Efficient Nonlocal Means for Denoising of Textural Patterns

Thomas Brox, Oliver Kleinschmidt, and Daniel Cremers

***Abstract*—The present paper contributes two novel techniques in the context of image restoration by nonlocal filtering. Firstly, we introduce an efficient implementation of the nonlocal means filter based on arranging the data in a cluster tree. The structuring of data allows for a fast and accurate preselection of similar patches. In contrast to previous approaches, the preselection is based on the same distance measure as used by the filter itself. It allows for large speedups, especially when the search for similar patches covers the whole image domain, i.e., when the filter is truly nonlocal. However, also in the windowed version of the filter, the cluster tree approach compares favorably to previous techniques in respect of quality versus computational cost. Secondly, we suggest an iterative version of the filter that is derived from a variational principle and is designed to yield non-trivial steady states. It reveals to be particularly useful in order to restore regular, textured patterns.**

***Index Terms*—RST-DNOI**

I. Introduction and related work

In the past two decades, we have seen the development of increasingly sophisticated filtering techniques for removing noise from a given input image $I : (\Omega \subset \mathbb{R}^2) \to \mathbb{R}$. While linear filtering

$$u(x) = G_\rho * I(x) = \int G_\rho(x')I(x - x')\,dx', \qquad (1)$$

with a Gaussian G_ρ of width $\rho > 0$, is known to blur relevant image structures, more sophisticated nonlinear filtering techniques were developed, such as the total variation filter [35], also known as the ROF model, which minimizes the cost functional:

$$E(u) = \int (I - u)^2\,dx + \lambda \int |\nabla u|\,dx. \qquad (2)$$

The ROF model is closely related to nonlinear diffusion filters [32], in particular to the total variation flow [1]

$$\begin{aligned} u(x,0) &= I(x) \\ \partial_t u(x,t) &= \operatorname{div}\left(\frac{\nabla u(x,t)}{|\nabla u(x,t)|}\right). \end{aligned} \qquad (3)$$

For the space-discrete, one-dimensional setting, the solution of (3) was shown to be the minimizer of (2) with $\lambda = t$ [37]. Also wavelet soft shrinkage can be made equivalent to (3).

Despite an enormous success in image enhancement and noise removal applications, approaches like the ROF filter remain spatially local, in the sense that at each location $x \in \Omega$ the update of u is determined only by derivatives of u at that same location x – see equation (3). A class of image filters which adaptively takes into account intensity information from more distant locations are the Yaroslavsky neighborhood filters [41]:

$$u(x) = \frac{\int K(x,y)I(y)\,dy}{\int K(x,y)\,dy}. \qquad (4)$$

They state a smoothed image $u(x)$ as the weighted average of pixels of the original image $I(x)$. K is a nonnegative kernel function which decays with the distance $d^2(x,y) = \gamma|x-y|^2 + |I(x) - I(y)|^2$. A typical choice is the Gaussian kernel $K(x,y) = \frac{1}{(2\pi h^2)^{D/2}} \exp\left(-\frac{d^2(x,y)}{2h^2}\right)$ with kernel width h and data dimensionality D. Application of this filter amounts to assigning large weights to pixels y and their intensities $I(y)$ which are similar in the sense that they are close to $(x, I(x))$ in space and in intensity. The parameter γ allows to adjust the relative importance of spatial and tonal similarity. Such filters are also known as local M-smoothers [13], [40]. A similar, but iterative, filter is the bilateral filter [36], [38]. Relations between such neighborhood filters and nonlinear diffusion filters have been investigated in [5], [28], [11].

Although these semi-local filters[1] substantially increase the number of candidate pixels for averaging compared to diffusion filters, they reveal a similar qualitative denoising behavior as nonlinear diffusion: whereas they preserve large scale structures, small scale structures are regarded as noise and are removed. Particularly in textured images, small-scale structures are not necessarily equivalent to noise. As a consequence, these filters tend to degrade textured areas.

A drastic improvement in this respect has been achieved by a small but decisive extension of the Yaroslavsky filter. Rather than considering only the center pixel in the similarity of two points, we can regard local balls (patches) around these points. This idea is inspired by works on texture synthesis [33], [17], [39] and has been proposed simultaneously with the nonlocal means filter [9] and the UINTA filter [2]. Both filters use a distance that considers not only the similarity of the central pixel, but also the similarity of its neighborhood:

$$d^2(x,y) = \int G_\rho(x')\big(I(x-x') - I(y-x')\big)^2\,dx'. \qquad (5)$$

The Gaussian kernel G_ρ, which is not to be confused with the kernel K, acts as a weighted neighborhood of size ρ. One could as well choose, for instance, a uniformly weighted box function. Since the above similarity measure takes into account complete patches instead of single pixel intensities, the filter is able to remove noise from textured images while preserving

T. Brox is with the Department of Computer Science, University of Dresden, 01187 Dresden, Germany. E-Mail: brox@inf.tu-dresden.de.

O. Kleinschmidt, and D. Cremers are with the Computer Vision Group, University of Bonn, Römerstr. 164, 53117 Bonn, Germany. E-Mail: {kleinsch,dcremers}@cs.uni-bonn.de.

[1] semi-local due to the spatial distance that plays a role in the similarity

the fine structures of the texture. Smoothing of 3D surfaces with this filter has been proven to be feasible as well [42].

Apart from ρ, the filter contains another important parameter, namely the width h of the kernel K. It quantifies how fast the weights decay with increasing dissimilarity of respective patches. The statistical reasoning in [3] allows to determine h automatically. Also the statistical derivation in [24] enables the automatic choice of h once the noise variance is estimated or known a priori.

Regarding the computational complexity of nonlocal filters reveals that a price must be payed for the astonishing results. At each pixel, weights to all other pixels have to be computed. This yields a computational complexity of $O(DN^2)$, where N is the number of pixels in the image, and D is the patch size. For larger images, this complexity is quite a burden. Hence, several approximations have been suggested. The most popular way is to restrict the search to patches in a local neighborhood [9], thus, turning the initially nonlocal filter into a semi-local one. This reduces the computational complexity to $O(DN)$. Similarly, we can apply random sampling, where samples from the vicinity of the reference patch are preferred [2]. Both strategies assume that the most similar patches are in the vicinity of the reference patch. Usually, this locality assumption works fine – remember that diffusion filters already work with a 4-neighborhood. However, it revokes the initial idea and properties of nonlocal filtering.

Speedups without necessarily abandoning the idea of nonlocal filtering have been achieved in [10], [27], [14], [20], [24]. In [10], patch comparison is performed only for a subset of reference patches and these weights are then used for restoring a whole block of pixels. In [27], [14], [20], [24] it was suggested to preselect patches for comparison that have similar means and gradients, or similar means and variances. [14] further proposed a parallelized implementation on 8 Xeon processors.

In Section II, we suggest to reduce the computational complexity to $O(DN \log N)$ by employing cluster trees. Like in [27], [14], [20], [24], the idea is to quickly discard a large amount of dissimilar patches. However, whereas these works base the preselection on some heuristic similarity criteria, the cluster tree structure selects the set of similar patches by means of the distance measure (5) itself, thus being consistent with the filter. The cluster tree arrangement thereby exploits concepts from data retrieval. Tree structures are well known for their eligibility to conduct huge amounts of data. In the context of nearest neighbor search, so-called kd-trees [6] have been very popular. Other indexing or tree structures have been suggested in [12], [29], [26], [30]. Very related are also works on randomized approximate nearest neighbor search, such as locality sensitive hashing [21], [16].

The nonlocal means filter as presented in [9] is a non-iterative filter. However, like the bilateral filter is an iterative version of the Yaroslavsky filter, we can as well iterate nonlocal means. In fact, the entropy minimization framework of the UINTA filter [2] exactly leads to such an iteration of the filter in [9]. Other iterative versions have been proposed in [25], [20], [19], [4]. In Section III, we introduce a variational formulation that leads to an iterative nonlocal filtering designed to yield non-trivial steady states. We presented a preliminary version of this iterative filter at a conference [7]. It turns out that this formulation is particularly practical to restore regular textural patterns, but it also improves the restoration of standard test images.

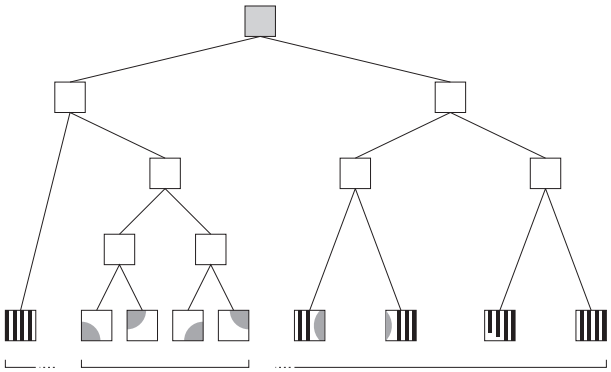

Fig. 1. Schematic illustration of a cluster tree. Leafs contain a relatively small set of similar patches.

II. Fast nonlocal means via cluster trees

A. Accelerating nonlocal means by preselecting patches

In the nonlocal means filter, almost all computation time is spent on computing distances between patches. However, only a relatively small part of all patches is sufficiently similar for their kernel weights $K(x, y)$ playing any role in the averaging. Hence, in order to speed up the filter, the basic idea of the approaches in [27], [14] has been to compute distances only for a reduced set of patches. Preselection of patches is performed by some alternative distances, which can be computed very quickly, such as the difference of the patches' means or variances. Indeed, this strategy leads to a significant speedup, particularly in case of large patches and large search windows. The disadvantage of this approach is that the preselection criterion is hardly related to the distance of patches. For instance, two patches with same means and variances usually comprise vastly different textural structures. Although this does little harm to the filtering outcome - for each patch in the preselected set the exact distance is computed - it reduces the efficiency of the method, as the preselected set still contains a large number of dissimilar patches. In some cases it has been observed that preselection techniques not only improve the speed but even increase the quality of the results [20], [24]. This is because they correspond to additional a-priori assumptions that are satisfied in some data sets. However, ideally such prior assumptions should be part of the distance and not part of the preselection technique; see Section II-C.

B. Cluster trees

As a remedy to this problem, we propose a different way to create sets of potentially similar patches. In particular, we propose a method that preselects patches by means of the same distance measure that is used in the filtering. In order to accomplish this, the basic idea is to arrange all image patches

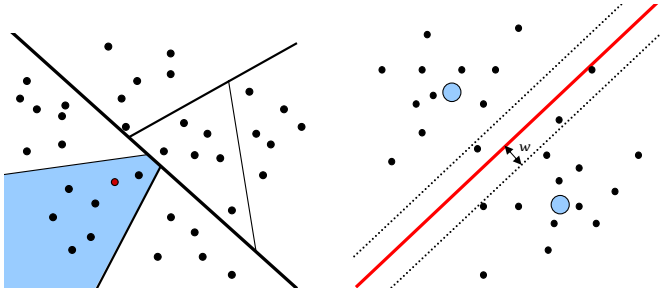

Fig. 2. **Left:** (a) Arrangement of data by a cluster tree allows to quickly access a small set of points whose similarity is measured in patch space. **Right:** (b) In order to increase accuracy, patches that are within w distance of a decision boundary are assigned to both sets.

in a binary tree; see Fig. 1 for an illustration. The root node of this tree contains all patches in the image. Performing a k-means clustering with $k = 2$ splits the patches into two clusters represented by the cluster means. These clusters represent the tree nodes at the first level. Each of these nodes can again be separated via k-means. This way, we can recursively build up the cluster tree. The separation is stopped as soon as the maximum distance of a member to the cluster center is smaller than the kernel width h or the number of patches in a node falls below a certain threshold $N_{min} = 30$. So each leaf node comprises at least N_{min} patches. Finding a local optimum with k-means takes linear time. It has to be applied at each level of the tree, and there are $\log N$ levels. Thus, building the tree runs in $O(DN \log N)$.

Once the tree is built up, we have immediate access to a set of similar patches in constant time. This is achieved by saving for each image patch its corresponding cluster in an index table. One can now apply the typical weighted averaging efficiently on this subset. In contrast to previous preselection techniques, usage of the same distance for clustering and for filtering ensures preselection and filtering by means of one consistent criterion. Fig. 2(a) illustrates this preselection in patch space.

It is clear, though, that the preselected set of patches is only an approximation of the exact set of nearest neighbors. Neighbors close to the reference patch could be part of a neighboring cluster. Such situations appear especially when the reference patch is close to a decision boundary in the tree. One could compute the exact set of nearest neighbors by considering additional branches in the tree that may contain nearest neighbor candidates. However, this so-called backtracking decreases the efficiency of the approach considerably. In many cases, the approximation is sufficient for nonlocal filtering, as one typically deals with a sufficiently large set of similar patches and it is not necessary to have access to the *optimum* set of nearest neighbors in order to achieve a reasonable averaging of intensities.

Nonetheless, we might be interested in increasing the accuracy of the nearest neighbor sets. This is feasible at the cost of some additional memory. Since the problematic areas are along the decision boundaries, we can assign patches close to such a boundary to both subsets. This is the idea of so-called spill trees [26]. In particular, a patch x is assigned to both subsets, if $d^2(x, c_1) < d^2(x, c_2) + w^2$ or, vice-versa, if $d^2(x, c_2) < d^2(x, c_1) + w^2$, where c_1 and c_2 denote the cluster centers of the two subsets; see Fig. 2(b). In case the overlap area w is as large as the support of the kernel, we could ensure the exact set of relevant nearest neighbors. For the Gaussian kernel with infinite support this is not feasible and having large overlap areas also increases the number of patches considered for averaging, which reduces the speedup of cluster trees. However, already choosing small w yields almost the same accuracy as the exact nonlocal means filter and does not demand much more memory than the version with $w = 0$. We will come back to this issue in Section IV.

C. Semi-local filtering and cluster trees

Empirical studies reveal that the nonlocal means filter often performs better when the search for similar patches is restricted to a local subdomain. This corresponds to the prior knowledge that best fitting patches are spatially close. This assumption is true for most images, though we will also show counterexamples later in Section IV. For the cluster tree improvement to be practical on a large variety of images, it must also provide the option to restrict the search space locally. This can be achieved by two strategies. The algorithmic approach is by subdividing the image into overlapping blocks. We used a rather large overlap of 50% in our experiments. A filtering result using cluster trees is computed independently for each block. The final result is then obtained as the weighted average of the block results with the weights $\alpha_i(x) = \exp(-2|x - c_i|^2/(s^2))$, where c_i denotes the center of block i, and s is the width of a block.

A more profound approach with the same effect is to shift the locality assumption to the distance measure by having a distance that consists of the patch similarity and the local proximity: $\tilde{d}^2(x, y) = d^2(x, y) + \gamma|x - y|^2$ with $d^2(x, y)$ as defined in (5) and γ steering the importance of the locality constraint, i.e., the size of the neighborhood. This bilateral filter version is more sound and in combination with the cluster tree it is usually even faster than the algorithmic way to enforce the locality constraint. In a similar manner we can integrate other prior assumptions by choosing corresponding distances. As the cluster tree approach works with general distances, it can accelerate all these approaches.

III. Iterative nonlocal means

A. Previous iterative approaches

The UINTA filter in [2], which was presented at the same time as the nonlocal means filter in [9], can be understood as the first iterative version of nonlocal means. The motivation for the iterative structure of the filter stems from an entropy minimization framework though. Let $u \in \mathbb{R}$ be the gray value of the center pixel of a patch and $v \in \mathbb{R}^{D-1}$ the gray values of the pixels in its neighborhood. Generating the most likely $\mathbf{u} := (u, v)^\top \in \mathbb{R}^D$ can then be formulated as maximizing the conditional probability density $p(u|v)$, or alternatively, the joint probability density $p(\mathbf{u})$. The concept in [2] is to estimate

this density from image samples $\mathbf{u}_i \equiv \mathbf{u}(y)$ via the kernel density estimator [31]

$$p(\mathbf{u}(x)) = \frac{1}{N}\sum_{i=1}^{N} K(\mathbf{u}(x), \mathbf{u}_i), \tag{6}$$

where $K(\mathbf{u}(x), \mathbf{u}(y))$ is the kernel function that also appears in the nonlocal means filter (usually written briefly as $K(x, y)$). Maximization of $p(\mathbf{u})$ with regard to u yields a variant of the mean shift algorithm [18], where the mean shift vector $\frac{\partial p(\mathbf{u})}{\partial \mathbf{u}}$ is projected to the u-axis. Like in [18], the UINTA filter ascents on the densities estimated from the updated samples $\mathbf{u}(y)$. In this framework, the nonlocal means filter from [9] is a single gradient ascent step in u with step size 1, where u is initialized with the intensities I of the noisy input image.

Another iterative nonlocal means algorithm has been suggested in [20]. Rather than from statistics, its motivation is based on the calculus of variation and diffusion processes. It is proposed to minimize the functional

$$E(u(x)) = \frac{1}{4}\int (u(x) - u(y))^2 K_I(x, y)\, dxdy. \tag{7}$$

The notation $K_I(x, y)$ reflects that weights are computed here only once on the basis of the input image I. This is in contrast to the iterative scheme of [2] described above. The Euler-Lagrange equation leads to the following iterative scheme:

$$\begin{aligned} u^0(x) &= I(x) \\ u^{k+1}(x) &= u^k(x) + \tau \int (u^k(y) - u^k(x)) K_I(x, y)\, dy. \end{aligned} \tag{8}$$

In order to ensure properties such as preservation of the average image intensity and convergence to a constant image, the weights are adapted in a special way [20]: (i) only the $m = 5$ largest weights for each point x as well as the four spatial neighbors are kept, all other weights are set to 0; (ii) in case that $K(x, y) = 0$ and $K(y, x) \neq 0$, $K(x, y)$ is set to $K(y, x)$. (i) ensures irreducibility and therefore convergence to a trivial steady state. (ii) ensures that the scheme is conservative, i.e., the mean intensity of the image is preserved. Extensions of [20] have been presented in [19] and [4]. The first replaces the L_2 norm by other norms, such as L_1, resulting in a nonlinear process. The second approach proposes extensions such as spatial bandwidth selection and how to deal optimally with color images. From the conceptual point of view, however, these approaches belong to one class, where the weights $K_I(x, y)$ depend on the input image and are not touched in the iterative process.

B. A new variational principle for nonlocal means

We propose here a variational formulation that is different from the methods above. It is based on a trivial variational principle for the nonlocal means filter, which can be written as:

$$E(u) = \int \left(u(x) - \frac{\int K_I(x, y) I(y)\, dy}{\int K_I(x, y)\, dy} \right)^2 dx. \tag{9}$$

We can derive an iterated form of the nonlocal means filter by extending this functional in a way that replaces K_I by K_u:

$$E(u) = \int \left(u(x) - \frac{\int K_u(x, y) I(y)\, dy}{\int K_u(x, y)\, dy} \right)^2 dx. \tag{10}$$

Thus, rather than imposing similarity of $u(x)$ to $I(y)$ for locations y where the input image $I(y)$ is similar to $I(x)$, we impose similarity to $I(y)$ for locations y where $u(x)$ is similar to the filtered image $u(y)$. This induces an additional feedback and further decouples the resulting image u from the input image I. The idea is that the similarity of patches can be judged more accurately from the already denoised signal than from the noisy input image. On the other hand, in contrast to the UINTA filter, a minimal coupling to the input image I is preserved, since averaging is over the original intensities $I(y)$ rather than the estimated intensities $u(y)$. Another strategy to keep a link to the original image is by an additional data fidelity term, as used, e.g., in [20], [4]. However, this linking is conceptually different as it keeps the result close to the noisy input and thereby reintroduces part of the noise, whereas our approach uses the original image for averaging, where the noise is removed completely when the number of points for averaging gets large enough. Note that the number of parameters is the same: in one case we must choose the iteration number, in the other case we must set the weight of the data fidelity term.

C. Fixed point iteration

Due to the introduced dependence of K on u, the minimizer of (10) is no longer the result of a weighted convolution, as in (9), but the solution of a nonlinear optimization problem. A straightforward way to approximate a solution of (10) is by an iterative scheme with iteration index k, where we start with the initialization $u^0 = I$. For fixed $u = u^k$ we are in a similar situation as with the conventional nonlocal means filter. In particular, we can compute the similarity measure $K_{u^k}(x, y)$ for the current image u^k and, as a consequence, we obtain an update on u

$$u^{k+1}(x) = \frac{\int K_{u^k}(x, y) I(y)\, dy}{\int K_{u^k}(x, y)\, dy} \tag{11}$$

This fixed point iteration resembles very much the iteration equation in [23], [22]. Although in these papers the iteration arises from a different motivation and is coupled with ideas to adapt the local window size, the interesting relation is that their method also averages over values from the input image, but computes distances in the denoised image.

D. Euler-Lagrange equation and gradient descent

An alternative way to find a solution of (10) is by computing its Euler-Lagrange equation, which states a necessary condition for a (local) minimum. We are interested in the gradient

$$\frac{\partial E(u)}{\partial u} = \left. \frac{\partial E(u + \epsilon h)}{\partial \epsilon} \right|_{\epsilon \to 0}. \tag{12}$$

After evaluation and substitution of integration variables, we end up with the following Euler-Lagrange equation:

$$\frac{\partial E(u)}{\partial u} = 0 = \left(u(x) - \frac{\int K_u(x,y)I(y)\,dy}{\int K_u(x,y)\,dy}\right)$$
$$+2\iint \left(u(z) - \left(\frac{\int K_u(z,y')I(y')\,dy'}{\int K_u(z,y')\,dy'}\right)\right)$$
$$\frac{I(y)K_u'(z,y)}{\int K_u(z,y')dy'} G_\rho(y-x)(u(z-y-x)-u(x))dydz$$
$$+2\iiint \left(u(z) - \left(\frac{\int K_u(z,y')I(y')\,dy}{\int K_u(z,y')\,dy'}\right)\right)$$
$$\frac{I(y)K_u(z,y)K_u'(z,y'')}{\left(\int K_u(z,y')dy'\right)^2} G_\rho(z-x)(u(x)-u(y''-z-x))dy''dydz,$$

where $K_u'(x,y)$ denotes the derivative of the kernel function K_u. After initializing $u^0 = I$, the gradient descent equation

$$u^{k+1} = u^k - \tau \frac{\partial E(u)}{\partial u} \tag{13}$$

converges to the next local minimum for some sufficiently small step size τ and $t \to \infty$. Obviously, gradient descent with the gradient being reduced to the first term in (III-D) leads for $\tau = 1$ to the same iterative scheme as in Section III-C. The additional two terms take the variation of K_u into account and ensure convergence. However, these terms induce a very large computational load in each iteration. In particular, the time complexity of the third term in each iteration is $O(MN^4)$, where N is the number of pixels in the image and M the number of pixels in the compared patch. For comparison, the first term only has a time complexity of $O(MN^2)$ and a nonlinear diffusion filter like TV flow has a time complexity of $O(N)$ in each iteration. Hence, in our experiments, we took only the first term into account, which comes down to the fixed point scheme with a flexible time step size. Empirically, this iterative scheme converged for sufficiently small time steps (we chose $\tau = 0.2$). Empirical investigation in other fields such as image segmentation [8] or optical flow estimation show that fixed point iteration and gradient descent can converge to different local optima, but gradient descent does not necessarily find the better optimum. We expect these findings to carry over to the case here, where experimental comparison is not possible due to the large computational complexity.

IV. Experiments

In the experimental evaluation, we quantified the speedup of the cluster tree improvement relative to its loss in quality. Moreover, we compared the proposed iterative technique to other iterative versions of nonlocal means. Finally, we also investigated the use of cluster trees in the iterative scheme. As a quantitative measure for quality we stick to the peak signal-to-noise ratio (PSNR) defined as

$$\mathrm{PSNR} = 10\log_{10}\left(\frac{255^2}{\frac{1}{|\Omega|}\sum_{i\in\Omega}(u_i - r_i)^2}\right), \tag{14}$$

where u denotes the restored image and r the noise free reference image. In all experiments we used the same Gaussian weighted 9×9 patches ($\rho = 2$). Image intensities were in a range between 0 and 255. All computations times are on a Pentium IV 3.4GHz. Figure 3 shows some test images we used. Since they are standard test images from [34] that are widely available and already contain fixed Gaussian noise with standard deviation 20, future comparison to other methods we did not implement is still feasible.

	PSNR	Time
Standard nonlocal means	30.31	41940ms
Random sampling, 100 samples	29.55	25410ms
Random sampling, 200 samples	30.16	72480ms
Spatial sub-sampling	29.51	5480ms
Preselection by mean and variance	29.80	17640ms
Cluster trees, no overlap	29.83	14440ms
Cluster trees, overlap of 5	30.08	19580ms
Cluster trees, overlap of 10	30.26	31330ms

TABLE I
Comparison of several fast nonlocal means implementations (non-iterative) using the Barbara test image and a 21×21 search window. Regarding both quality and computation time, the cluster tree implementation compares well to existing techniques; see also Fig. 4.

A. Comparison of fast, semi-local and nonlocal implementations

First we regarded the fast implementation of nonlocal means via cluster trees. Table I shows a comparison of efficient techniques that have been suggested in the literature. For the comparison we used the 512×512 Barbara test image from Fig.3(a). Given a fixed local search window size of 21×21 pixels, the fastest technique is spatial sub-sampling, i.e., using the distance computed for a point x also for its 8 neighbors [10]. However, the speedup of this method comes at a cost. Fig. 4 shows a zoom into the Barbara image. Clearly, the spatial sub-sampling blurs the image. This is also reflected by the lowest PSNR value. The cluster tree implementation is the second fastest method and it does not cause blurring effects or any other systematic artifacts. It outperforms the preselection of patches by mean and variance as well as random sampling. For the preselection we used the thresholds given in [14].

The speedup that can be achieved with the cluster tree implementation becomes much more significant when the search window size increases. This is shown in Table II. Except for the preselection methods (by mean and variance or via cluster trees), all methods scale linearly with the window size, which can become computationally demanding. Note that using the distance for preselecting patches is much more efficient than the approximation by mean and variance, which selects too many patches that are not similar. Also note that the cluster tree implementation is much more efficient in case of a true nonlocal filtering without any local search window. Then, filtering can be implemented with one large cluster tree instead of many smaller ones and multiple computations at the same pixel due to overlapping local blocks are no longer necessary. This explains why the implementation of semi-locality by including proximity of points in the distance, see Section II-C, is usually more efficient than its algorithmic implementation via local windows.

Fig. 3. Some standard test images including Gaussian noise with standard deviation 20 used in the experiments. **From left to right:** (a) Barbara (512×512). (b) House (256×256). (c) Lena (512×512). (d) Peppers (256×256). (e) Boats (512×512).

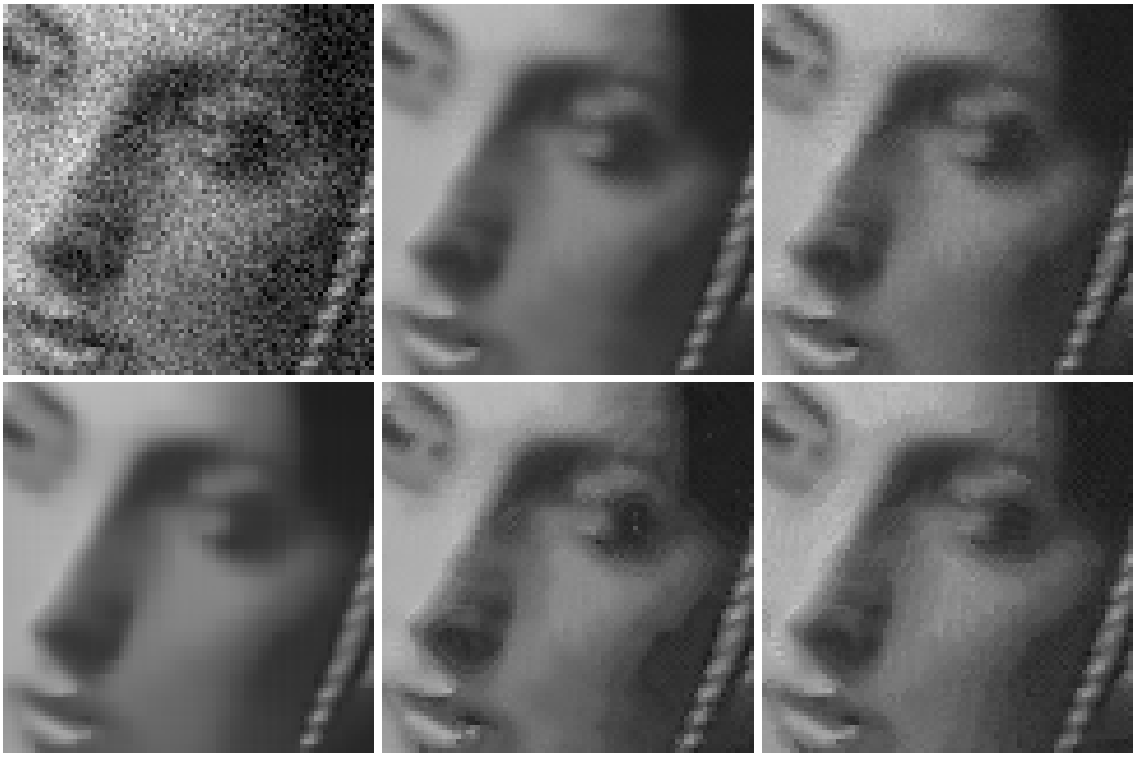

Fig. 4. Zoom into the Barbara image. **Top row, from left to right:** (a) Noisy input image. (b) Standard nonlocal means. (c) Random sampling with 100 samples. **Bottom row:** (d) Spatial sub-sampling. Blurring and block artifacts are visible. (e) Preselection by mean and variance. (f) Cluster trees with no overlap. All methods were run with a 21×21 search window. Apart from spatial sub-sampling, the cluster tree implementation is fastest and its quality is at least as good as that of other acceleration techniques; see also Table I.

	17×17	33×33	65×65	129×129	257×257	no window
Standard nonlocal means	27s	106s	410s	1539s	5378s	16107s
Random sampling	13s	50s	209s	902s	4126s	15362s
Spatial sub-sampling	**4s**	15s	65s	262s	957s	3241s
Preselection by mean and variance	13s	34s	88s	221s	591s	1529s
Cluster trees, $w = 0$	12s	**14s**	**14s**	**14s**	**14s**	**14s**
Cluster trees, $w = 5$	15s	31s	58s	101s	101s	101s
Cluster trees, $w = 10$	22s	61s	149s	303s	687s	797s

TABLE II

COMPUTATION TIMES OF SEVERAL FAST NONLOCAL MEANS IMPLEMENTATIONS DEPENDING ON THE SEARCH WINDOW SIZE. THE METHODS WERE RUN WITH THE 512×512 BARBARA TEST IMAGE. FOR GROWING WINDOW SIZES, THE CLUSTER TREE IMPLEMENTATION IS INCREASINGLY FASTER THAN EXISTING TECHNIQUES.

B. Semi-local versus fast nonlocal filtering of textured images

Table III demonstrates that the common habit in the literature to use only very small search windows (partially only 11×11 pixels [20]) is not always advantageous. Especially in case of strongly textured images, larger windows are beneficial, as the comparison of the nonlocal means filter with a 21×21 and a 41×41, respectively, shows. In such cases, cluster trees yield the largest speedup. The local search window can even be dropped completely, which further increases accuracy and still the method is faster than all other implementations.

	Barbara	Time	Iterations	House	Lena	Peppers	Boats
ROF model	27.05	7.8s	10	31.47	31.40	29.82	29.40
Nonlocal means filter	30.31	52.2s	1	32.49	31.78	29.62	29.34
UINTA filter	30.14	385s	9	32.59	31.79	29.75	29.54
Gilboa-Osher	30.20	48.9s	9	32.55	31.95	30.28	29.89
Gilboa-Osher L_1	29.43	57.3s	80	32.17	31.39	30.04	29.53
Proposed iterative filter	30.33	419s	10	32.74	32.08	30.04	29.69
[22]	30.37	n.a.	n.a.	32.90	32.54	30.59	30.12
[15]	31.78	10s	n.a.	33.77	33.05	31.29	30.88

TABLE IV
PSNRS OF SEVERAL FILTERS ON STANDARD NOISY TEST IMAGES INCLUDING WGN WITH STANDARD DEVIATION 20. ALL PATCH BASED METHODS WERE RUN WITH A 21×21 SEARCH WINDOW. THE PROPOSED ITERATIVE FILTER COMPARES WELL TO EXISTING TECHNIQUES. ONLY TWO VERY SOPHISTICATED RECENT FILTERING TECHNIQUES [22], [15] CAN OUTPERFORM OUR COMPARATIVELY SIMPLE METHOD. CLUSTER TREES HAVE NOT BEEN USED FOR RESULTS IN THIS TABLE.

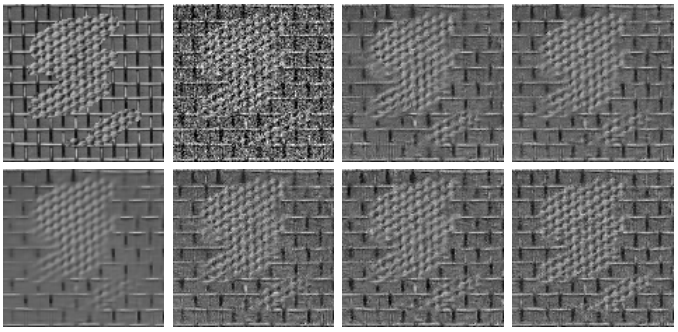

Fig. 5. Comparison of acceleration techniques for a heavily textured image. For corresponding PSNR values and computation times see Table III. **Top row, from left to right:** (a) Reference image with 115×113 pixels. (b) Gaussian noise with standard deviation 40 added. (c) Nonlocal means filter with a 21×21 and (d) a 41×41 search window. The larger window yields a favorable result. **Bottom row:** (e) Nonlocal means by spatial subsampling and a 41×41 window. Blurring effects by the subsampling are clearly visible. (f) Sampled nonlocal means with 312 samples from a 41×41 window. (g) Preselection of patches via mean and variance in a 41×41 window. (h) Nonlocal means with cluster trees and an overlap of 10. Patches from the whole image domain are considered. Cluster tree based denoising compares well to the other methods.

	window	PSNR	Time
Standard nonlocal means	21×21	20.56	1570ms
Standard nonlocal means	41×41	21.11	5440ms
Random sampling, 312 samples	41×41	20.52	4080ms
Spatial sub-sampling	41×41	19.95	720ms
Preselection by mean and variance	41×41	20.68	1780ms
Cluster trees, no overlap	none	20.32	460ms
Cluster trees, overlap of 5	none	20.48	540ms
Cluster trees, overlap of 10	none	21.13	970ms

TABLE III
QUANTITATIVE COMPARISON OF FAST NONLOCAL MEANS IMPLEMENTATIONS CORRESPONDING TO FIG. 5. TRUE NONLOCAL FILTERING WITH THE CLUSTER TREE IMPLEMENTATION IS FASTER THAN SEMI-LOCAL IMPLEMENTATIONS AND THE ACCURACY IS BETTER.

C. Comparison of iterative nonlocal means filters

Table IV quantitatively compares the iterative nonlocal means filters that can currently be found in the literature. Since the cluster tree approximation leads to a small decrease in the PSNR, we used the standard implementation in this experiment. The step size was set to $\tau = 0.2$ as suggested for the UINTA filter in [3]. The Gilboa-Osher filter was implemented as stated in [20]. We also implemented its nonlinear version based on the L_1 norm proposed in [19]. The kernel

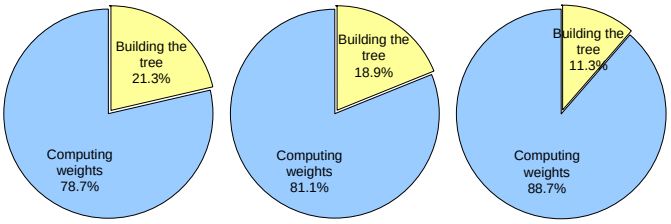

Fig. 6. Relative time taken for building the tree depending on the overlap width w. The image was of size 512×512 pixels. **From left to right:** (a) $w = 0$. (b) $w = 5$. (c) $w = 10$. Although building the tree dominates the asymptotic complexity, in practice, most time is spent for computing the distances of the selected patches.

size h and the number of iterations were manually optimized. The same h was used for all iterations. Actually the UINTA filter optimizes h in each iteration automatically. However, this decreased the PSNR compared to the version with a fixed but manually chosen h. Hence, for having a fair comparison, h was manually chosen for the UINTA filter, as well.

The PSNR values show that differences between the iterative filters are rather small[2]. The filter proposed in the present paper compares favorably, especially when the images contain regular textures. In case of images with rather homogeneous areas, the Gilboa-Osher filter can benefit from emphasizing the direct neighbors of a pixel. Only the two very sophisticated state-of-the-art filtering methods [22] and [15] can achieve higher PSNRs.

D. Cluster tree implementation and iterated filters

The cluster tree implementation can also be used to speed up some of the iterative methods investigated in this paper. In case of the Gilboa-Osher filter, the speedup is not large, since the weights have to be computed only once. Moreover, the Gilboa-Osher filter is a semi-local filter by construction and does not benefit from larger search windows. The UINTA filter as well as the iterative filter proposed in this paper both benefit from the cluster tree in each iteration. Note, however, that in both filters the distances change after each iteration. Consequently, the tree has to be rebuilt in each iteration and cannot be recycled. However, this is not a problem. Although

[2]It might seem surprising that in a few cases, the UINTA filter performs slightly worse than the non-iterative filter. This is possible, since the UINTA filter was run with time steps of 0.2, whereas a single iteration of nonlocal means corresponds to a time step size of 1.

building the tree structure, which takes $O(DN \log N)$ time, dominates the asymptotic complexity of the filter, interestingly, this asymptotic complexity is not relevant for image sizes encountered in practice. Fig. 6 shows the ratio of computation time spent for building the cluster tree and for filtering the image by means of the preselected set of patches, respectively. Building the tree structure is responsible for less than a quarter of the total computation time. The reason is that the preselection sets still contain numerous similar patches for which distances need to be computed. When increasing the overlap width w, the number of patches in a set becomes even larger. This dominance of distance computations over the asymptotic complexity also explains why the cluster tree on the whole image domain can be much faster than the nonlocal means filter with the search restricted to a local window, which has a time complexity of only $O(DN)$. The extra effort for building the tree is easily amortized by the good preselection of similar patches.

An experimental result when applying the cluster tree implementation to the iterative filter proposed in this paper is depicted in Fig. 7. Whereas for the test images in Fig. 3, the impact of iterating the filter was rather small, in this textured image the difference between the iterative and the non-iterative filter is much larger. Especially the regular textures are restored very well by the iterative filter. Thanks to the cluster tree implementation, computation times of the iterative filter are still acceptable.

Fig. 8 shows another texture image, where additionally the noise level is very high. Among the compared filters, only the UINTA filter and the proposed iterative filter can restore the original image. The proposed filter preserves the contrast of textured structures a little better than the UINTA filter. Comparing the results in Fig. 8(f) and Fig. 8(g) with respect to quality and computation time, shows again that the cluster tree implementation on the whole domain outperforms a standard implementation restricted to a local search window.

E. Image evolution

Finally, Fig. 9 depicts the evolution of the image when iterating the proposed filter, the UINTA filter with fixed h, and the UINTA filter with adaptive h, respectively. Empirically, all filters converge to a steady state. This steady state is in all cases a regular texture pattern. However, thanks to the stronger influence of the input image, the proposed filter preserves this pattern much better than the UINTA filter. This is because only the similarity is computed by means of the denoised image u. The averaging is still over the intensities of the input image I.

The motion of the texture boundaries in Fig. 9 looks like curvature motion: corners tend to be rounded and convex areas successively shrink until they finally disappear. This behavior can be explained by using the relationship between the median filter and curvature motion. It is well-known that iterated median filtering approximates curvature motion. The non-local means filter acts similar to the median filter: it tends to replace the central pixel value by a value that stems from the weighted averaging of similar patches. If a patch comprises two different texture patterns, like at a texture boundary, the texture pattern that covers more pixels in the patch will usually dominate the distance and, thus, the central pixel will tend to be replaced by a value that is consistent with the dominant texture pattern. This happens particularly in places where the texture boundary has non-zero curvature. In such places one texture covers more pixels than the other and the filtering step will displace the contour towards the less represented texture. Iterative application of the filter therefore tends to decrease the curvature and shrink convex texture areas. It has to be noted, though, that this rule is only approximative, since the distance between patches not only depends on the ratio of pixels belonging to one or the other pattern but is influenced also by the contrast and frequency of the patterns.

V. Conclusions

In this paper, we presented a new technique to speed up nonlocal means filters. The proposed arrangement of patches in a cluster tree allows for the efficient and sound preselection of similar patches by means of the same distance measure as used by the filter itself. A direct comparison to existing techniques has shown that this consistency in the design of the filter is rewarded by favorable results regarding the quality and speedup. Additionally, we suggested an iterative filter that is built upon a variational principle and is different from the UINTA filter in the sense that it comprises a stronger coupling to the noisy input image. As shown by the experimental evaluation, this stronger coupling preserves the contrast of textural structures much better. The two contributions of this paper can also be combined very well. The emerging filter yields high quality filtering in reasonable time. This is especially the case for textural patterns, where large search windows are needed. We have demonstrated that the filter yields interesting evolutions that resemble curvature motion with level lines defined by texture rather than intensity.

Acknowledgements

We would like to thank Ross Whitaker and Charles Kervrann for interesting discussions, and the three anonymous reviewers for their valuable comments. Moreover, we acknowledge the financial support by the German Research Foundation (DFG).

References

[1] F. Andreu, C. Ballester, V. Caselles, and J. M. Mazón. Minimizing total variation flow. *Differential and Integral Equations*, 14(3):321–360, 2001.

[2] S. Awate and R. Whitaker. Higher-order image statistics for unsupervised, information-theoretic, adaptive image filtering. In *Proc. International Conference on Computer Vision and Pattern Recognition*, pages 44–51, 2005.

[3] S. Awate and R. Whitaker. Unsupervised, information-theoretic, adaptive image filtering for image restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(3):364–376, Mar. 2006.

[4] N. Azzabou, N. Paragios, F. Cao, and F. Guichard. Variable bandwidth image denoising using image-based noise models. In *Proc. International Conference on Computer Vision and Pattern Recognition*, 2007.

[5] D. Barash. A fundamental relationship between bilateral filtering, adaptive smoothing and the nonlinear diffusion equation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(6):844–847, 2002.

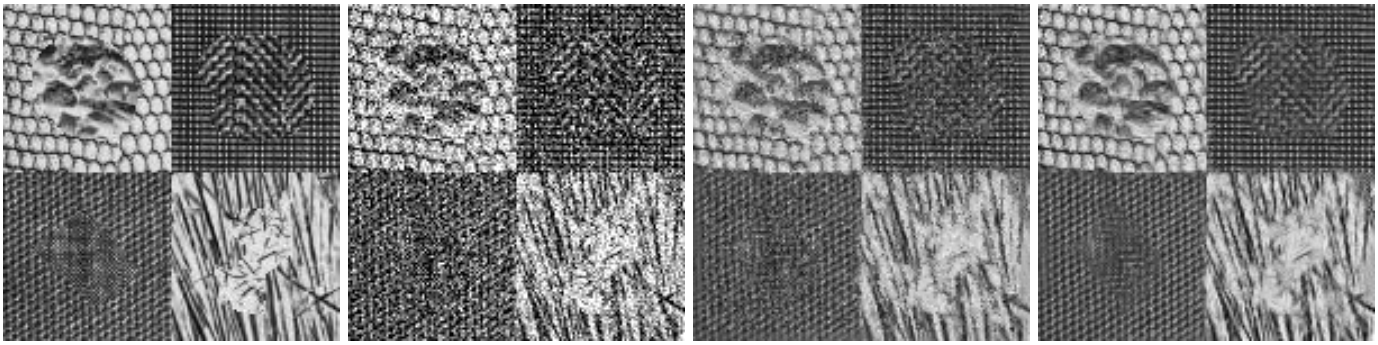

Fig. 7. **From left to right:** (a) Reference image showing 8 different textures (150×150 pixels). (b) Gaussian noise with standard deviation 50 added. (c) Nonlocal means filter implemented via the cluster tree, $\mathrm{PSNR} = 18.11$, computation time: 1210ms. (d) Proposed iterative filter implemented via the cluster tree after 11 iterations, $\mathrm{PSNR} = 20.27$, computation time: 13300ms. The regular textures are reconstructed much more precisely than with the non-iterative filter.

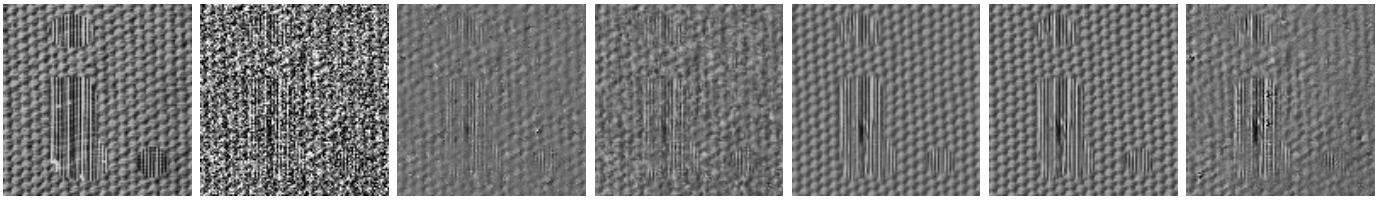

Fig. 8. Comparison of iterative filters on a textured image disturbed by severe noise. Except for the last result in (g), filtering was truly nonlocal (no local search window). **From left to right:** (a) Reference image (119×121 pixels). (b) Gaussian noise with standard deviation 70 added. (c) Non-iterative filter. $\mathrm{PSNR} = 17.49$, computation time: 63s. (d) Gilboa-Osher filter. $\mathrm{PSNR} = 17.94$, 3 iterations, computation time: 64s. (e) UINTA filter implemented via the cluster tree. $\mathrm{PSNR} = 20.05$, 15 iterations, computation time: 10s. (f) Proposed iterative filter implemented via the cluster tree. $\mathrm{PSNR} = 20.23$, 17 iterations, computation time: 11s. (g) Iterative filter with naive search in a 41×41 window. $\mathrm{PSNR} = 18.30$, 8 iterations, computation time: 57s. Only the UINTA filter and the proposed iterative filter yield satisfactory reconstructions.

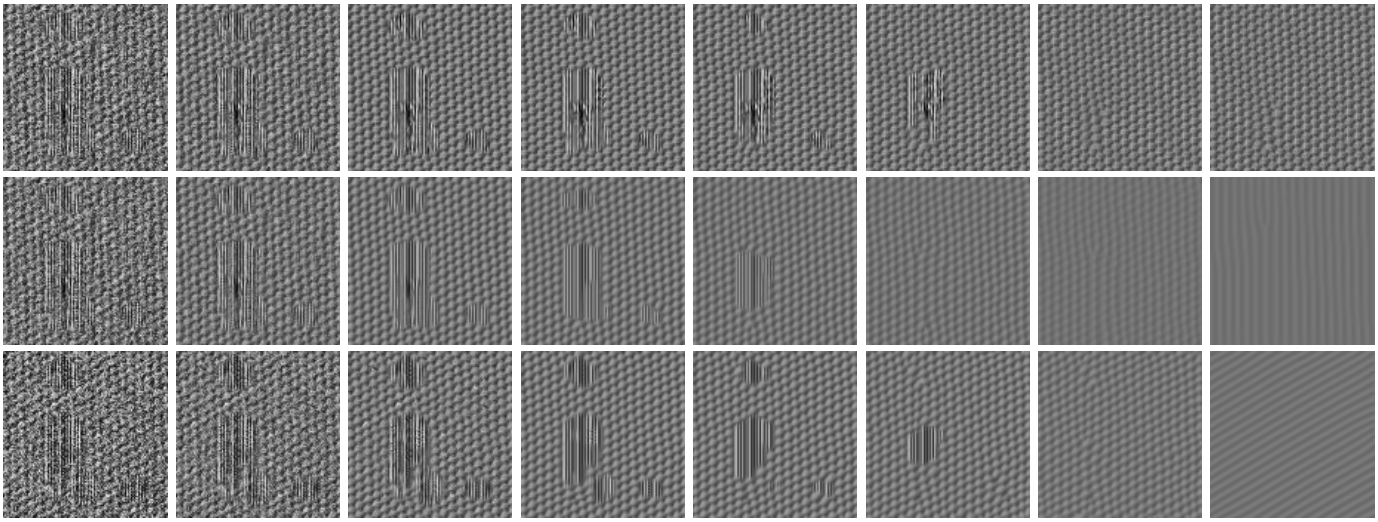

Fig. 9. Convergence behavior for the image from Fig. 8(b). All results were computed with the cluster tree speedup. **From left to right:** Result after 5, 10, 20, 50, 100, 200, 400, and 1000 iterations. **Top row:** Proposed iterative filter. **Center row:** UINTA filter. **Bottom row:** UINTA filter with adaptively chosen h in each iteration. All three iterative schemes converge to a single homogeneous texture pattern. The UINTA filter, however, tends to degrade the texture with the number of iterations, independent of whether h is adapted or not.

[6] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[7] T. Brox and D. Cremers. Iterated nonlocal means for texture restoration. In F. Sgallari, A. Murli, and N. Paragios, editors, *Scale Space and Variational Methods in Computer Vision*, volume 4485 of *LNCS*, pages 13–24. Springer, 2007.

[8] T. Brox and D. Cremers. On local region models and the statistical interpretation of the piecewise smooth Mumford-Shah functional. supplementary online material. Available at http://www-cvpr.iai.uni-bonn.de/pub/pub/brox_ijcv08_sup.pdf, 2007.

[9] A. Buades, B. Coll, and J. M. Morel. A non-local algorithm for image denoising. In *Proc. International Conference on Computer Vision and Pattern Recognition*, pages 60–65, 2005.

[10] A. Buades, B. Coll, and J. M. Morel. A review of image denoising algorithms, with a new one. *SIAM Interdisciplanary Journal*, 4(2):490–530, 2005.

[11] A. Buades, B. Coll, and J. M. Morel. The staircasing effect in neighborhood filters and its solution. *IEEE Transactions on Image Processing*, 15(6):1499–1505, 2006.

[12] A. Buzo, A. H. Gray, R. M. Gray, and J. D. Markel. Speech coding based upon vector quantization. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(5):562–574, 1980.

[13] C. K. Chu, I. Glad, F. Godtliebsen, and J. S. Marron. Edge-preserving smoothers for image processing. *Journal of the American Statistical Association*, 93(442):526–556, 1998.
[14] P. Coupé, P. Yger, and C. Barillot. Fast nonlocal means denosing for 3D MR images. In R. Larsen, M. Nielsen, and J. Sporring, editors, *Proc. International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 4191 of *LNCS*, pages 33–40, 2006.
[15] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3d transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8), Aug. 2007.
[16] T. Darell, P. Indyk, and G. Shakhnarovich, editors. *Nearest Neighbor Methods in Learning and Vision: Theory and Practice*. MIT Press, 2006.
[17] A. Efros and T. Leung. Texture synthesis by non-parametric sampling. In *Proc. International Conference on Computer Vision*, pages 1033–1038, Corfu, Greece, Sept. 1999.
[18] K. Fukunaga and L. Hostetler. The estimation of the gradient of a density function with applications in pattern recognition. *IEEE Transactions on Information Theory*, 21(1):32–40, 1975.
[19] G. Gilboa, J. Darbon, S. Osher, and T. Chan. Nonlocal convex functionals for image regularization. Technical Report CAM-06-57, Department of Mathematics, University of California at Los Angeles, CA, U.S.A., Oct. 2006.
[20] G. Gilboa and S. Osher. Nonlocal linear image regularization and supervised segmentation. Technical Report CAM-06-47, Department of Mathematics, University of California at Los Angeles, CA, U.S.A., Sept. 2006.
[21] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proc. 13th Annual ACM Symposium on Theory of Computing*, pages 604–613, 1998.
[22] C. Kervrann and J. Boulanger. Local adaptivity to variable smoothness for exemplar-based image denoising and representation. *International Journal of Computer Vision*. To appear.
[23] C. Kervrann and J. Boulanger. Optimal spatial adaptation for patch-based image denoising. *IEEE Transactions on Image Processing*, 15(10):2866–2878, 2006.
[24] C. Kervrann, J. Boulanger, and P. Coupé. Bayesian non-local means filter, image redundancy and adaptive dictionaries for noise removal. In F. Sgallari, A. Murli, and N. Paragios, editors, *Scale Space and Variational Methods in Computer Vision*, volume 4485 of *LNCS*, pages 520–532. Springer, 2007.
[25] S. Kindermann, S. Osher, and P. W. Jones. Deblurring and denoising of images by nonlocal functionals. *SIAM Interdisciplinary Journal*, 4(4):1091–1115, 2005.
[26] T. Liu, A. Moore, A. Gray, and K. Yang. An investigation of practical approximate nearest neighbor algorithms. In *Proc. Neural Information Processing Systems*, pages 825–832, 2005.
[27] M. Mahmoudi and G. Sapiro. Fast image and video denoising via nonlocal means of similar neighborhoods. *Signal Processing Letters*, 12(12):839–842, 2005.
[28] P. Mrázek, J. Weickert, and A. Bruhn. On robust estimation and smoothing with spatial and tonal kernels. In R. Klette, R. Kozera, L. Noakes, and J. Weickert, editors, *Geometric Properties from Incomplete Data*, pages 335–352. Springer, Dordrecht, 2006.
[29] S. Nene and S. Nayar. A simple algorithm for nearest neighbor search in high dimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(9):989–1003, 1997.
[30] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *Proc. International Conference on Computer Vision and Pattern Recognition*, pages 2161–2168, 2006.
[31] E. Parzen. On the estimation of a probability density function and the mode. *Annals of Mathematical Statistics*, 33:1065–1076, 1962.
[32] P. Perona and J. Malik. Scale space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:629–639, 1990.
[33] K. Popat and R. Picard. Novel cluster-based probability model for texture synthesis, classification, and compression. In *Proc. SPIE Visual Communications and Image Processing*, 1993.
[34] J. Portilla, V. Strela, M. Wainwright, and E. Simoncelli. Image denoising using scale mixtures of gaussians in the wavelet domain. *IEEE Transactions on Image Processing*, 12(11):1338–1351, 2003.
[35] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.
[36] S. M. Smith and J. M. Brady. SUSAN: A new approach to low-level image processing. *International Journal of Computer Vision*, 23(1):45–78, May 1997.
[37] G. Steidl, J. Weickert, T. Brox, P. Mrázek, and M. Welk. On the equivalence of soft wavelet shrinkage, total variation diffusion, total variation regularization, and SIDEs. *SIAM Journal on Numerical Analysis*, 42(2):686–713, May 2004.
[38] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Proc. Sixth International Conference on Computer Vision*, pages 839–846, Bombay, India, Jan. 1998. Narosa Publishing House.
[39] L.-Y. Wei and M. Levoy. Deterministic texture analysis and synthesis using tree structure vector quantization. In *Proc. Brazilian Symposium on Computer Graphics and Image Processing*, pages 207–214, 1999.
[40] G. Winkler, V. Aurich, K. Hahn, and A. Martin. Noise reduction in images: some recent edge-preserving methods. *Pattern Recognition and Image Analysis*, 9(4):749–766, 1999.
[41] L. P. Yaroslavsky. *Digital Picture Processing - An Introduction*. Springer, 1985.
[42] S. Yoshizawa, A. Belyaev, and H.-P. Seidel. Smoothing by example: mesh denoising by averaging with similarity-based weights. In *Proc. International Conference on Shape Modeling and Applications*, pages 38–44, June 2006.

Thomas Brox received the Ph.D. degree in computer science from the Saarland University, Saarbrücken, Germany in 2005. Subsequently, he spent two years as a postdoctoral researcher at the University of Bonn, Germany. Since October 2007, Dr. Brox is a professor at the University of Dresden in Germany. His research interests include image segmentation, optical flow estimation, texture analysis, density estimation, and pose estimation. For his work on optical flow estimation, he received the Longuet-Higgins Best Paper Award at the ECCV 2004.

Oliver Kleinschmidt is a graduate student in computer science at the University of Bonn, Germany. His research interests are in texture analysis and fast indexing techniques.

Daniel Cremers received Bachelor degrees in Mathematics (1994) and Physics (1994), and a Master's degree in Theoretical Physics (1997) from the University of Heidelberg. In 2002 he obtained a PhD in Computer Science from the University of Mannheim, Germany. Subsequently he spent two years as a postdoctoral researcher at the University of California at Los Angeles and one year as a permanent researcher at Siemens Corporate Research in Princeton, NJ. Since October 2005 he heads the Research Group for Computer Vision at the University of Bonn, Germany. His publications received several awards. Prof. Cremers serves as a proposal reviewer for several funding organizations and has given more than 80 talks and invited speeches over the last seven years.