Blind Deblurring Using Internal Patch Recurrence

Tomer Michaeli and Michal Irani

Dept. of Computer Science and Applied Mathematics
Weizmann Institute of Science, Israel

Abstract. Recurrence of small image patches across different scales of a natural image has been previously used for solving ill-posed problems (*e.g.,* super-resolution from a single image). In this paper we show how this multi-scale property can also be used for "blind-deblurring", namely, removal of an unknown blur from a blurry image. While patches repeat 'as is' across scales in a *sharp* natural image, this cross-scale recurrence significantly diminishes in blurry images. We exploit these *deviations from ideal patch recurrence* as a cue for recovering the underlying (unknown) blur kernel. More specifically, we look for the blur kernel k, such that if its effect is *"undone"* (if the blurry image is deconvolved with k), the patch similarity across scales of the image will be maximized. We report extensive experimental evaluations, which indicate that our approach compares favorably to state-of-the-art blind deblurring methods, and in particular, is more robust than them.

Keywords: Blind deblurring, blind deconvolution, blur kernel estimation, internal patch recurrence, fractal property, statistics of natural images.

1 Introduction

Photos often come out blurry due to camera shake, defocus or low-grade optics. Undoing this undesired effect has attracted significant research efforts over the last decade. In cases in which the blur is uniform (same across the entire image), the blurry image y is often modeled as having been obtained from the desired sharp image x as

$$y = k * x + n, \tag{1}$$

where $*$ denotes convolution, k is some blur kernel and n is noise.

Since both the blur k and the sharp image x are unknown, and since many different pairs of x and k may result in the same blurry image y, blind deblurring heavily relies on the availability of prior knowledge on x. Most existing algorithms rely, either explicitly or implicitly, on the fact that images contain enough step edges. This assumption is formulated in various ways. Some studies assume simple parametric probability models, which promote sparsity of image gradients [5,17,12,13,10]. Others assume a parametric form for the spectrum of the image [8], which decays polynomially with frequency (corresponding to the Fourier transform of step edges). Finally, many approaches employ heuristic methods for detecting and/or enhancing edges in the blurry image. These range from setting a threshold on the image gradients [9] to shock and bilateral filtering [1,19,2].

Gradient priors model interactions between *pairs* of pixels. In recent years, the advantage of using priors over larger neighborhoods (patches) has been recognized. Patch priors model more complex structures and dependencies in larger neighborhoods. Such priors have led to state-of-the-art results in various inverse problems [16] including *non-blind* deblurring [3,22,4] (namely, deblurring with a *known* blur kernel). Recently, Sun *et al.* [18] used a patch prior learned from an external collection of sharp natural images for *blind* deblurring (*unknown* blur kernel). This resulted in a significant improvement in performance over all the previous blind deblurring methods [13,10,1,19,2].

In this paper, we present an approach for blind-deblurring, which is based on the *internal patch recurrence* property within a single natural image. It was empirically shown by [7,20] that almost any small image patch in a natural image (5×5 or 7×7) re-appears "as is" (without shrinking the patch) in smaller scaled-down versions of the image (Fig. 1(a)). This observation was successfully used for various *non-blind* inverse problems (where the degradation process is known), including single-image super-resolution [7,6] and image-denoising [21].

The cross-scale recurrence property was also recently used in [14] for *blind* Super-Resolution (SR). While, superficially, blind-deblurring can be thought of as a special case of blind-SR with a magnification factor $\alpha = 1$, there is a conceptual difference between the two. The goal in blind-SR [14] is to recover an α-times larger image, whose blur is α-times narrower than in the input image (thus imitating an optical zoom-in). Consequently, as opposed to blind-deblurring, the optimal SR blur kernel k_{SR} is *not* the point spread function (PSF) of the camera. Rather, as shown in [14], it is given in the Fourier domain by the following PSF ratio: $K_{\mathrm{SR}}(\omega) = \mathcal{PSF}(\omega)/\mathcal{PSF}(\omega/\alpha)$, where α is the SR magnification factor. Thus, for a magnification factor $\alpha = 1$, the optimal SR blur kernel of [14] reduces to $K_{SR}(\omega) \equiv 1$, namely, a *delta function* in the spatial domain. This is *regardless of the blur in the input image*. Therefore, the blind-SR algorithm of [14] cannot be used for blind deblurring. Put differently, in blind-deblurring we seek to recover the PSF, and not the *ratio* between two PSFs as in blind-SR. Nevertheless, we show that the cross-scale patch recurrence property can still serve as a strong prior for blind-deblurring, but requires a different strategy.

Our approach is conceptually simple. While patches repeat across scales in a *sharp* natural image (Fig. 1(a)), this cross-scale recurrence significantly diminishes in blurry images (Fig. 1(b)). We exploit these *deviations from ideal patch recurrence* as a cue for recovering the underlying (unknown) blur kernel. This is done by seeking a blur kernel k, such that if its effect is *undone* (if y is deconvolved by k), the patch similarity across scales will be maximized. Moreover, while the blur is strong in the original scale, the blur decreases at coarser scales of the image. Thus, sharper image patches "naturally emerge" in coarser scales of the blurry image (*e.g.,* Fig. 1(b)). The patches in coarser image scales can thus serve as a good patch prior (sharper examples) for deblurring the input scale. This allows recovery of the unknown blur kernel. We show that blind deblurring based on the internal patch recurrence prior compares favorably to all previous blind-deblurring approaches. We further show that this is a very stable prior, in the sense that it rarely diverges on any input image (unlike other priors).

The rest of this paper is organized as follows. Section 2 provides an overview of our approach and explains the intuition underlying the optimization process. Section 3 is

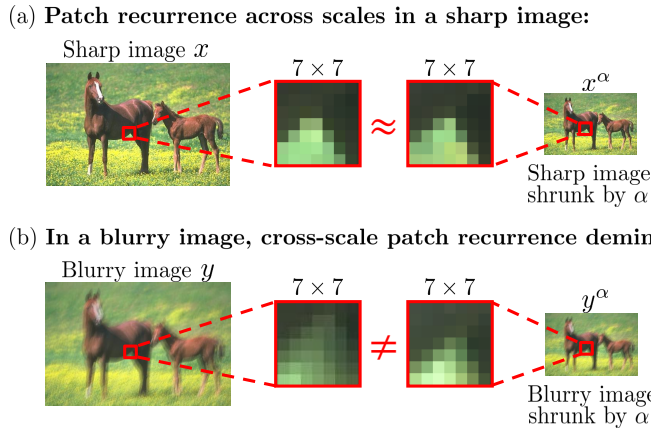

Fig. 1. ***The cross-scale patch recurrence is strong in sharp images and weak in blurry images***. (a) Small patches (*e.g.,* 5×5, 7×7) tend to recur across scales in an "ideal" (sharp) natural image x. Namely, if we down-scale x by a factor of α, then for most patches in x, there exist almost identical patches in the down-scaled image x^α. (b) In contrast, in a blurry image $y = x * k$, this is no longer true. The similarity between patches in y and in its down-scaled version y^α is significantly reduced. Patches in the down-scaled version y^α tend to be α-times sharper than their corresponding patches in y. Thus, down-scaling generates a pool of sharper patches, which can be used as a prior for removing the blur in y.

devoted to an in-depth explanation of our algorithm. Finally, in Section 4, we demonstrate and compare the performance of our algorithm to other state-of-the art methods.

2 Overview of the Approach

We start with a high-level overview of our approach, focusing on the intuition behind the proposed method. We defer the detailed definitions and derivations to Section 3.

While patches repeat across scales in a *sharp* natural image under *ideal* downscaling (Fig. 1(a)), this cross-scale recurrence significantly diminishes in blurry images (Fig. 1(b)). We thus seek a blur kernel k, such that if its effect is *undone* (if y is deconvolved by k), the patch similarity across scales will be maximized. More specifically, we look for an image $\hat{x}$ and a blur kernel $\hat{k}$ such that on the one hand, $\hat{x}$ satisfies the patch recurrence property (namely, strong similarity between patches across scales of $\hat{x}$), and, on the other hand, $\hat{k} * \hat{x}$ is close to the blurry image y. This is done by solving the optimization problem

$$\arg\min_{\hat{x},\hat{k}} \underbrace{\|y - \hat{k} * \hat{x}\|^2}_{\text{data term}} + \lambda_1 \underbrace{\rho(\hat{x}, \hat{x}^\alpha)}_{\text{image prior}} + \lambda_2 \underbrace{\|\hat{k}\|^2}_{\substack{\text{kernel} \\ \text{prior}}}, \tag{2}$$

where $\hat{x}^\alpha$ is an α-times smaller version of $\hat{x}$. The second term $\rho(\hat{x}, \hat{x}^\alpha)$ measures the degree of *dissimilarity* between patches in $\hat{x}$ and their Nearest Neighbor patches (NNs) in $\hat{x}^\alpha$. The third term is a regularizer on the kernel k.

Note that as opposed to blind-SR [14], where the optimal SR kernel is the one which maximizes patch similarity across scales of the *input image*, here we seek a different kernel – the kernel k that (when undone) maximizes patch similarity across scales of the *unknown output image*.

Our optimization problem (2) may be interpreted as a joint MAP estimation of x and k (coined MAP$_{x,k}$ in [12]), which was shown by [12] to lead to wrong (trivial) results. However, as opposed to the simple prior used in [12], under which the MAP$_{x,k}$ strategy indeed favors blurry reconstructions, our prior $\rho(\hat{x},\hat{x}^\alpha)$ avoids such solutions. This is because small patches in a sharp $\hat{x}$, have similar patches (NNs) in its down-scaled version $\hat{x}^\alpha$ (see Fig. 1(a)). Therefore, for a sharp $\hat{x}$, the penalty $\rho(\hat{x},\hat{x}^\alpha)$ is small. On the other hand, patches in a blurry $\hat{x}$, are *less similar* to patches in its down-scaled $\hat{x}^\alpha$ (Fig. 1(b)). Therefore, for a blurry image $\hat{x}$, the penalty $\rho(\hat{x},\hat{x}^\alpha)$ is large.

The objective (2) is not convex (see the definition of $\rho(\hat{x},\hat{x}^\alpha)$ in Sec. 3.2), and has no closed-form solution. We solve it using an alternating iterative minimization procedure comprising of three steps in each iteration, as described in Algorithm 1 below. The iterative process is initialized with the blur kernel $\hat{k}$ being a delta function, and $\hat{x}$ is initially the blurry input image y.

Input: Blurry image y
Output: Blur kernel $\hat{k}$
Initialize $\hat{k}=\delta$ and $\hat{x}=y$;
for $t=1,\ldots,T$ **do**
1. **Image Prior Update:** Down-scale image $\hat{x}$ by a factor of α to obtain $\hat{x}^\alpha$ (Sec. 3.1).
2. **Deblurring:** Minimize (2) w.r.t $\hat{x}$, holding $\hat{k}$ and $\hat{x}^\alpha$ fixed (Sec. 3.2).
3. **Kernel Update:** Minimize (2) w.r.t $\hat{k}$, holding $\hat{x}$ and $\hat{x}^\alpha$ fixed (Sec. 3.3).

end

Algorithm 1: Kernel estimation.

At first sight, our iterative approach may seem similar to other methods, such as [1,19,18], which iterate between an x-step (updating $\hat{x}$ with $\hat{k}$ fixed) and a k-step (updating $\hat{k}$ with $\hat{x}$ fixed). However, close inspection reveals that our x-step is fundamentally different. Rather than using a *fixed generic* prior on natural images, we use an *evolving image-specific* prior based on patches extracted from the down-scaled (sharper) version of the previous image extimate $\hat{x}$. Since our estimate $\hat{x}$ gets sharper from iteration to iteration, *the prior also changes from iteration to iteration*.

Step 1: The purpose of Step 1 of the algorithm is to produce an image $\hat{x}^\alpha$, which serves as *a pool of sharper patches*. Intuitively, if we shrink a blurry image $\hat{x}$ by a factor of α, then the result $\hat{x}^\alpha$ contains α-times less the amount of blur. For example, if we scale-down $\hat{x}$ by a factor of $\alpha=2$, then an edge smeared over 10 pixels in $\hat{x}$ would appear smeared over only 5 pixels in $\hat{x}^\alpha$. However, the image $\hat{x}^\alpha$ is also α-times smaller. In Section 3.1 we prove that, despite the fact that $\hat{x}^\alpha$ is smaller, the pool of small patches (*e.g.,* 5×5) extracted from $\hat{x}^\alpha$ is roughly *the same* as the pool of small patches extracted from the larger image $\hat{x}$, only α-times sharper. This is due to the recurrence of small

patterns at various sizes in the *continuous* scene (see Section 3.1).

Step 2: Step 1 resulted in an image $\hat{x}^\alpha$, which provides a pool of patches that are α-times sharper than those in the image estimate $\hat{x}$. These patches are used in Step 2 as examples for how patches in $\hat{x}$ should look like if we were to sharpen them by a factor of α. To construct a new α-times sharper $\hat{x}$, we minimize (2) with respect to $\hat{x}$ while holding $\hat{k}$ and $\hat{x}^\alpha$ fixed. Disregarding the last term in (2), which does not depend on $\hat{x}$, this amounts to solving

$$\arg\min_{\hat{x}} \|y - \hat{k} * \hat{x}\|^2 + \lambda_1 \rho(\hat{x}, \hat{x}^\alpha). \tag{3}$$

This is in fact the deblurring of y by the current kernel estimate $\hat{k}$, where the prior is represented by the patches in $\hat{x}^\alpha$. In practice, this step tries to assemble a new sharper $\hat{x}$ from the sharper patches in $\hat{x}^\alpha$, as shown in Fig. 2(d). For example, in the *first* iteration (in which $\hat{k} = \delta$), this process results in an image $\hat{x}$, which is close to y, but at the same time its patches are similar to the α-times sharper patches in $\hat{x}^\alpha$. Therefore, intuitively, the image $\hat{x}_1$ recovered in the first iteration contains α-times less the amount of blur than y. At the second iteration, the image $\hat{x}_2$ is α-times sharper than $\hat{x}_1$, and thus α^2-times sharper than y. The image $\hat{x}_\ell$ at the ℓ-th iteration is α^ℓ times sharper than y, and intuitively tends to x for large ℓ.

Step 3: Finally, we update the kernel estimate $\hat{k}$, by computing the blur between the current deblurred estimate $\hat{x}$ and the input image y. Thus, in the ℓ-th iteration, we recover the kernel $\hat{k}_\ell$ such that $y = \hat{k}_\ell * \hat{x}_\ell$. Since for large enough ℓ, $\hat{x}_\ell$ converges to x, the kernel estimate $\hat{k}_\ell$ converges to k. This is the final output of our algorithm.

To speed up the convergence, as well as to avoid getting stuck in a local minimum, the above process is performed coarse-to-fine in a pyramid data structure.

3 Detailed Description of the Algorithm

We now explain in detail each step of Alg. 1.

3.1 Step 1: Generating Sharper Patches by Down-Scaling by a Factor α

The purpose of Step 1 of Alg. 1 is to produce from the current image estimate, $\hat{x}$, a pool of patches that are less blurry. We now formally explain why shrinking a *blurry* image y by a factor of α, generates an α-times smaller image y^α, which contains *approximately the same pool of patches* as in (the larger) image y, only α-times sharper.

Glasner *et al.* [7] showed that most patches in a *sharp* natural image, recur multiple times in its scaled-down version[1]. As further noted in [14], the source of this patch recurrence is the repetitions of small patterns at various sizes in the *continuous*

[1] For example, according to [7], approximately 90% of the 5×5 patches in a *sharp* natural image, recur "as is" 10 or more times in the image scaled-down to $3/4$ of the size ($\alpha = 4/3$).

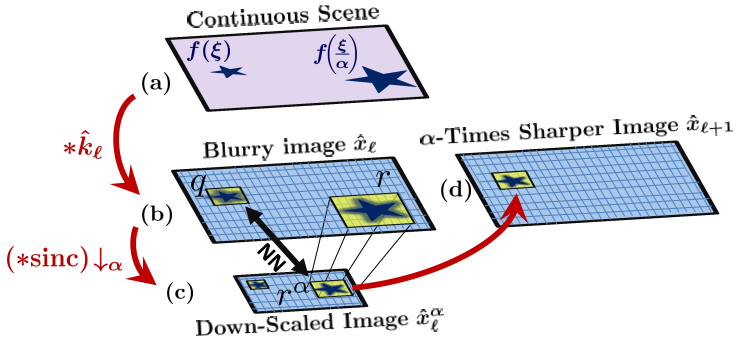

Fig. 2. ***Enforcing the cross-scale patch prior in each iteration.*** (a) A small pattern recurs in the continuous scene at multiple sizes (*blur stars*). (b) At the ℓ-th iteration, the image estimate $\hat{x}_\ell$ corresponds to the convolution of the scene with the kernel estimate $\hat{k}_\ell$. Thus the two patterns in the scene appear as two blurry patches q and r in the image $\hat{x}_\ell$. (c) In the down-scaled version $\hat{x}_\ell^\alpha$, the child patch of r contains the same structure as the patch q in $\hat{x}_\ell$, only α-times sharper. (d) We construct a sharper image $\hat{x}_{\ell+1}$ such that each of its patches is constrained to be similar to its sharper version in $\hat{x}_\ell^\alpha$ (*e.g.,* the new version of q in $\hat{x}_{\ell+1}$ should be similar to the sharper patch r^α in $\hat{x}_\ell^\alpha$).

scene. Consider a small pattern $f(\xi)$ in the continuous scene which recurs elsewhere as $f(\xi/\alpha)$, *i.e.,* α times larger (represented by blue stars in Fig 2(a)). Ignoring sampling issues for the moment, these two patterns are convolved with the blur of the camera $k(\xi)$, and appear in the observed image as the patches q and r (Fig. 2(b)):

$$q(\xi) = k(\xi) * f(\xi), \quad r(\xi) = k(\xi) * f\left(\frac{\xi}{\alpha}\right). \tag{4}$$

Now, if we shrink the blurry image by a factor of α, then the patch r becomes

$$r^\alpha(\xi) = r(\alpha\xi) = \alpha \cdot k(\alpha\xi) * f(\xi). \tag{5}$$

In other words, $r^\alpha(\xi)$ corresponds to the same continuous structure, $f(\xi)$, but convolved with the α-times narrower kernel $\alpha \cdot k(\alpha\xi)$, rather than with $k(\xi)$. This implies that the patch r^α in the smaller image is exactly an α-times sharper version of the patch q in the original blurry image, as visualized in Fig 2(c).

The above shows that shrinking an image by a factor of α produces a pool of patches *of the same size* that are α-times sharper. In Step 2 of the algorithm we use this pool of sharper patches as a nonparametric prior for the purpose of sharpening the blurry image by a factor of α (see Sec. 3.2). Thus, at the first iteration of the algorithm, we recover an image of the scene blurred with the narrower kernel $\alpha \cdot k(\alpha\xi)$. In the second iteration, we further reduce the blur to $\alpha^2 \cdot k(\alpha^2\xi)$, and so on. As visualized in Fig. 3(a) by the red solid curves, the residual blur in the sequence of recovered images becomes narrower and narrower and eventually converges to $\lim_{\ell\to\infty} \alpha^\ell \cdot k(\alpha^\ell\xi) = \delta(\xi)$.

However, the analysis so far assumed *continuous* signals, whereas in practice we work with discrete images. Had the image $\hat{x}_\ell$ recovered in the ℓ-th iteration corresponded to point-wise samples of $\alpha^\ell \cdot k(\alpha^\ell\xi) * f(\xi)$, we would eventually tend to

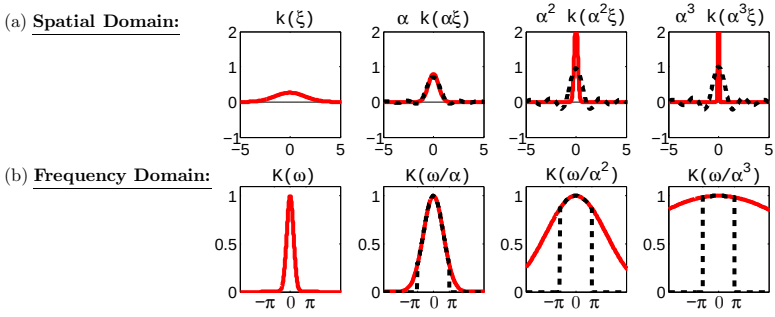

Fig. 3. ***The residual blur in repeated sharpening iterations***. Continuous-domain sharpening (*solid red curve*) tends in the spatial domain (a) to $\lim_{\ell\to\infty} \alpha^\ell \cdot k(\alpha^\ell \xi) = \delta(\xi)$ and in the Frequency domain (b) to $\lim_{\ell\to\infty} K(\omega/\alpha^\ell) = K(0) = 1$. Aliasing-aware sharpening (*dashed black curve*) tends in the spatial domain (a) to $\mathrm{sinc}(\xi)$ and in the frequency domain (b) to $\mathrm{rect}(\omega)$.

point-wise samples of the continuous $f(\xi)$, which would cause aliasing effects. Indeed, as shown in Fig. 3(b) by the red solid curves, the Fourier transform of $\alpha^\ell \cdot k(\alpha^\ell \xi)$, which is $K(\omega/\alpha^\ell)$, converges to[2] $\lim_{\ell\to\infty} K(\omega/\alpha^\ell) = K(0) = 1$ for all ω. Therefore, eventually, all frequencies are retained prior to sampling.

To avoid undesired aliasing effects, we want the recovered $\hat{x}_\ell$ to correspond to samples of the continuous scene $f(\xi)$ convolved with the *band-limited* blur kernel $K(\omega/\alpha^\ell) \cdot \mathrm{rect}(\omega)$, where $\mathrm{rect}(\omega) = 1$ for $|\omega| < \pi$ (the Nyquist frequency[3]) and is zero elsewhere. The logic here is to shrink the blur in the spatial domain (expand it in the frequency domain), but not beyond the Nyquist frequency π (*i.e.,* zero all frequencies above π). Indeed, as illustrated in Fig. 3(b) by the black dashed curves, the function $K(\omega/\alpha^\ell) \cdot \mathrm{rect}(\omega)$ tends to $K(0) \cdot \mathrm{rect}(\omega) = \mathrm{rect}(\omega)$, as ℓ tends to infinity. Therefore, in this case, $\hat{x}_\ell$ converges to samples of the continuous scene convolved with the ideal low-pass filter $\mathrm{sinc}(\xi)$. This is illustrated in Fig. 3(a) by the black dashed curves.

To summarize, the down-scaling operation we perform on the blurry image $\hat{x}_\ell$ should be done so that patches in the resulting $\hat{x}_\ell^\alpha$ are discrete versions of the continuous scene blurred with $K(\omega/\alpha^\ell) \cdot \mathrm{rect}(\omega)$. In the Supplementary Material, as well as in www.wisdom.weizmann.ac.il/~vision/BlindDeblur.html, we provide a proof that if the camera blur $K(\omega)$ is bandlimited to π (so that the blurry image y does not suffer from aliasing), then down-sampling with a sinc kernel leads exactly to the desired result. Therefore, in Step 1 of the algorithm, the down-scaling is performed using a sinc kernel.

3.2 Step 2: Deblurring Using Internal Patch Recurrence

In Step 2 of the algorithm we minimize (2) with respect to $\hat{x}$ while holding $\hat{k}$ and $\hat{x}^\alpha$ fixed, which corresponds to solving Eq. (3). This step is in effect a deblurring of y by

[2] We assume that $\int k(\xi)d\xi = 1$, which implies that in the frequency domain, $K(0) = 1$.

[3] Assuming that the sampling period is 1, the Nyquist frequency is π.

the current kernel estimate, $\hat{k}$. Note that $\hat{k}$ may still be far from the correct k, so that the deblurred $\hat{x}$ we seek to construct is not yet a sharp image. This is in contrast to standard *non-blind* deblurring methods, which rely on priors for *sharp* natural images and seek to recover a *sharp* deconvolved image. Our deconvolved image $\hat{x}$, which is still *partially blurry* is obtained in (3) by using patches from the smaller image $\hat{x}^\alpha$ as a prior. These patches contain "just the right" amount of residual blur, and therefore serve as a good nonparametric prior for the current deblurring step.

Our approach for solving the "partial" deblurring problem (3) is very similar to the non-blind deblurring method of Zoran and Weiss [22]. However, instead of using their natural image prior (which was learned from an external database of sharp patches), our prior is learned from the patches in $\hat{x}^\alpha$. Problem (3) can be written in vector form as

$$\arg\min_{\hat{x}} \|y - \hat{\boldsymbol{K}}\hat{x}\|^2 + \lambda_1 \rho(\hat{x}, \hat{x}^\alpha), \tag{6}$$

where $\hat{\boldsymbol{K}}$ is a matrix that corresponds to convolution with $\hat{k}$. We start by giving a formal definition of the function $\rho(\hat{x}, \hat{x}^\alpha)$. As in [22], we define $\rho(\hat{x}, \hat{x}^\alpha)$ as minus the *expected log likelihood* (EPLL) of patches in $\hat{x}$. Namely, $\rho(\hat{x}, \hat{x}^\alpha) = -\sum_j \log p(\boldsymbol{Q}_j \hat{x})$, where $\boldsymbol{Q}_j$ is a matrix that extracts the j-th patch from $\hat{x}$. However, as opposed to [22], here we learn the probability $p(\boldsymbol{Q}_j \hat{x})$ from the patches in $\hat{x}^\alpha$. Specifically, letting $\boldsymbol{R}_i$ denote the matrix which extracts the i-th patch from $\hat{x}^\alpha$, we approximate $p(\boldsymbol{Q}_j \hat{x})$ using nonparametric density kernel estimation as

$$p(\boldsymbol{Q}_j \hat{x}) = c \sum_i \exp\left\{ -\frac{1}{2h^2} \|\boldsymbol{Q}_j \hat{x} - \boldsymbol{R}_i \hat{x}^\alpha\|^2 \right\}, \tag{7}$$

where h is a bandwidth parameter and c is a constant independent of $\hat{x}$. This results in the prior term

$$\rho(\hat{x}, \hat{x}^\alpha) = -\sum_j \log \left(\sum_i \exp\left\{ -\frac{1}{2h^2} \|\boldsymbol{Q}_j \hat{x} - \boldsymbol{R}_i \hat{x}^\alpha\|^2 \right\} \right). \tag{8}$$

Having defined $\rho(\hat{x}, \hat{x}^\alpha)$, we now proceed to derive an algorithm for minimizing the objective (6). In Appendix B, we show that substituting (8) into (6) and setting the gradient to zero, leads to the requirement that

$$\left(\hat{\boldsymbol{K}}^T \hat{\boldsymbol{K}} + \beta \boldsymbol{I} \right) \hat{x} = \hat{\boldsymbol{K}}^T y + \beta z. \tag{9}$$

Here, $\boldsymbol{I}$ is the identity matrix and $\beta = \lambda_1 M^2 / h^2$, where M is the patch size. z is an image constructed by replacing each patch in $\hat{x}$ by a weighted average of its nearest neighbor (NN) patches in $\hat{x}^\alpha$ (for full expressions see Appendix B). Equation (9) cannot be solved in closed form since z depends nonlinearly on $\hat{x}$. Instead, we alternate a few times between solving for $\hat{x}$ (using (9)) and for z (using (14)–(16) in Appendix B).

What this process boils down to is the following: In the first phase, we replace each patch in the current image $\hat{x}$ by a weighted average of its NNs (using L_2 distance) from

the (sharper) image $\hat{x}^{\alpha}$ (see Fig. 2(d)). This phase actually enforces our prior, which is that patches in the recovered image should be similar to patches in $\hat{x}^{\alpha}$. The resulting image z, however, does not necessarily conform to the data fidelity term, which requires that when the reconstruction is blurred with $\hat{k}$, it should be similar to y. Thus, in the second phase, we plug z back into (9) and update $\hat{x}$. We then repeat the NN search for the patches in the updated $\hat{x}$, generate an updated z, etc. Alternating these phases a few times, leads to an image $\hat{x}$ which satisfies both requirements. Namely, the patches of $\hat{x}$ are similar to those in $\hat{x}^{\alpha}$, *and* its blurry version $\hat{x} * \hat{k}$ resembles y.

3.3 Step 3: Kernel Update

Step 3 in Alg. 1 corresponds to updating the kernel $\hat{k}$, given the current estimate of the image $\hat{x}$. Disregarding the second term in (2), which does not depend on $\hat{k}$, and requiring that the kernel entries be nonnegative, our optimization problem can be written in vector form as

$$\arg\min_{\hat{k}\geq 0} \|y - \hat{\boldsymbol{X}}\hat{k}\|^2 + \lambda_2\|\hat{k}\|^2, \tag{10}$$

where $\hat{\boldsymbol{X}}$ is a matrix that corresponds to convolution with our current image estimate $\hat{x}$.

As explained above, the residual blur in the ℓ-th iteration, is intuitively $K(\omega/\alpha^{\ell})$ in the Fourier domain. Consequently, the kernel recovered in the ℓ-th iteration, should approximately correspond to $K(\omega)/K(\omega/\alpha^{\ell})$. For large ℓ, we have that $K(\omega/\alpha^{\ell}) \approx 1$ and the recovered kernel becomes close to the correct $K(\omega)$. However, for small ℓ, the kernel $K(\omega)/K(\omega/\alpha^{\ell})$ may still be very different from $K(\omega)$ and, in particular, it can have negative values in the spatial domain. Consequently, we impose the nonnegativity constraint in (10) only during the last few iterations of Algorithm 1.

3.4 Implementation Details

To speed up the convergence of the algorithm we work in a coarse-to-fine manner. That is, we apply Alg. 1 on each of the levels of an image pyramid constructed from the blurry input image y. The recovered $\hat{x}$ and $\hat{k}$ at each pyramid level are interpolated to constitute an initial guess for the next pyramid level. The pyramid is constructed with scale-gaps of $\alpha = 4/3$ using down-scaling with a sinc. The number of pyramid levels is chosen such that, at the coarsest level, the blur is smaller than the size of the patches used in the deblurring stage (5×5 patches in our implementation). Additional speed up is obtained by using the fast approximate NN search of [15] in the deblurring step, working with a single NN per patch.

For computational efficiency, we solve the large linear system of equations (9) in the Fourier domain. Specifically, it is easy to verify that the matrix $\hat{\boldsymbol{K}}^T$ appearing in (9) corresponds to convolution with a mirrored version of $\hat{k}$, which is equivalent to multiplication by $K^*(\omega)$ in the frequency domain. It thus follows that solving for $\hat{x}$ while fixing z can be implemented as

$$\hat{X}(\omega) = \frac{\hat{K}^*(\omega)Y(\omega) + \beta Z(\omega)}{|\hat{K}(\omega)|^2 + \beta}. \tag{11}$$

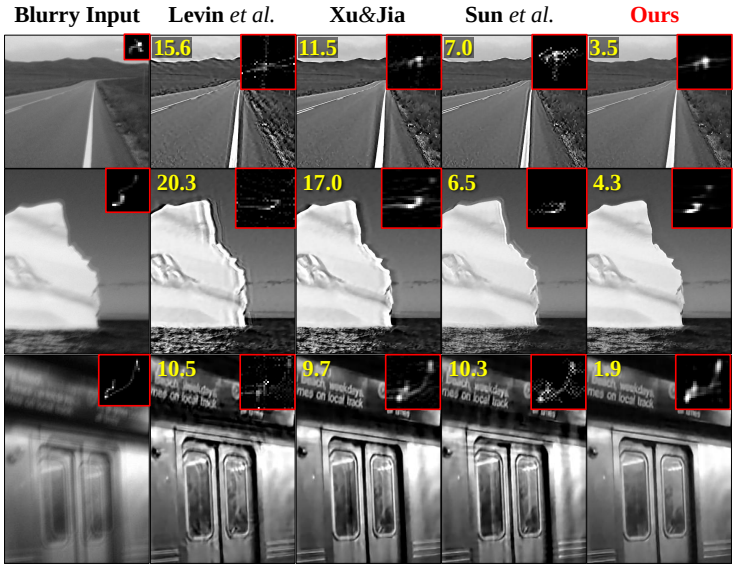

Fig. 4. ***Example deblurring results.*** The left column shows the blurry input image and the ground-truth blur kernel. The other columns show the deblurring results obtained with the kernels estimated by each of the tested methods. The recovered kernel is shown at the top-right corner of each recovered image. The number on each image is its error ratio r. Please zoom-in on screen to see the differences (see www.wisdom.weizmann.ac.il/~vision/BlindDeblur.html for full sized images and more results).

We use FFTs with proper padding to avoid undesired border effects. This formulation is about 50 times faster than *e.g.,* using conjugate gradients to solve this least-squares problem, as done in [22].

In our current implementation we apply 8 iterations of Alg. 1 per pyramid level. We enforce the nonnegativity constraint in (10) starting from the 5th iteration. For gray-values in the range $[0, 255]$, we use $\beta = 0.4$ in the deblurring step (9) and $\lambda_2 = 7.5^2$ in the kernel update step (10).

4 Experiments

We tested our algorithm on the large database introduced by Sun *et al.* [18]. This database comprises 640 large natural images of diverse scenes (typically 1024×768), which were obtained by synthetically blurring 80 high-quality images with the 8 blur kernels from [12] and adding 1% white Gaussian noise. The kernels range in size from 13×13 to 27×27. We present qualitative and quantitative comparisons to the blind deblurring algorithms of [1,19,13,2,10,18]. Specifically, we follow the protocol of [18],

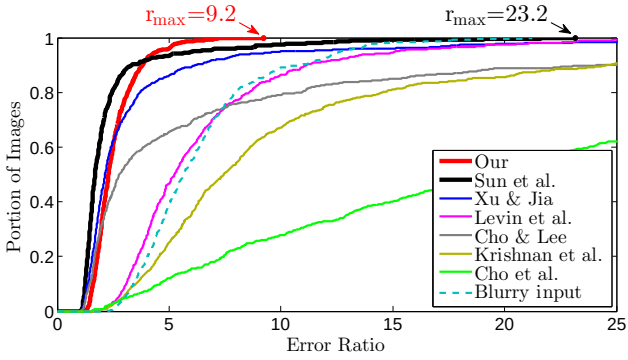

Fig. 5. ***Cumulative distribution of error ratios.***

which used the kernel recovered by each method[4] to perform deblurring with the state-of-the-art non-blind deblurring method of [22]. Since the blur kernel can only be recovered up to a global translation, we align the deblurred image with the ground-truth image in order to compute the error. Following the setting of Sun *et al.* [18], we do not assume that the size of the kernel is known and thus always recover a 51×51 kernel.[5]

We measure the quality of a recovered blur kernel $\hat{k}$, using the *error ratio* measure (proposed in [12] and commonly used by others):

$$r = \frac{\|x - \hat{x}_{\hat{k}}\|^2}{\|x - \hat{x}_k\|^2}, \tag{12}$$

where $\hat{x}_{\hat{k}}$ corresponds to deblurring with the recovered kernel $\hat{k}$, and $\hat{x}_k$ corresponds to deblurring with the ground-truth kernel k. The smaller r is, the better the reconstruction. In principle, if $r = 1$, we achieve "ground-truth performance" (*i.e.,* performance of nonblind deblurring with the ground-truth kernel). However, we empirically observe that the deblurring results are still visually pleasing for error-ratios $r \leq 5$, when using the non-blind deblurring of [22] (see Appendix A for a more detailed explanation).

Fig. 4 shows a few visual examples of the kernel estimates, the deblurring results, and their corresponding error ratios r, obtained by us and by the best competing methods [13,19,18] on several images from the database of [18] (all deblurred using the method of [22]). Complex textures with strong edges, such as the sea in the second row, are better represented by the internal image-specific patch prior, than by any of the other more generic priors (please zoom-in on screen to see fine image details). For example, it seems that the sea regions do not conform to the assumption of sparsity of image gradients of Levin *et al.*[13], and that patches within them do not find good NNs in the external patch prior of Sun *et al.*[18]. The sea region, therefore, distracts most blind deblurring methods, and leads to inaccurate kernel estimates. In contrast, the sea

[4] For [18], we report results with the "natural" patch prior, which performs slightly better than their "synthetic" patch prior. For all other algorithms, we used the results posted by [18] (see their paper for additional details).

[5] When we provide our algorithm the correct kernel size, our results significantly improve.

Table 1. Quantitative comparison of all methods over the entire database (640 blurry images)

	Average performance (mean error ratio)	**Worst-case performance** (highest error-ratio)	**Success rate** (percent of images with error ratio < 5)
Our	2.6	**9.2**	**95.9%**
Sun *et al.*[18]	**2.4**	23.2	93.4%
Xu & Jia [19]	3.6	65.7	86.1%
Levin *et al.*[13]	6.6	41.2	46.7%
Cho & Lee [1]	8.7	112.6	65.6%
Krishnan *et al.*[10]	11.6	133.7	25.1%
Cho *et al.*[2]	28.1	165.6	11.9%

is self-similar within the image, at least across small scale-gaps. Therefore, the internal patch recurrence prior used by our method manages to produce more accurate kernel estimates in such difficult cases.

The graph in Fig. 5 shows the cumulative distribution of error-ratios over the entire database. The statistics indicate that our algorithm and the algorithm of Sun *et al.*[18], which are the only patch-based methods, outperform all other approaches by a large gap. The method of [18] is slightly more accurate than ours at very low error ratios. Nevertheless, empirical inspection shows that the visual differences between results with error-ratios smaller than 3 (when using the deblurring of [22]) are often indistinguishable. As can be seen, our method is *more robust* than all competing approaches, in the sense that it rarely fails to recover the kernel with reasonable accuracy (low $r_{\max}$). In fact, as we show in Fig. 6, even our *worst* result over the entire database (namely, the recovered image with the highest error-ratio, $r_{\max} = 9.2$), is still slightly better than the input blurry image, both visually and in terms of error. In contrast, the worst results of the other methods obtain high errors and are significantly worse than the blurry inputs.

Table 1 further compares the performance of the various blind deblurring methods using three quantitative measures: (i) the *average performance*, (ii) the *worst-case performance*, and (iii) the *success rate*. The *average performance* corresponds to the mean of the error-ratios attained for all images in the database[6]. As can be seen, our average error-ratio is close to that of Sun *et al.*[18] and lower than the rest of the competing methods. Interestingly, only three methods attain an *average* error-ratio smaller than 5 (which can be considered as a threshold for good deblurring; see Appendix A): our method, Sun *et al.* [18], Xu and Jia [19]. This suggests that the visual quality of the remaining methods [1,13,2,10] is unsatisfactory on average.

The *worst-case performance* is the highest error-ratio over the entire database. It measures the robustness of the methods. As can be seen in Table 1, our method is *more robust* than all competing approaches, in the sense that it rarely fails to recover the kernel with reasonable accuracy.

The *success rate* is the percent of images which obtained good-quality deblurring (*i.e.,* an error ratio below 5). As can be seen in Table 1, our method attains an error ratio

[6] Note that the geometric mean used by Sun *et al.*[18] is not sensitive to a small number of severe failures, which is why we prefer the standard (arithmetic) mean.

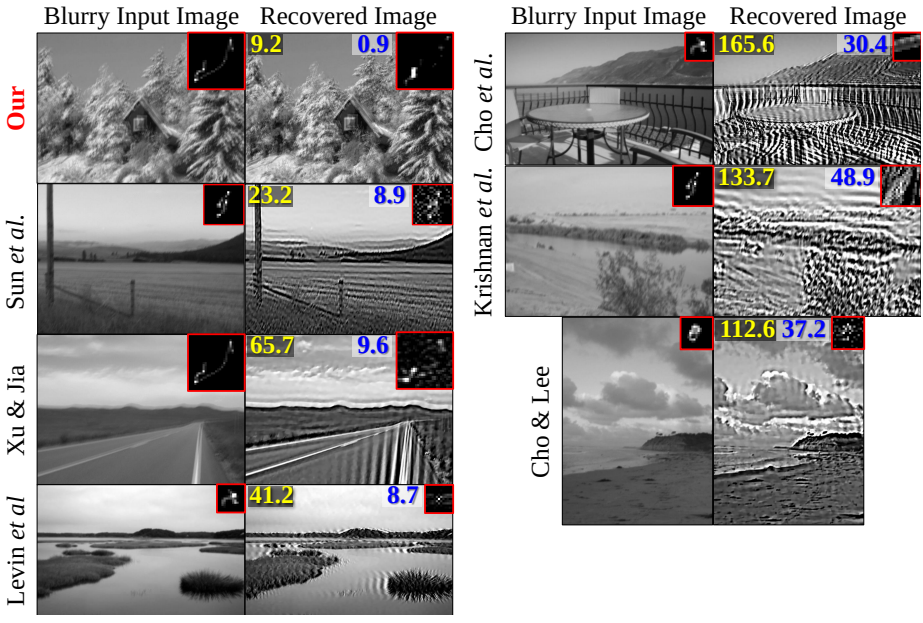

Fig. 6. ***Worst results.*** For each algorithm, the result with the highest error-ratio is shown along with the recovered kernel and the corresponding error-ratio (number in yellow). The number in blue is the ratio between the error of the (output) deblurred image and the error of the input (blurry) image. Values below and above 1 indicate, respectively, improvement or degradation in quality. As can be seen, our worst-case result is still better than the blurry input image while the worst-case results of the competing methods are significantly worse than their input images. See www.wisdom.weizmann.ac.il/~vision/BlindDeblur.html for full sized images.

larger than 5 only 4.1% of the times, which correspond to 26 out of the 640 images in the database. The worst of these 26 'failure cases' can be seen in Fig. 6.

5 Summary

In this paper we presented a blind deblurring method, which uses *internal* patch recurrence as a cue for estimation of the blur kernel. Our key observation is that patch recurrence across scales is strong in sharp images, but weak in blurry images. We seek a blur kernel k, such that if *"undone"* (if the blurry image is deconvolved with k), the patch similarity across scales will be maximized. Extensive empirical evaluations confirm that the internal patch recurrence property is a strong prior for image deblurring, and exhibits higher robustness than other priors. We attribute this to the fact that each image uses *its own image-specific* patch prior.

Acknowledgments. Thanks to Shahar Kovalski, Yuval Bahat and Maria Zontak. Funded in part by the Israel Science Foundation, Israel Ministry of Science, Citigroup Foundation and a Viterbi Fellowship (Technion).

A What is "Good" Blind Deblurring?

The error-ratio measure r in Eq. (12) depends on the type of *non-blind* deblurring method used. We use the state-of-the-art non-blind deblurring method of [22] (which was also the setting used in [18]). We empirically observe that images recovered with an error-ratio smaller than 5 are usually visually pleasing, while error-ratios above 5 are often associated with distracting artifacts. Levin *et al.* [13] reported a threshold of 3 between good and bad visual results. However, their error ratios were computed with the non-blind deblurring of [11]. Using a simulation study (see the Supplementary Material and `www.wisdom.weizmann.ac.il/~vision/BlindDeblur.html` for details), we computed the best linear fit between the two types of error ratios, and found that an error ratio of 3 with [11] indeed corresponds to an error-ratio of approximately 5 with [22]. Thus, based on our observations and those of Levin *et al.* [13], we regard 5 as a threshold for good-quality deblurring when using the non-blind deblurring of [22].

B Derivation of Equation (9)

Assuming the patches are $M \times M$, substituting (8) into (6) and setting the gradient to zero, leads to the requirement that

$$\left(\hat{\boldsymbol{K}}^T \hat{\boldsymbol{K}} + \beta \frac{1}{M^2} \sum_j \boldsymbol{Q}_j^T \boldsymbol{Q}_j \right) \hat{x} = \hat{\boldsymbol{K}}^T y + \beta z, \tag{13}$$

were $\beta = \lambda_1 M^2 / h^2$ and

$$z = \frac{1}{M^2} \sum_j \boldsymbol{Q}_j^T z_j, \tag{14}$$

with

$$z_j = \sum_i w_{i,j} \boldsymbol{R}_i \hat{x}^\alpha \tag{15}$$

and

$$w_{i,j} = \frac{\exp\left\{-\frac{1}{2h^2} \|\boldsymbol{Q}_j \hat{x} - \boldsymbol{R}_i \hat{x}^\alpha\|^2\right\}}{\sum_m \exp\left\{-\frac{1}{2h^2} \|\boldsymbol{Q}_j \hat{x} - \boldsymbol{R}_m \hat{x}^\alpha\|^2\right\}}. \tag{16}$$

It is easy to verify that, up to border effects, multiplying a column-stacked image by $\sum_j \boldsymbol{Q}_j^T \boldsymbol{Q}_j$ is equivalent to multiplying all pixels of the image by M^2, or, in other words, that $\sum_j \boldsymbol{Q}_j^T \boldsymbol{Q}_j = M^2 \boldsymbol{I}$. Substituting this term into (13) leads to (9).

Note that z_j in (15) can be interpreted as an approximation of the patch $\boldsymbol{Q}_j \hat{x}$, which uses the patches $\{\boldsymbol{R}_i \hat{x}^\alpha\}$ from the small image $\hat{x}^\alpha$ as examples. In practice, most of the weights $w_{i,j}$ are very small, so that each patch in $\hat{x}$ is actually approximated by a very small number of its NNs in $\hat{x}^\alpha$ (for efficiency, we often use only a single NN – see Section 3.4). The matrix $\boldsymbol{Q}_j^T$ takes a patch and places it at location j in an image. Therefore, the image z in (14) can be thought of as an approximation of the image $\hat{x}$, where the prior is learned from the patches in $\hat{x}^\alpha$.

References

1. Cho, S., Lee, S.: Fast motion deblurring. In: ACM Transactions on Graphics (TOG). vol. 28, p. 145 (2009)
2. Cho, T.S., Paris, S., Horn, B.K., Freeman, W.T.: Blur kernel estimation using the radon transform. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 241–248 (2011)
3. Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K.: Image restoration by sparse 3d transform-domain collaborative filtering. In: SPIE Electronic Imaging. vol. 6812
4. Danielyan, A., Katkovnik, V., Egiazarian, K.: BM3D frames and variational image deblurring. IEEE Transactions on Image Processing 21(4), 1715–1728 (2012)
5. Fergus, R., Singh, B., Hertzmann, A., Roweis, S.T., Freeman, W.T.: Removing camera shake from a single photograph. In: ACM Transactions on Graphics (TOG). vol. 25, pp. 787–794 (2006)
6. Freedman, G., Fattal, R.: Image and video upscaling from local self-examples. ACM Transactions on Graphics (TOG) 30(2), 12 (2011)
7. Glasner, D., Bagon, S., Irani, M.: Super-resolution from a single image. In: IEEE International Conference on Computer Vision (ICCV) (2009)
8. Goldstein, A., Fattal, R.: Blur-kernel estimation from spectral irregularities. In: Computer Vision–ECCV 2012, pp. 622–635. Springer (2012)
9. Joshi, N., Szeliski, R., Kriegman, D.: PSF estimation using sharp edge prediction. In: IEEE Conference onComputer Vision and Pattern Recognition (CVPR). pp. 1–8 (2008)
10. Krishnan, D., Tay, T., Fergus, R.: Blind deconvolution using a normalized sparsity measure. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 233–240 (2011)
11. Levin, A., Fergus, R., Durand, F., Freeman, W.T.: Image and depth from a conventional camera with a coded aperture. ACM Transactions on Graphics (TOG) 26(3), 70 (2007)
12. Levin, A., Weiss, Y., Durand, F., Freeman, W.T.: Understanding and evaluating blind deconvolution algorithms. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1964–1971 (2009)
13. Levin, A., Weiss, Y., Durand, F., Freeman, W.T.: Efficient marginal likelihood optimization in blind deconvolution. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2657–2664 (2011)
14. Michaeli, T., Irani, M.: Nonparametric blind super-resolution. In: IEEE International Conference on Computer Vision (ICCV) (December 2013)
15. Olonetsky, I., Avidan, S.: Treecann–K-D tree coherence approximate nearest neighbor algorithm. In: Computer Vision–ECCV 2012, pp. 602–615. Springer (2012)
16. Roth, S., Black, M.J.: Fields of experts. International Journal of Computer Vision 82(2), 205–229 (2009)
17. Shan, Q., Jia, J., Agarwala, A.: High-quality motion deblurring from a single image. In: ACM Transactions on Graphics (TOG). vol. 27, p. 73 (2008)
18. Sun, L., Cho, S., Wang, J., Hays, J.: Edge-based blur kernel estimation using patch priors. In: IEEE International Conference on Computational Photography (ICCP). pp. 1–8 (2013)
19. Xu, L., Jia, J.: Two-phase kernel estimation for robust motion deblurring. In: Computer Vision–ECCV, pp. 157–170. Springer (2010)
20. Zontak, M., Irani, M.: Internal statistics of a single natural image. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2011)
21. Zontak, M., Mosseri, I., Irani, M.: Separating signal from noise using patch recurrence across scales. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1195–1202 (2013)

22. Zoran, D., Weiss, Y.: From learning models of natural image patches to whole image restoration. In: IEEE International Conference on Computer Vision (ICCV). pp. 479–486 (2011)