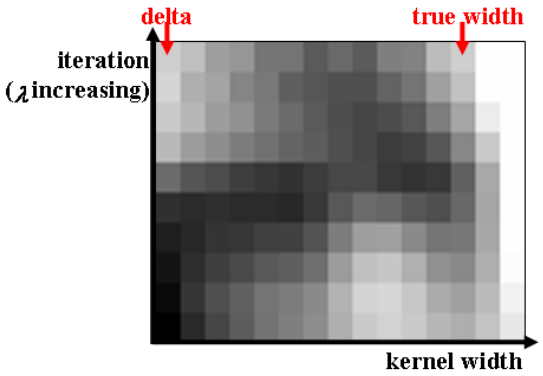


Figure 15. -$\text{MAP}_{x,k}$ scores as a function of likelihood weight λ and kernel width (dark values favored).

References

[1] G. R. Ayers and J. C. Dainty. Interative blind deconvolution method and its applications. *Opt. Lett.*, 1988.

[2] D. Brainard and W. Freeman. Bayesian color constancy. *JOSA*, 1997.

[3] M. M. Bronstein, A. M. Bronstein, M. Zibulevsky, and Y. Y. Zeevi. Blind deconvolution of images using optimal sparse representations. *Image Processing, IEEE Transactions on*, 14(6):726–736, 2005.

[4] R. Fergus, B. Singh, A. Hertzmann, S.T. Roweis, and W.T. Freeman. Removing camera shake from a single photograph. *SIGGRAPH*, 2006.

[5] J. R. Fienup. Phase retrieval algorithms: a comparison. *Applied Optics*, 21:2758–2769, August 1982.

[6] R.W. Gerchberg and W.O. Saxton. A practical algorithm for the determination of phase from image and diffraction plane pictures. *Optik*, 1972.

[7] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Prentice Hall, January 2002.

[8] M. Hayes. The reconstruction of a multidimensional sequence from the phase or magnitude of its fourier transform. *IEEE Trans. On Acoustics, Speech, and Signal Processing*, 1982.

[9] Jiaya Jia. Single image motion deblurring using transparency. In *CVPR*, 2007.

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

[11] A. K. Katsaggelos and K. T. Lay. Maximum likelihood blur identification and image restoration using the em algorithm. *IEEE Trans. Signal Processing*, 1991.

[12] S. M. Kay. *Fundamentals of Statistical Signal Processing: Estimation Theory*. Prentice Hall, 1997.

[13] D. Kundur and D. Hatzinakos. Blind image deconvolution. *IEEE Signal Processing Magazine*, 1996.

[14] R. G. Lane and R. H. T. Bates. Automatic multidimensional deconvolution. *J. Opt. Soc. Am. A*, 4(1):180–188, 1987.

[15] A. Levin, R. Fergus, F. Durand, and W. Freeman. Image and depth from a conventional camera with a coded aperture. *SIGGRAPH*, 2007.

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

[17] A. C. Likas and N. P. Galatsanos. A variational approach for bayesian blind image deconvolution. *IEEE Trans. on Signal Processing*, 2004.

[18] J. W. Miskin and D. J. C. MacKay. Ensemble learning for blind image separation and deconvolution. In *Advances in Independent Component Analysis*. Springer, 2000.

[19] R. Molina, A. K. Katsaggelos, J. Abad, and J. Mateos. A bayesian approach to blind deconvolution based on dirichlet distributions. In *ICASSP*, 1997.

[20] S. Roth and M.J. Black. Fields of experts: A framework for learning image priors. In *CVPR*, 2005.

[21] Q. Shan, J. Jia, and A. Agarwala. High-quality motion deblurring from a single image. *SIGGRAPH*, 2008.

[22] Q. Shan, W. Xiong, and J. Jia. Rotational motion deblurring of a rigid object from a single image. In *ICCV*, 2007.

[23] E P Simoncelli. Bayesian denoising of visual images in the wavelet domain. In *Bayesian Inference in Wavelet Based Models*. Springer-Verlag, New York, 1999.

[24] E. Thiébaut and J.-M. Conan. Strict a priori constraints for maximum-likelihood blind deconvolution. *J. Opt. Soc. Am. A*, 12(3):485–492, 1995.

[25] Y. Weiss and W. T. Freeman. What makes a good model of natural images? In *CVPR*, 2007.

input

Ground truth

SSD err=32.2, err ratio=1
Deconvolution with ground truth kernel

SSD err=38.4, err ratio=1.19
Fergus *et al*.

SSD err=83.5, err ratio=2.59
Shan *et al*.

SSD err=69.2, err ratio=2.14
Shan *et al*. kernel, sparse deconv

SSD err=188.5, err ratio=5.84
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=211.9, err ratio=6.57
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=162.5, err ratio=5.04
Gaussian

Figure 16. Comparing deconvolution algorithms, im 1, kernel 1

input

Ground truth

SSD err=37.0, err ratio=1
Deconvolution with ground truth kernel

SSD err=39.3, err ratio=1.06
Fergus *et al*.

SSD err=72.0, err ratio=1.94
Shan *et al*.

SSD err=53.6, err ratio=1.44
Shan *et al*. kernel, sparse deconv

SSD err=166.4, err ratio=4.49
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=264.7, err ratio=7.14
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=168.5, err ratio=4.54
Gaussian

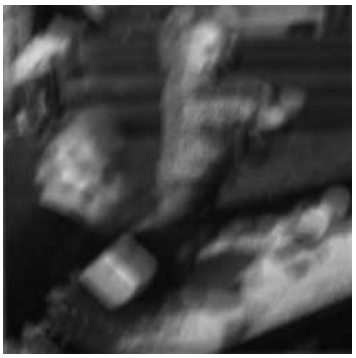
Figure 17. Comparing deconvolution algorithms, im 1, kernel 2

input | Ground truth | SSD err=25.3, err ratio=1
Deconvolution with ground truth kernel

SSD err=28.7, err ratio=1.13
Fergus *et al*. | SSD err=41.6, err ratio=1.64
Shan *et al*. | SSD err=38.7, err ratio=1.52
Shan *et al*. kernel, sparse deconv

SSD err=82.7, err ratio=3.26
$\mathrm{MAP}_{x,k}$, edges reweighting | SSD err=122.1, err ratio=4.81
$\mathrm{MAP}_{x,k}$, no edges reweighting | SSD err=39.6, err ratio=1.56
Gaussian

Figure 18. Comparing deconvolution algorithms, im 1, kernel 3



input

Ground truth

SSD err=59.9, err ratio=1
Deconvolution with ground truth kernel

SSD err=135.5, err ratio=2.26
Fergus *et al*.

SSD err=604.5, err ratio=10.08
Shan *et al*.

SSD err=583.2, err ratio=9.72
Shan *et al*. kernel, sparse deconv

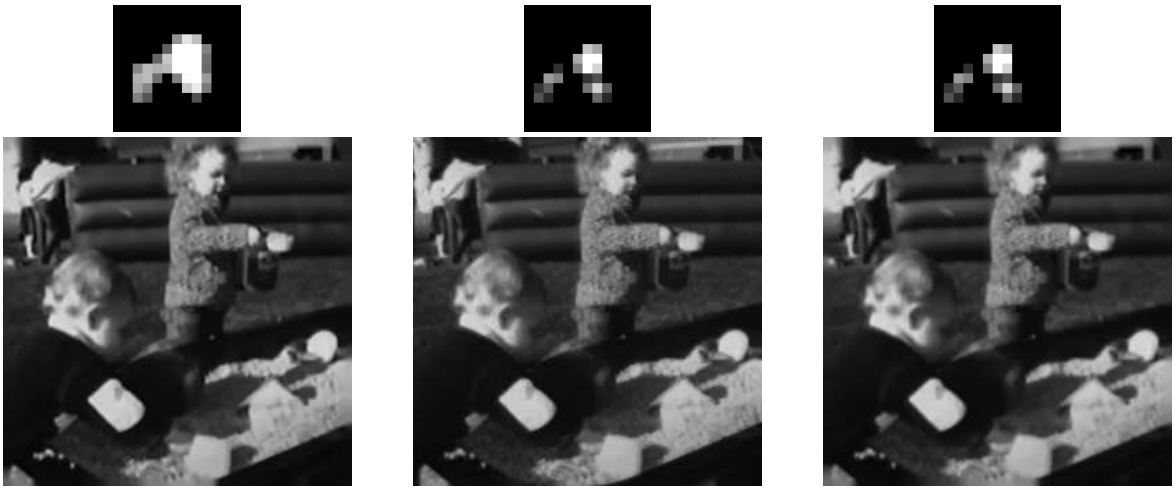
SSD err=497.6, err ratio=8.29
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=645.4, err ratio=10.76
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=315.7, err ratio=5.26
Gaussian

Figure 19. Comparing deconvolution algorithms, im 1, kernel 4

input | Ground truth | SSD err=20.7, err ratio=1
Deconvolution with ground truth kernel



SSD err=27.1, err ratio=1.30
Fergus *et al*. | SSD err=45.8, err ratio=2.21
Shan *et al*. | SSD err=40.6, err ratio=1.96
Shan *et al*. kernel, sparse deconv

SSD err=64.3, err ratio=3.10
$\mathrm{MAP}_{x,k}$, edges reweighting | SSD err=110.5, err ratio=5.33
$\mathrm{MAP}_{x,k}$, no edges reweighting | SSD err=53.78, err ratio=2.59
Gaussian

Figure 20. Comparing deconvolution algorithms, im 1, kernel 5

input

Ground truth

SSD err=15.9, err ratio=1
Deconvolution with ground truth kernel

SSD err=44.45, err ratio=2.79
Fergus *et al*.

SSD err=104.8, err ratio=6.58
Shan *et al*.

SSD err=94.0, err ratio=5.90
Shan *et al*. kernel, sparse deconv

SSD err=59.4, err ratio=3.73
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=202.4, err ratio=12.71
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=172.9, err ratio=10.8
Gaussian

Figure 21. Comparing deconvolution algorithms, im 1, kernel 6

input

Ground truth

SSD err=24.3, err ratio=1
Deconvolution with ground truth kernel

SSD err=212.3, err ratio=8.73
Fergus *et al*.

SSD err=414.1, err ratio=17.04
Shan *et al*.

SSD err=401.0, err ratio=16.5
Shan *et al*. kernel, sparse deconv

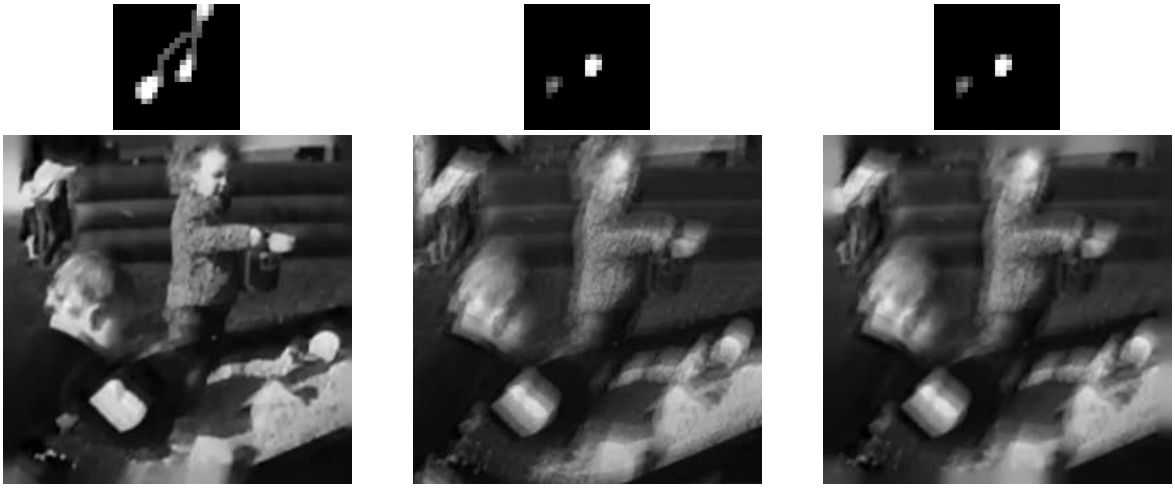
SSD err=103, err ratio=4.2
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=412, err ratio=16.9
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=376, err ratio=15.5
Gaussian

Figure 22. Comparing deconvolution algorithms, im 1, kernel 7

input | Ground truth | SSD err=30, err ratio=1
Deconvolution with ground truth kernel



SSD err=53, err ratio=1.7
Fergus *et al*. | SSD err=458, err ratio=15.2
Shan *et al*. | SSD err=450, err ratio=15.0
Shan *et al*. kernel, sparse deconv

SSD err=327, err ratio=10.8
$\mathrm{MAP}_{x,k}$, edges reweighting | SSD err=458, err ratio=15.2
$\mathrm{MAP}_{x,k}$, no edges reweighting | SSD err=559, err ratio=18.6
Gaussian

Figure 23. Comparing deconvolution algorithms, im 1, kernel 8

input

Ground truth

SSD err=43, err ratio=1
Deconvolution with ground truth kernel

SSD err=55, err ratio=1.2
Fergus *et al*.

SSD err=162, err ratio=3.6
Shan *et al*.

SSD err=150, err ratio=3.4
Shan *et al*. kernel, sparse deconv

SSD err=248, err ratio=5.6
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=272, err ratio=6.1
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=79, err ratio=1.8
Gaussian

Figure 24. Comparing deconvolution algorithms, im 2, kernel 1

input

Ground truth

SSD err=50.6, err ratio=1
Deconvolution with ground truth kernel

SSD err=64.3, err ratio=1.2
Fergus *et al*.

SSD err=191, err ratio=3.7
Shan *et al*.

SSD err=175, err ratio=3.4
Shan *et al*. kernel, sparse deconv

SSD err=343, err ratio=6.7
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=348, err ratio=6.8
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=164, err ratio=3.2
Gaussian

Figure 25. Comparing deconvolution algorithms, im 2, kernel 2

input

Ground truth

SSD err=40, err ratio=1
Deconvolution with ground truth kernel

SSD err=52.7, err ratio=1.3
Fergus *et al*.

SSD err=88, err ratio=2.1
Shan *et al*.

SSD err=84, err ratio=2.0
Shan *et al*. kernel, sparse deconv

SSD err=169, err ratio=4.1
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=185, err ratio=4.5
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=129, err ratio=3.1
Gaussian

Figure 26. Comparing deconvolution algorithms, im 2, kernel 3

input

Ground truth

SSD err=79, err ratio=1
Deconvolution with ground truth kernel

SSD err=123, err ratio=1.5
Fergus *et al*.

SSD err=195, err ratio=2.4
Shan *et al*.

SSD err=182, err ratio=2.3
Shan *et al*. kernel, sparse deconv

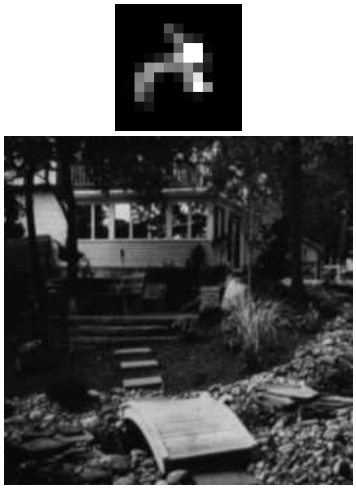
SSD err=481, err ratio=6.1
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=574, err ratio=7.26
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=189, err ratio=2.4
Gaussian

Figure 27. Comparing deconvolution algorithms, im 2, kernel 4

input



Ground truth

SSD err=26, err ratio=1
Deconvolution with ground truth kernel

SSD err=38, err ratio=1.4
Fergus *et al*.

SSD err=106, err ratio=3.9
Shan *et al*.

SSD err=100, err ratio=3.7
Shan *et al*. kernel, sparse deconv

SSD err=141, err ratio=5.2
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=161, err ratio=6
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=89, err ratio=3.3
Gaussian

Figure 28. Comparing deconvolution algorithms, im 2, kernel 5

input

Ground truth

SSD err=20, err ratio=1
Deconvolution with ground truth kernel

SSD err=84, err ratio=4.2
Fergus *et al*.

SSD err=198, err ratio=9.9
Shan *et al*.

SSD err=186, err ratio=9.3
Shan *et al*. kernel, sparse deconv

SSD err=227, err ratio=11.3
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=260, err ratio=13
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=84, err ratio=4.2
Gaussian

Figure 29. Comparing deconvolution algorithms, im 2, kernel 6

input

Ground truth

SSD err=39, err ratio=1
Deconvolution with ground truth kernel

SSD err=153, err ratio=3.8
Fergus *et al*.

SSD err=322, err ratio=8.2
Shan *et al*.

SSD err=315, err ratio=8.0
Shan *et al*. kernel, sparse deconv

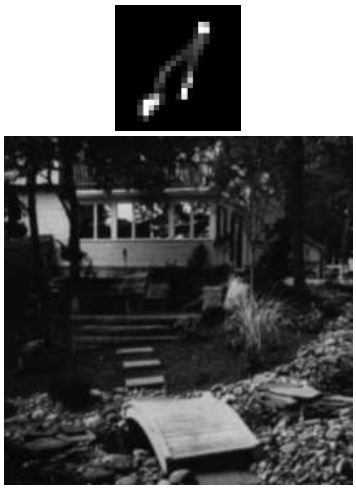
SSD err=266, err ratio=6.7
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=551, err ratio=14
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=296, err ratio=7.5
Gaussian

Figure 30. Comparing deconvolution algorithms, im 2, kernel 7

input



Ground truth

SSD err=43, err ratio=1
Deconvolution with ground truth kernel

SSD err=92, err ratio=2.1
Fergus *et al*.

SSD err=362, err ratio=8.2
Shan *et al*.

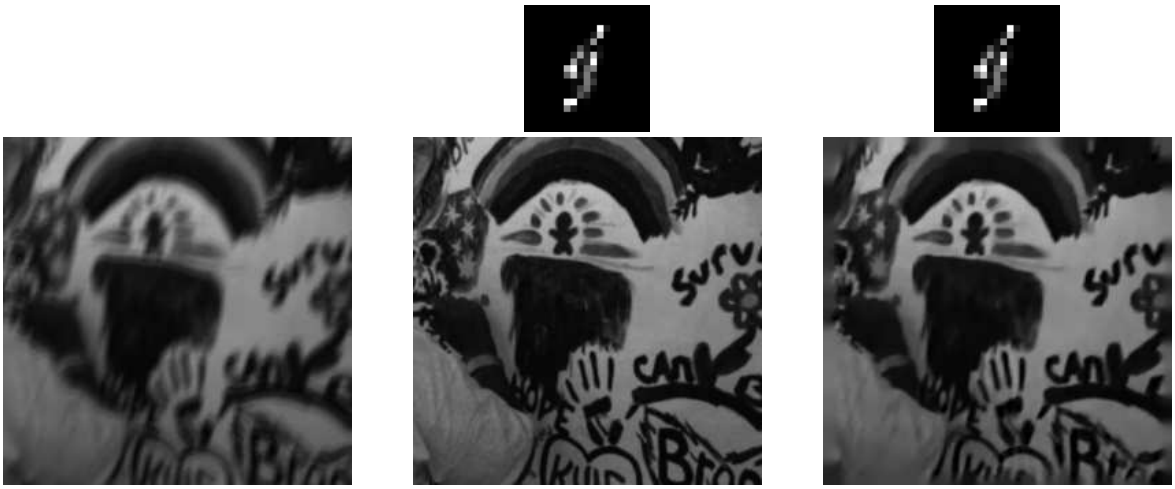
SSD err=513, err ratio=11.7
Shan *et al*. kernel, sparse deconv

SSD err=421, err ratio=9.6
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=522, err ratio=11.9
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=161, err ratio=3.6
Gaussian

Figure 31. Comparing deconvolution algorithms, im 2, kernel 8



input

Ground truth

SSD err=31.2, err ratio=1
Deconvolution with ground truth kernel

SSD err=37.4, err ratio=1.2
Fergus *et al*.

SSD err=99.9, err ratio=3.2
Shan *et al*.

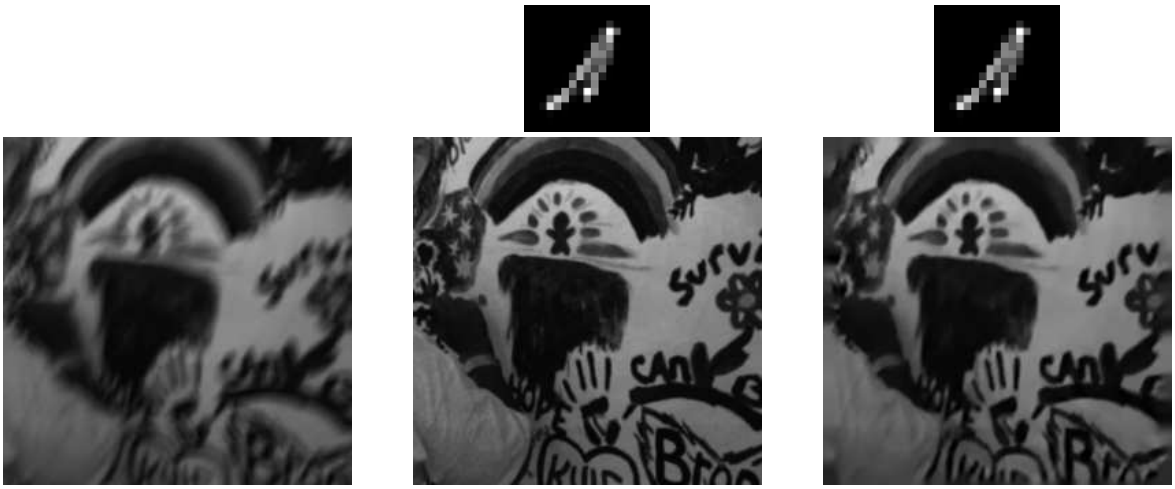
SSD err=83.5, err ratio=2.6
Shan *et al*. kernel, sparse deconv

SSD err=101.4, err ratio=3.2
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=211.2, err ratio=6.7
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=110.5, err ratio=3.5
Gaussian

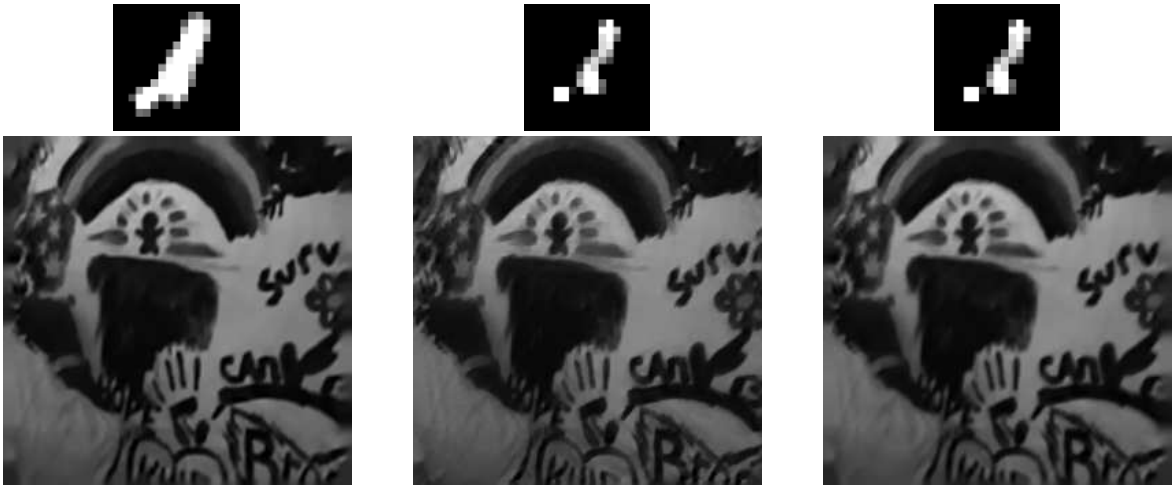
Figure 1. Comparing deconvolution algorithms, im 3, kernel 1



input

Ground truth

SSD err=35.3, err ratio=1
Deconvolution with ground truth kernel



SSD err=39.1, err ratio=1.1
Fergus *et al*.

SSD err=91.9, err ratio=2.6
Shan *et al*.

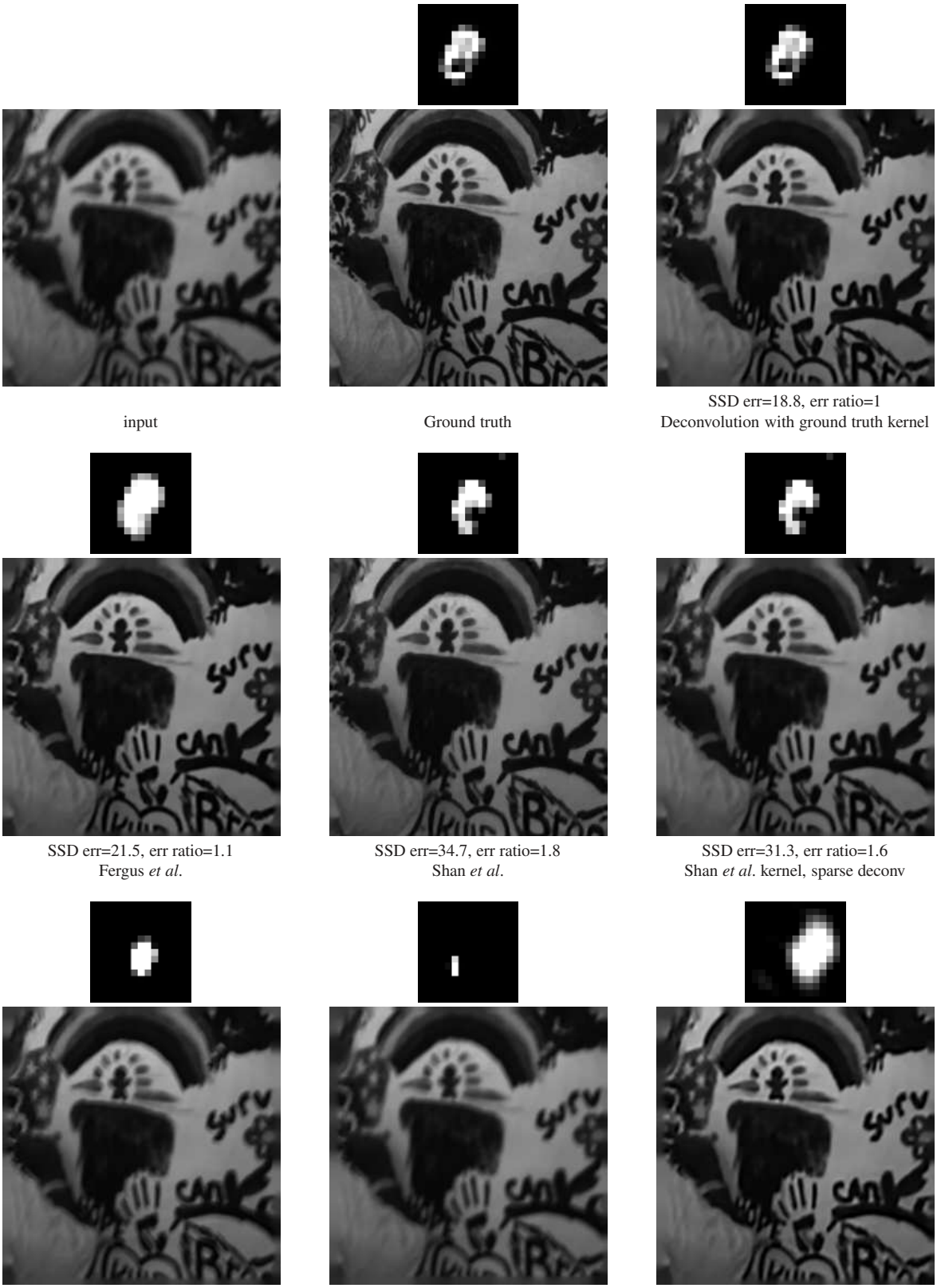
SSD err=64.6, err ratio=1.8
Shan *et al*. kernel, sparse deconv

SSD err=254.2, err ratio=7.1
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=287.9, err ratio=8.1
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=223.9, err ratio=6.3
Gaussian

Figure 2. Comparing deconvolution algorithms, im 3, kernel 2



input

Ground truth

SSD err=18.8, err ratio=1
Deconvolution with ground truth kernel

SSD err=21.5, err ratio=1.1
Fergus *et al*.

SSD err=34.7, err ratio=1.8
Shan *et al*.

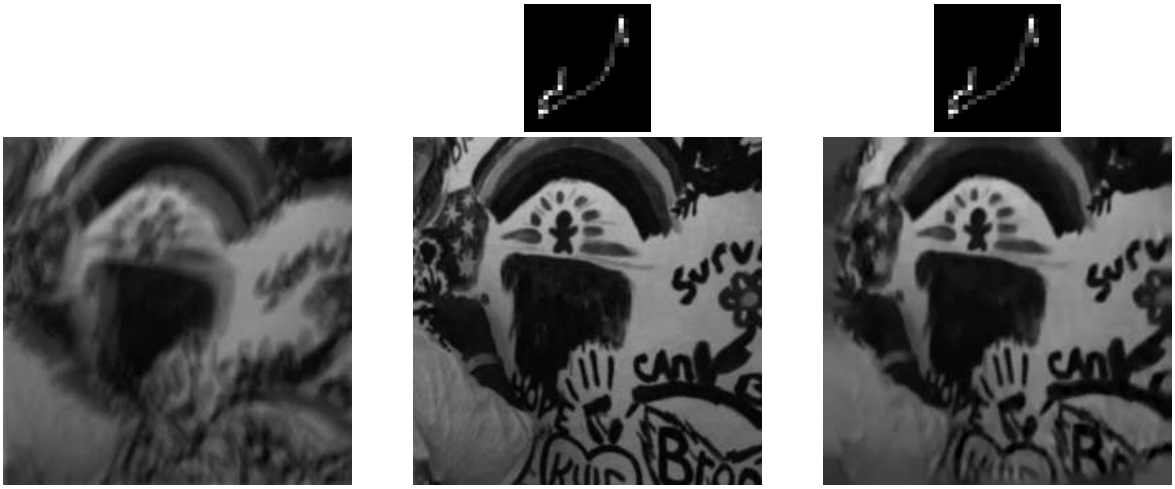
SSD err=31.3, err ratio=1.6
Shan *et al*. kernel, sparse deconv

SSD err=95.4, err ratio=5.1
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=115.0, err ratio=6.1
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=39.8, err ratio=2.1
Gaussian

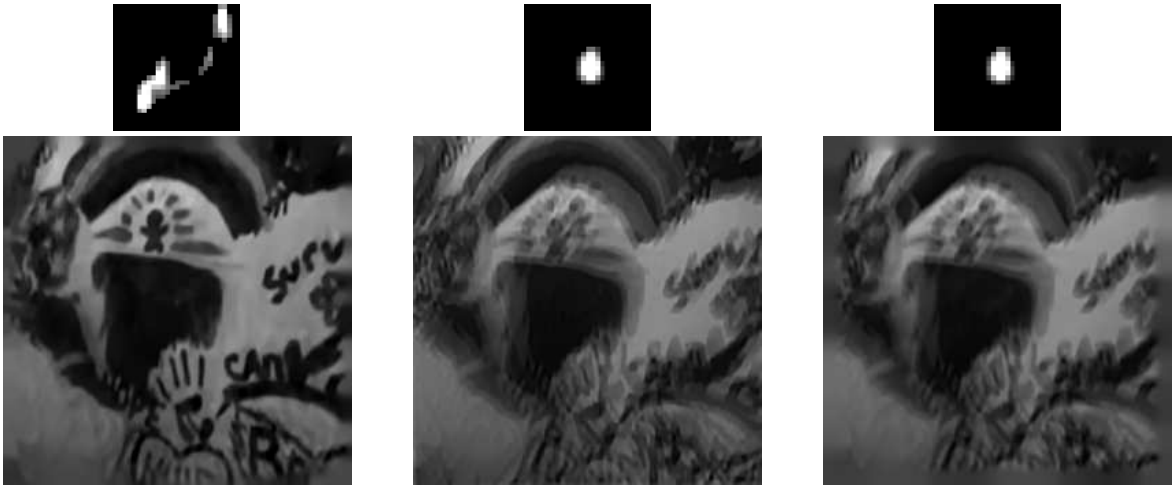
Figure 3. Comparing deconvolution algorithms, im 3, kernel 3



input

Ground truth

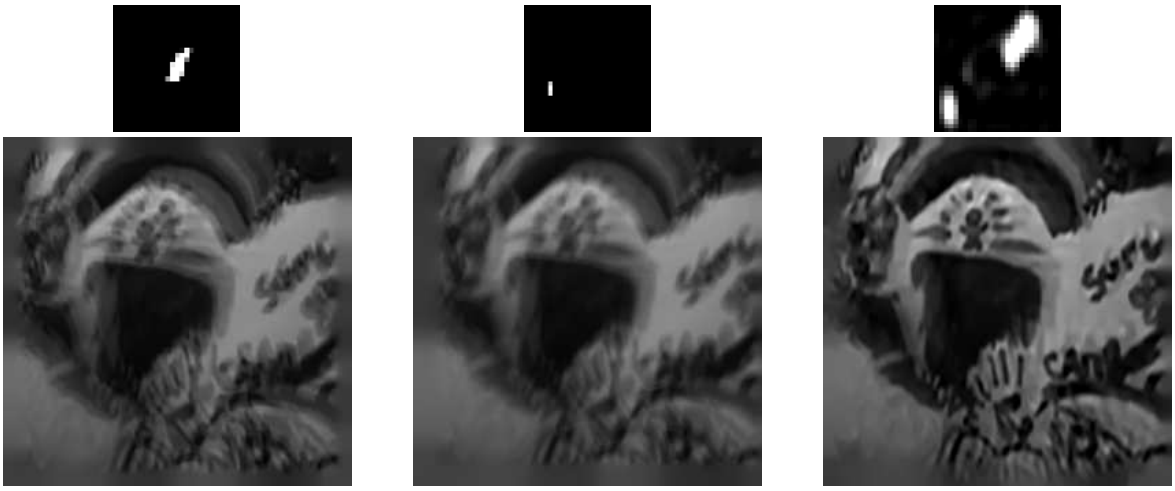
SSD err=45.2, err ratio=1
Deconvolution with ground truth kernel



SSD err=87.5, err ratio=1.9
Fergus *et al*.

SSD err=601.4, err ratio=13.3
Shan *et al*.

SSD err=580.6, err ratio=12.8
Shan *et al*. kernel, sparse deconv

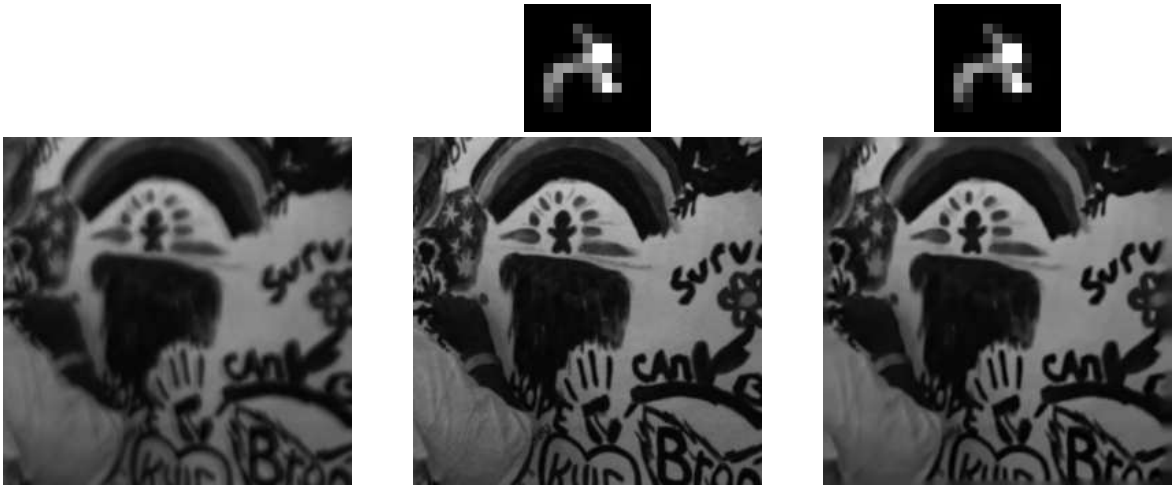


SSD err=596.0, err ratio=13.2
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=589.9, err ratio=13.0
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=204.6, err ratio=4.5
Gaussian

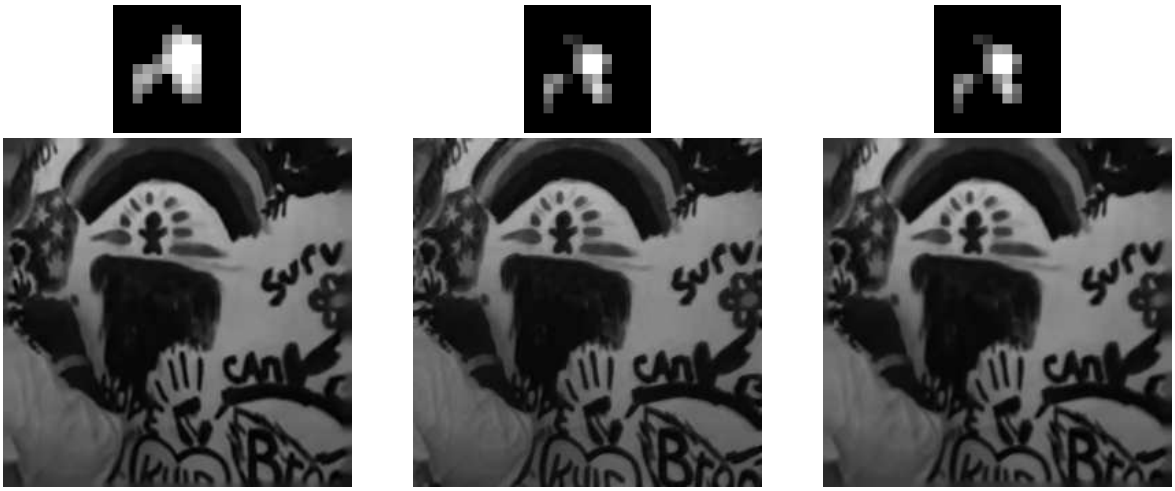
Figure 4. Comparing deconvolution algorithms, im 3, kernel 4



input

Ground truth

SSD err=15.2, err ratio=1
Deconvolution with ground truth kernel



SSD err=22.0, err ratio=1.4
Fergus *et al*.

SSD err=39.9, err ratio=2.6
Shan *et al*.

SSD err=33.7, err ratio=2.2
Shan *et al*. kernel, sparse deconv

SSD err=84.09, err ratio=5.5
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=113.3, err ratio=7.4
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=50.6, err ratio=3.3
Gaussian

Figure 5. Comparing deconvolution algorithms, im 3, kernel 5

input

Ground truth

SSD err=10.6, err ratio=1
Deconvolution with ground truth kernel

SSD err=33.6, err ratio=3.1
Fergus *et al*.

SSD err=84.9, err ratio=7.9
Shan *et al*.

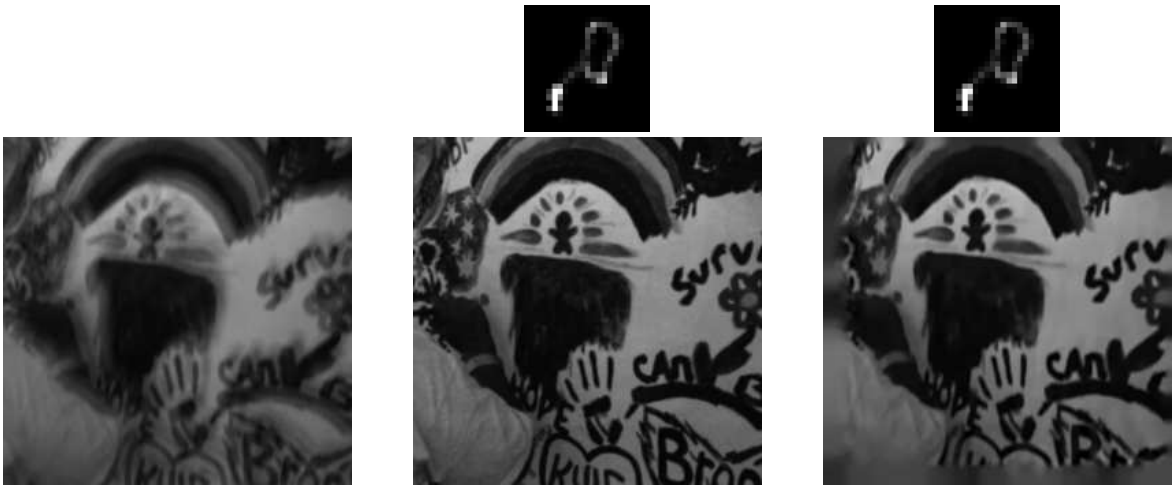
SSD err=71.2, err ratio=6.6
Shan *et al*. kernel, sparse deconv

SSD err=156.1, err ratio=14.6
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=209.6, err ratio=19.6
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=80, err ratio=7.5
Gaussian

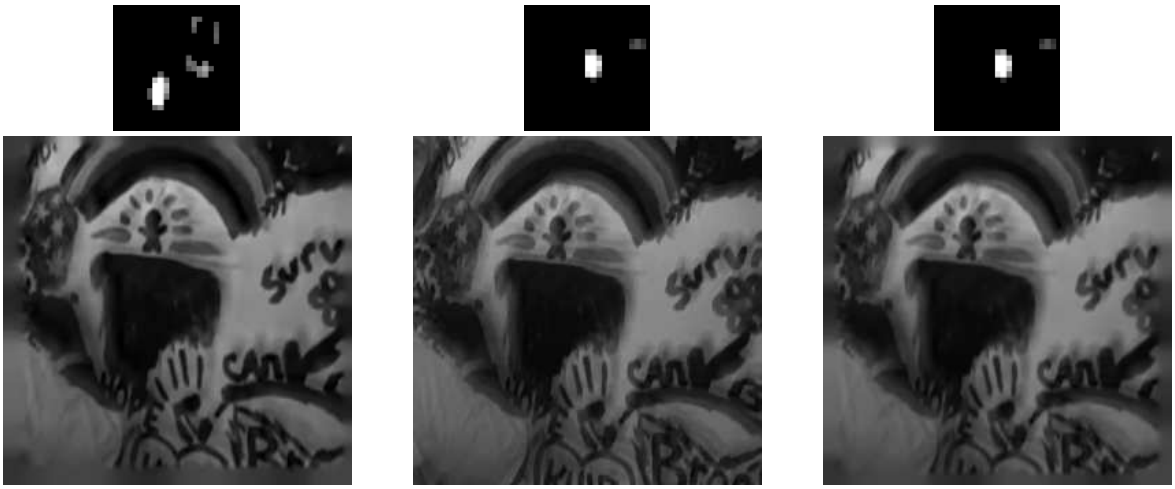
Figure 6. Comparing deconvolution algorithms, im 3, kernel 6



input

Ground truth

SSD err=16.9, err ratio=1
Deconvolution with ground truth kernel



SSD err=139.8, err ratio=8.2
Fergus *et al*.

SSD err=326.5, err ratio=19.2
Shan *et al*.

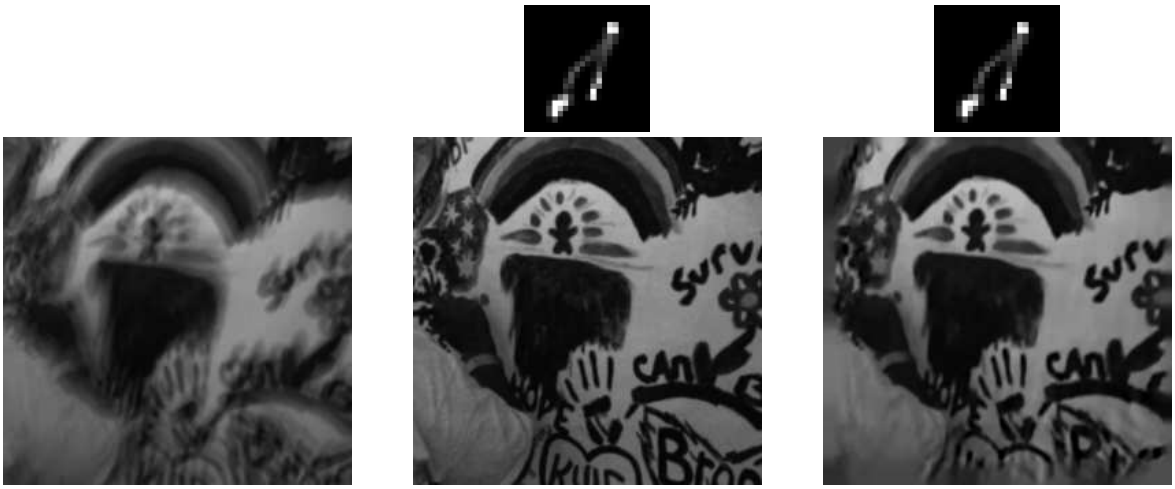
SSD err=315.3, err ratio=18.6
Shan *et al*. kernel, sparse deconv

SSD err=237.5, err ratio=14.0
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=394.8, err ratio=23.3
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=175.0, err ratio=10.3
Gaussian

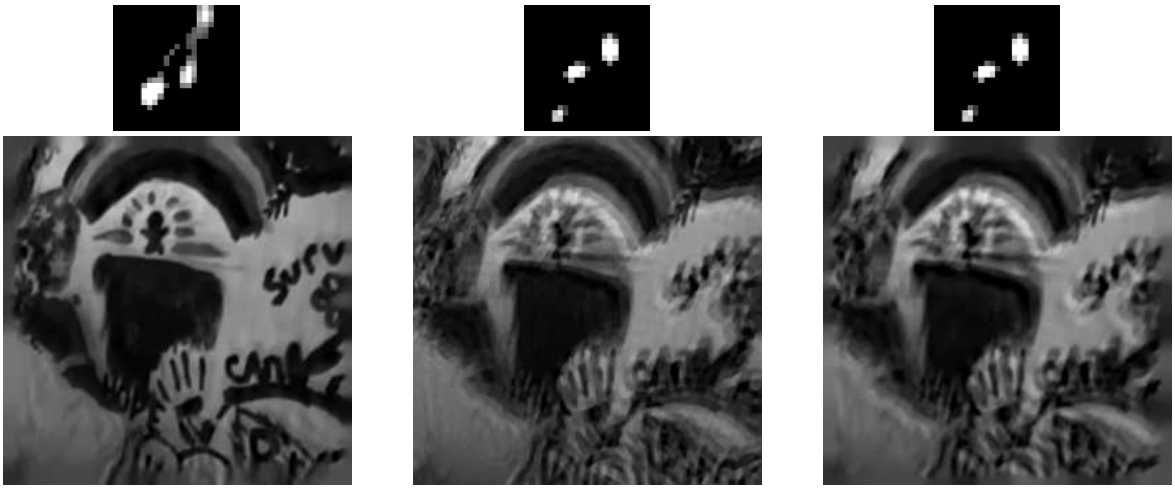
Figure 7. Comparing deconvolution algorithms, im 3, kernel 7



input

Ground truth

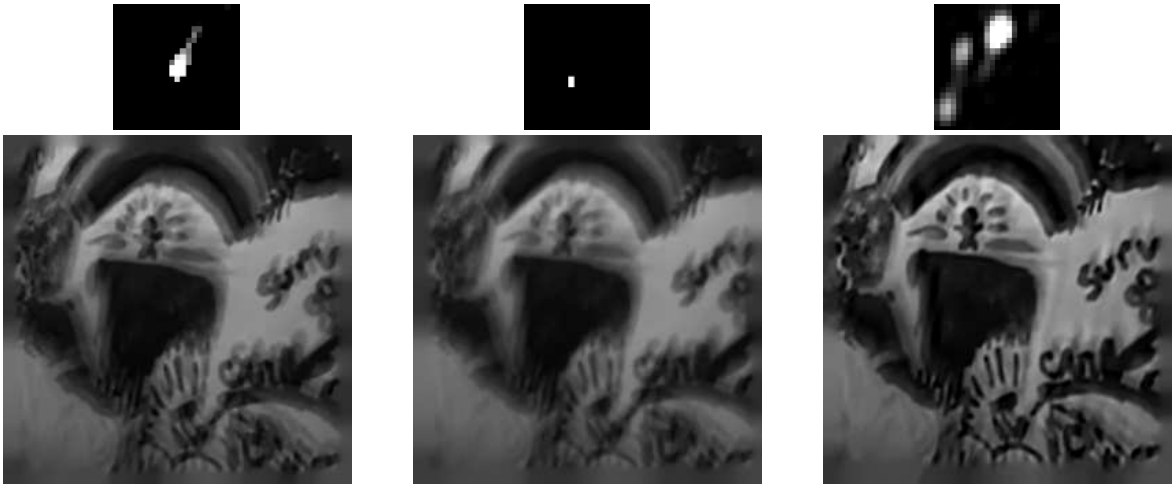
SSD err=29.9, err ratio=1
Deconvolution with ground truth kernel



SSD err=57.6, err ratio=1.9
Fergus *et al*.

SSD err=462.8, err ratio=15.4
Shan *et al*.

SSD err=515.4, err ratio=17.2
Shan *et al*. kernel, sparse deconv

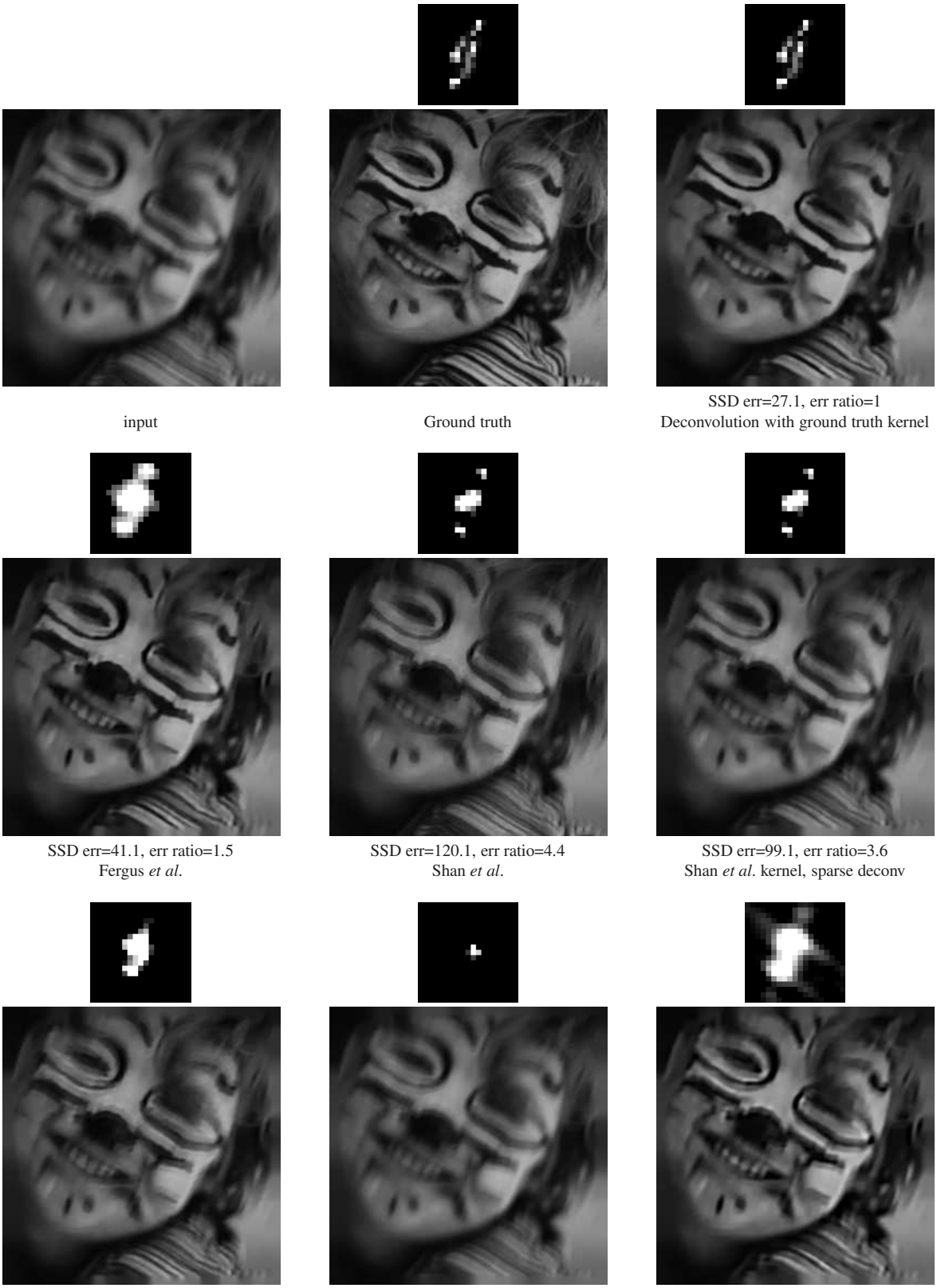


SSD err=430.3, err ratio=14.4
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=490.5, err ratio=16.4
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=197.1, err ratio=6.5
Gaussian

Figure 8. Comparing deconvolution algorithms, im 3, kernel 8



input

Ground truth

SSD err=27.1, err ratio=1
Deconvolution with ground truth kernel

SSD err=41.1, err ratio=1.5
Fergus *et al*.

SSD err=120.1, err ratio=4.4
Shan *et al*.

SSD err=99.1, err ratio=3.6
Shan *et al*. kernel, sparse deconv

SSD err=116.7, err ratio=4.3
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=173.3, err ratio=6.3
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=113.7, err ratio=4.1
Gaussian

Figure 9. Comparing deconvolution algorithms, im 4, kernel 1

input

Ground truth

SSD err=41.5, err ratio=1
Deconvolution with ground truth kernel

SSD err=92.8, err ratio=2.2
Fergus *et al*.

SSD err=204.3, err ratio=4.9
Shan *et al*.

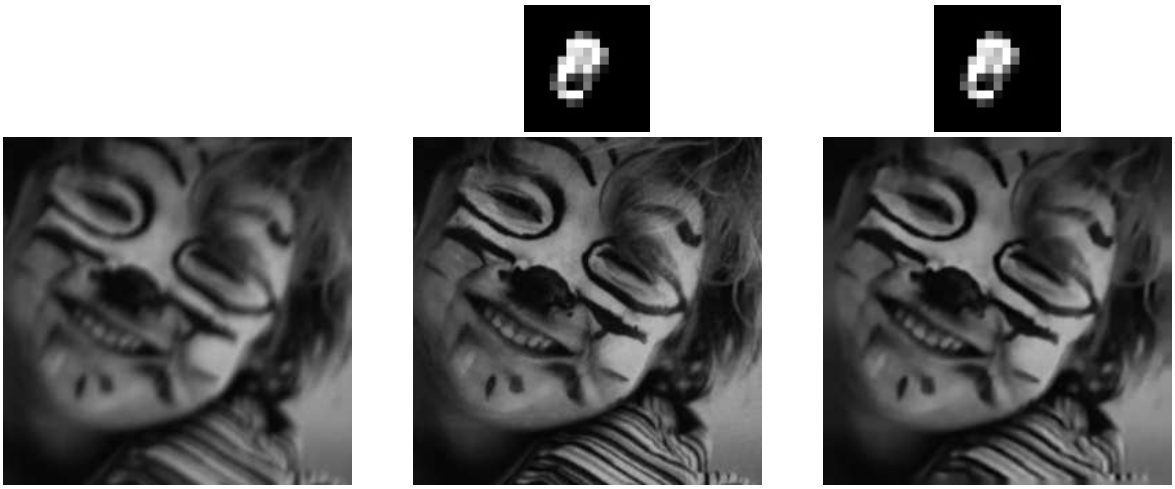
SSD err=180.8, err ratio=4.3
Shan *et al*. kernel, sparse deconv

SSD err=240.2, err ratio=5.7
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=244.1, err ratio=5.8
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=120.7, err ratio=2.9
Gaussian

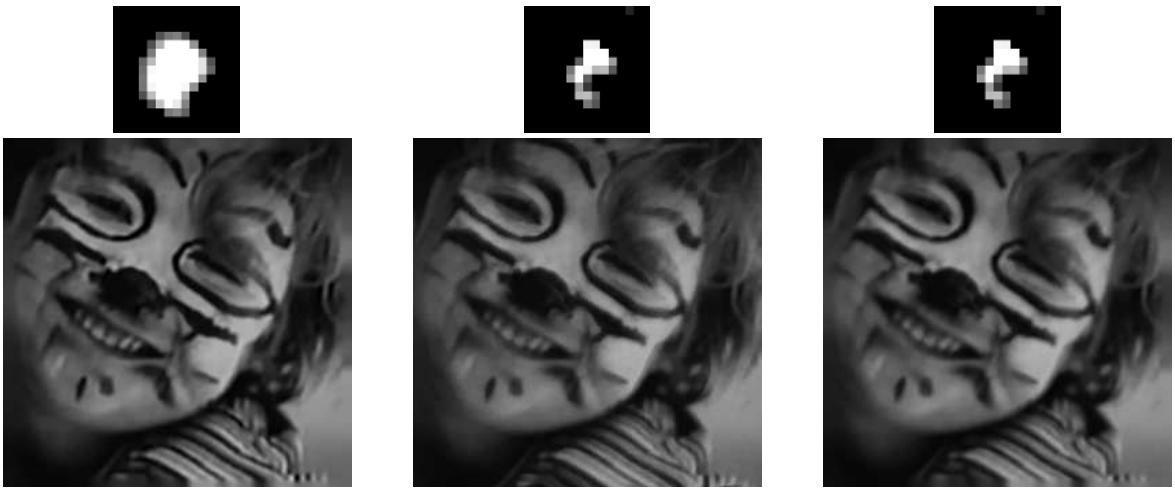
Figure 10. Comparing deconvolution algorithms, im 4, kernel 2



input

Ground truth

SSD err=14.5, err ratio=1
Deconvolution with ground truth kernel



SSD err=18.1, err ratio=1.2
Fergus *et al*.

SSD err=40.8, err ratio=2.8
Shan *et al*.

SSD err=33.6, err ratio=2.3
Shan *et al*. kernel, sparse deconv

SSD err=65.9, err ratio=4.5
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=89.3, err ratio=6.1
$\mathrm{MAP}_{x,k}$, no edges reweighting

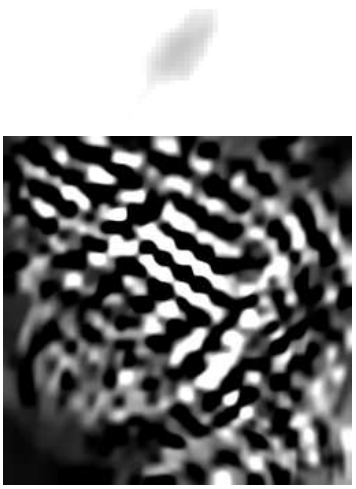
SSD err=68.2, err ratio=4.6
Gaussian

Figure 11. Comparing deconvolution algorithms, im 4, kernel 3

input

Ground truth

SSD err=42.0, err ratio=1
Deconvolution with ground truth kernel



SSD err=13,251, err ratio=316.8
Fergus *et al*.

SSD err=457, err ratio=10.9
Shan *et al*.

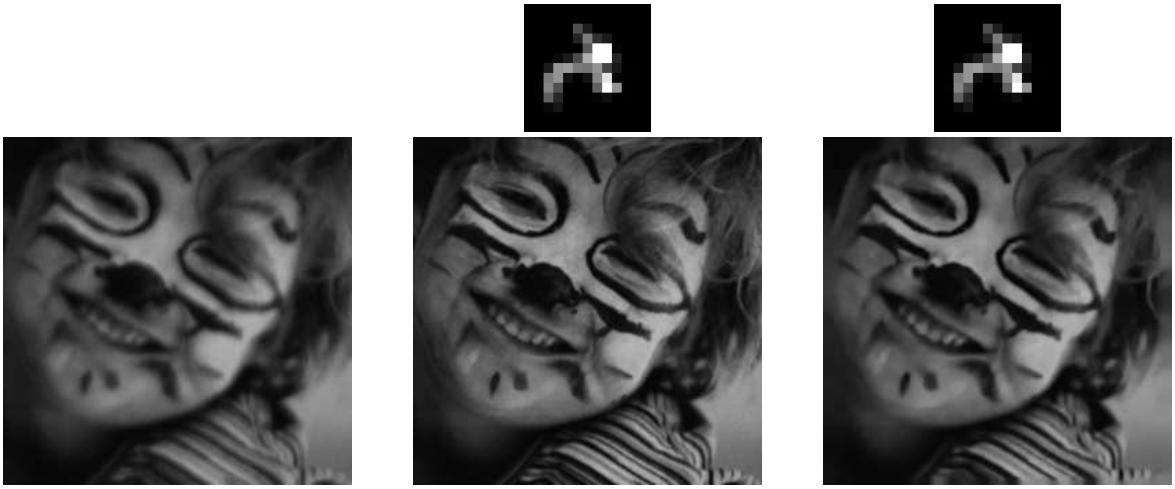
SSD err=430, err ratio=10.3
Shan *et al*. kernel, sparse deconv

SSD err=425, err ratio=10.1
$\text{MAP}_{x,k}$, edges reweighting

SSD err=806, err ratio=19.3
$\text{MAP}_{x,k}$, no edges reweighting

SSD err=124, err ratio=2.9
Gaussian

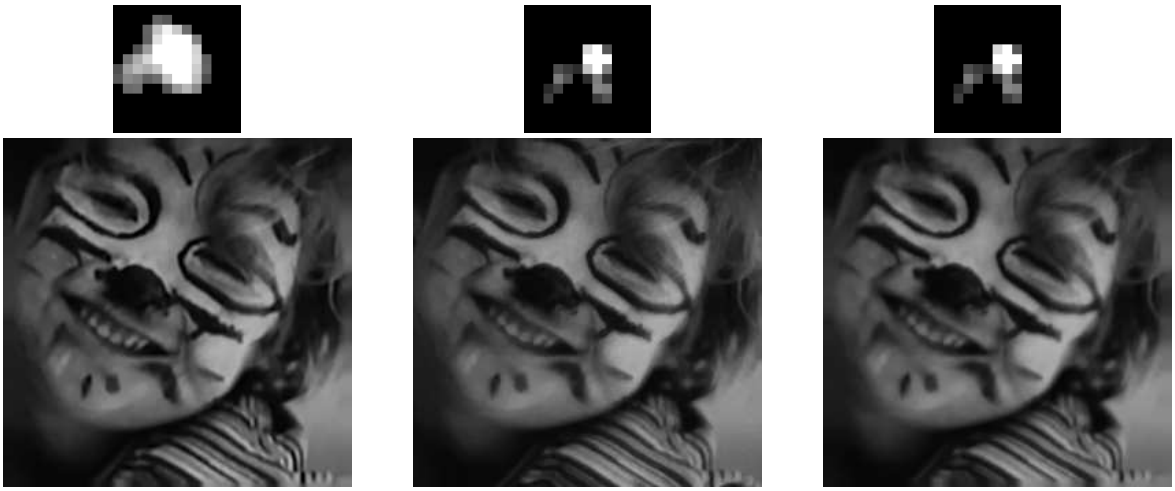
Figure 12. Comparing deconvolution algorithms, im 4, kernel 4



input

Ground truth

SSD err=15.3, err ratio=1
Deconvolution with ground truth kernel



SSD err=20.0, err ratio=1.3
Fergus *et al*.

SSD err=44.5, err ratio=2.9
Shan *et al*.

SSD err=35.8, err ratio=2.3
Shan *et al*. kernel, sparse deconv

SSD err=55.1, err ratio=3.6
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=81.5, err ratio=5.3
$\mathrm{MAP}_{x,k}$, no edges reweighting

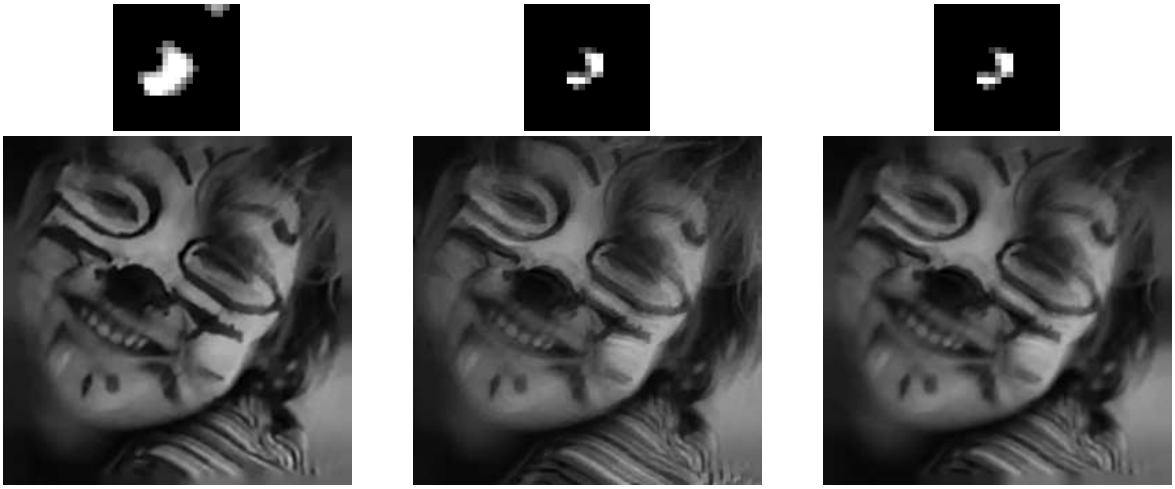
SSD err=40.9, err ratio=2.7
Gaussian

Figure 13. Comparing deconvolution algorithms, im 8, kernel 5

input

Ground truth

SSD err=18.6, err ratio=1
Deconvolution with ground truth kernel



SSD err=46.7, err ratio=2.5
Fergus *et al*.

SSD err=138.9, err ratio=7.4
Shan *et al*.

SSD err=121.4, err ratio=6.5
Shan *et al*. kernel, sparse deconv

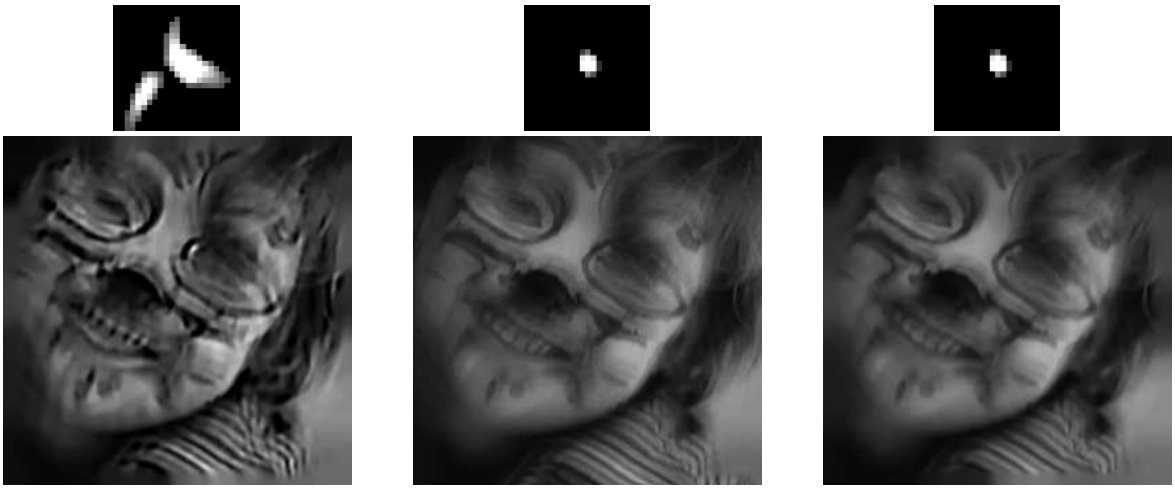
SSD err=132.8, err ratio=7.1
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=176.1, err ratio=9.4
$\mathrm{MAP}_{x,k}$, no edges reweighting

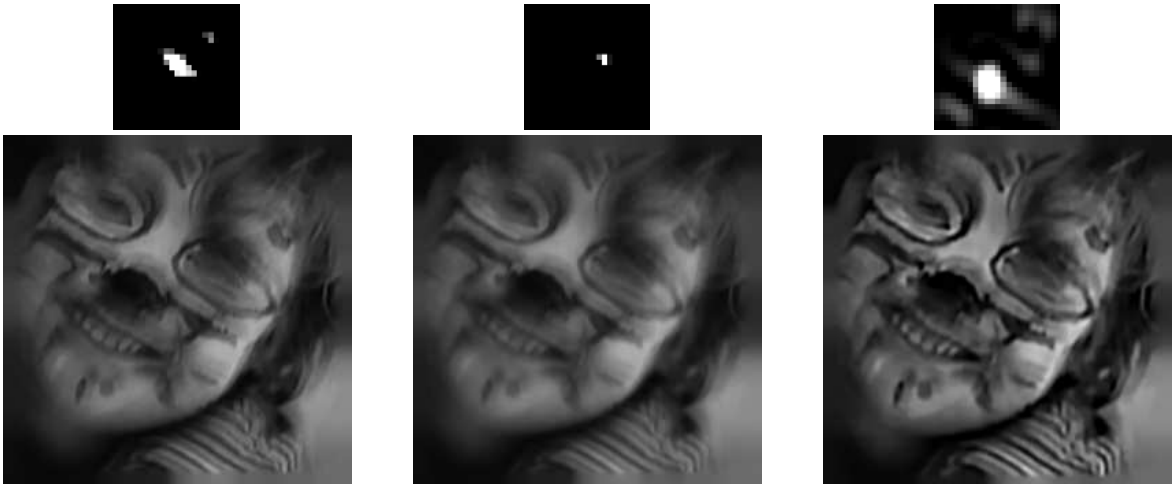
SSD err=89.5, err ratio=4.8
Gaussian

Figure 14. Comparing deconvolution algorithms, im 4, kernel 6

input | Ground truth | SSD err=16.3, err ratio=1
Deconvolution with ground truth kernel



SSD err=504.3, err ratio=30.9
Fergus *et al*. | SSD err=333.6, err ratio=20.4
Shan *et al*. | SSD err=318.8, err ratio=19.5
Shan *et al*. kernel, sparse deconv



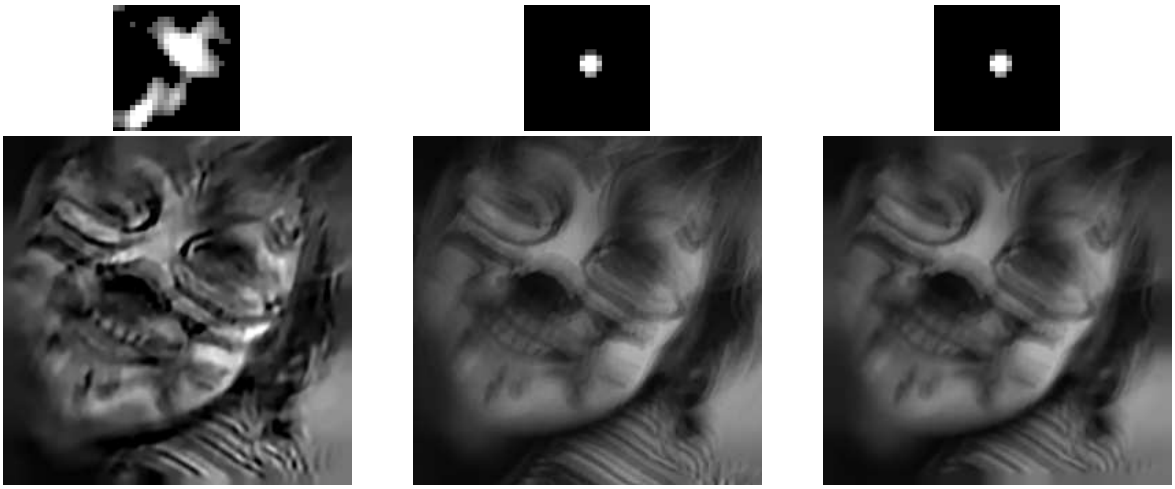
SSD err=288.2, err ratio=17.6
$\mathrm{MAP}_{x,k}$, edges reweighting | SSD err=342.9, err ratio=21.0
$\mathrm{MAP}_{x,k}$, no edges reweighting | SSD err=301.6, err ratio=18.5
Gaussian

Figure 15. Comparing deconvolution algorithms, im 4, kernel 7

input

Ground truth

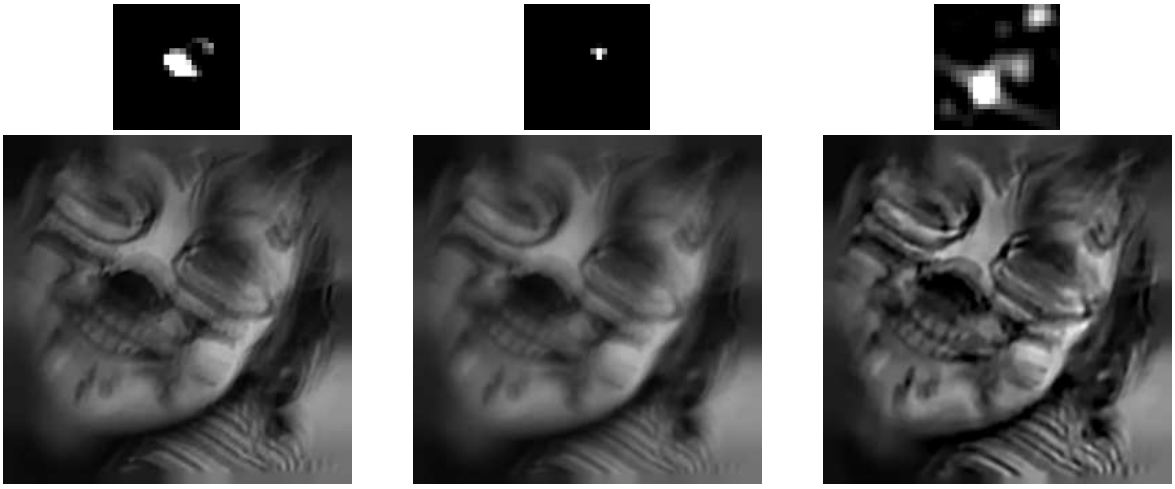
SSD err=27.6, err ratio=1
Deconvolution with ground truth kernel



SSD err=786.5, err ratio=28.4
Fergus *et al*.

SSD err=392.3, err ratio=14.1
Shan *et al*.

SSD err=377.9, err ratio=13.6
Shan *et al*. kernel, sparse deconv



SSD err=345.7, err ratio=12.5
$\mathrm{MAP}_{x,k}$, edges reweighting

SSD err=393.1, err ratio=14.2
$\mathrm{MAP}_{x,k}$, no edges reweighting

SSD err=524.5, err ratio=18.9
Gaussian

Figure 16. Comparing deconvolution algorithms, im 4, kernel 8