Stochastic Image Denoising

Francisco Estrada[1]
http://www.cs.utoronto.ca/~strider
David Fleet[1]
fleet@cs.utoronto.ca
Allan Jepson[2]
jepson@cs.utoronto.ca

[1] University of Toronto at Scarborough
1265 Military Road, M1C 1A4
Toronto, On. CANADA

[2] University of Toronto
6 King's College Road, M5S 3G4
Toronto, On. CANADA

Abstract

We present a novel, probabilistic algorithm for image noise removal. We show that suitably constrained random walks over small image neighbourhoods provide a good estimate of the appearance of a pixel, and that a stable estimate can be obtained with a small number of samples. We provide a through evaluation and comparison of the proposed algorithm over a large standardized data set. Results show that our method consistently outperforms competing approaches for image denoising.

1 Introduction

The problem of removing noise from images, and the more general problem of reconstructing a signal that has been corrupted in some way have a long and well travelled history. Considerable research exists both for general approaches, and for specific domains where knowledge about the nature of the signal is used to customize or tune the estimation process. Despite the amount of existing research, there is a gap between the introduction of state-of-the-art denoising methods, and their broad adoption outside of very specific applications.

Several factors contribute to this situation. Current state-of-the-art methods are complex, and either require training image sets that provide meaningful statistics about the domain of application, or assume particular distributions based on empirical observation (see [21] for example). This makes it difficult to implement, extend, or adapt these methods to work with images from specific domains. In addition to the above, we lack a suitable benchmark for comparing existing denoising methods. The result is that there are no solid foundations upon which to make an informed choice of denoising method for a particular problem, including medical imaging, astronomy, photography (particularly in low light conditions), and restoration of archival footage.

With the above considerations in mind, we propose a novel algorithm for image denoising. Our method is conceptually simple, relying on Monte Carlo simulation to sample a subset of all possible random walks that start at a given pixel, and using the probability of travelling between pairs of pixels as a weight to combine them into an estimate of what the noise-free pixels should look like. We provide a thorough comparison between our method and three alternate algorithms on images from the Berkeley Segmentation Database (BSD) [17]. We will show that our algorithm outperforms alternate methods representative

British Machine Vision Conf., Sept 2009

of the state-of-the-art in image denoising. Our algorithm achieves superior denoising results while preserving greater detail. We show applications of our algorithm on real-world images from the domains of medical imaging, high-resolution digital photography, and astronomy, and discuss potential extensions to our framework.

2 Previous Work

There is a large body of literature dedicated to the topic of image denoising. Our review here is brief due to space limitations, and is intended to highlight the broad categories of existing algorithms and to provide appropriate background for our work. For an extensive and thorough review of image denoising literature please see [4].

We group image denoising algorithms into three broad categories. The first is comprised of algorithms related to anisotropic diffusion or anisotropic smoothing [28]. Such algorithms perform smoothing over subsets of pixels that are considered similar to one another, avoiding blurring across strong image boundaries while reducing noise. The original formulation of anisotropic image smoothing is due to Perona and Malik [19]. It has since been expanded and improved upon, and has been shown to be directly related to robust estimation processes [3]. Methods based on minimizing total variation ([22],[6],[5]), as well as the bilateral filter and its derivatives ([24],[2]) are further examples of this class of algorithms.

The second class of denoising methods exploits natural image statistics learnt over a suitable training set. Typically, relevant statistics are represented through distributions of wavelet coefficients (see for example [21], [20], and [7]). The denoising process involves careful shrinkage of the coefficients to reduce the effects of noise. The third class of algorithms uses sampling techniques to combine information from multiple image patches that are similar in appearance. Since the structure in such patches is correlated, but the noise is not, the set of patches can be used to determine the appropriate coefficient shrinkage that reduces noise while preserving structure (see [9], and [12]), or the weights to be used for combining the appearance of different pixels (as in [4], and [16]).

Regardless of the specific method, the goal of the process is to find an image that is as free of visible noise as possible, while preserving fine image detail. Each category of algorithms described above has specific limitations. Anisotropic smoothing methods perform well along strong edges but suffer in the presence of texture. Algorithms that use coefficient statistics can achieve their best performance only if suitable coefficient statistics are available for the domain of application and have difficulty dealing with structured noise. Exemplar based methods work well for repetitive image structure but have trouble preserving fine detail that is not periodic in nature. As a consequence, there is still a need for simple yet effective denoising algorithms that work well in images from multiple domains.

In what follows, we describe a novel denoising method related to robust anisotropic image smoothing. The algorithm is able to significantly reduce the amount of visible image noise while preserving fine detail and sharp edges. It is conceptually simple, and achieves superior denoising performance compared to current state-of-the-art algorithms.

3 Denoising With Random Walks

The relationship between denoising and random walks has been noted previously. Black et al. [3] point out a direct connection between random walks in 3×3 neighbourhoods and robust anisotropic diffusion. Krim and Bao [13] further explore the application of such random walks to 1-D signal denoising. In both of these previous efforts the random walk process is limited to the immediate neighbours of a pixel, and the denoising process iteratively produces a denoised estimate. More recently, Azzabou *et al.* [1] apply random walks to learn brightness distributions over image neighbourhoods. Models for these distributions are then used with a particle filter for denoising. The problem with this approach is its high computational cost. We argue that accurately modelling the pixel neighbourhoods is not necessary to obtain a good denoised estimate.

Our algorithm is based on random walks over arbitrary neighbourhoods surrounding a given pixel. The size and shape of each neighbourhood are determined by the configuration and similarity of nearby pixels. We draw inspiration from the use of random walks to determine pixel similarity in the context of image segmentation [18],[10]. In [10] an image, or in general an arbitrary image patch of size $n*m$ is converted into a graph $G(V,E)$, where V is a set of nodes, one for each pixel, and E is a set of edges linking neighbouring nodes together. The weight or strength of the edge $e(i,j) \in E$ linking pixels i, and j is proportional to the similarity between these pixels. The graph G is represented as a Markov matrix M.

Simulating t steps of a random walk on G involves computing a matrix M^t, and this computation can be performed efficiently using the leading eigenvectors of M. It is thus possible to simulate a random walk started at an arbitrary pixel x_0 for an arbitrary number of steps. Since the probability of transitioning between neighbouring pixels depends on their similarity, the simulation yields a diffusion pattern or *blur kernel* [10] which favours diffusion over pixels that are similar to x_0. The estimated blur kernel gives the weights with which pixels in the patch might be combined to produce an anisotropically smoothed estimate of the starting pixel. While not used for denoising in [10], it was noted that blur kernels should produce good estimates of the true values of pixels in the presence of noise.

The original blur kernel formulation, however, has some drawbacks. It requires the computation of a very large graph and its associated eigenvectors. For an image of $n \times m$ pixels, the Markov matrix M has $(nm)^2$ entries and, though sparse, it quickly becomes unmanageable. Alternatively, we could compute the blur kernels over smaller neighbourhoods centred at each pixel. But this would require a significant computational effort. We argue that, similar to modelling pixel neighbourhoods in [1], computing accurate blur kernels is not necessary to achieve an equivalent anisotropic smoothing effect. We propose replacing the blur kernel formulation by direct random sampling within the neighbourhood of x_0, and we will show that it is possible to obtain stable denoising estimates using a small number of random samples.

3.1 Random Walk Definition

Assuming that pixels within the neighbourhood of x_0 are likely to have been generated by the same random process, we want the weights used to mix these pixels during denoising to depend on their similarity with respect to x_0. At the same time, we require the random walk to follow a smooth path from x_0 to any other pixel in the neighbourhood, so the transition probabilities should also depend on the similarity of neighbouring pixels along any given path.

With this in mind, we define a random walk originating at pixel x_0 as an ordered sequence of pixels $T_{0,k} = \{x_0, x_1, \ldots, x_k\}$ visited along the path from x_0 to x_k. Within this sequence, the probability of a transition between two consecutive pixels x_j and x_{j+1} is defined to be

$$p(x_{j+1}|x_j) = \frac{1}{K} e^{\left(\frac{-d(x_0,x_{j+1})^2}{2\sigma^2}\right)} e^{\left(\frac{-d(x_j,x_{j+1})^2}{2\sigma^2}\right)}, \tag{1}$$

where K is a normalization constant, $d(x_i, x_j)$ is a dissimilarity measure relating two image pixels, and σ is a scaling parameter. The first term in Eq. 1 accounts for the similarity between x_0 and neighbouring pixels, it prevents blurring across soft brightness or colour gradients. The second term imposes a preference for smooth transitions and prevents crossing over strong edges. Note that typical anisotropic smoothing formulations [19], [3], as well as the blur kernel definition from [10] include only the second term, while the bilateral filter [24] employs only the first.

A sequence $T_{0,k+1}$ is generated from $T_{0,k}$ by sampling the neighbourhood of x_k and choosing a neighbour with probability $p(x_{k+1}|x_k)$. Under the first-order Markov assumption, the probability of a sequence starting at x_0 is given by

$$p(T_{1,k}|x_0) = \Pi_{j=1}^{k} p(x_j|x_{j-1}). \tag{2}$$

We have thus far avoided discussing the form of $d(x_i, x_j)$. This is to emphasize that the above formulation can be applied to any domain as long as a suitable dissimilarity function and neighbourhood structure can be defined. For our image denoising problem we use the Euclidean distance between the *RGB* values of the corresponding pixels (assuming values in [0,1]). But we could apply the same process to optical flow data or disparity maps for example. Given the above, we propose a denoising algorithm based on repeatedly sampling sequences $T_{0,k}$ starting at a specific image pixel, and combining the visited pixels along each path to obtain the denoised estimate.

3.2 Stochastic Denoising

The denoised estimate $\vec{I}(x_0^*)$ for a given pixel x_0 is computed from a set of m random walk sequences (trials) $T_{0,k}^i$, $i = 1, \ldots, m$, all beginning at x_0. For each pixel x_j, $j = 1, \ldots, k$ in the sequence $T_{0,k}^i$, we compute a mixing weight, W_j^i based on $p(T_{1,j}^i|x_0)$. Intuitively, since the transition probabilities depend on pixel similarity, if this probability is large then there is a smooth path of similar pixels between x_0 and x_j. Otherwise, the sequence has either crossed a strong boundary, or has wandered into a region consisting of pixels dissimilar to x_0. However, we do not use $p(T_{1,j}^i|x_0)$ directly. There are two reasons for this. First, except for uniform image regions, the probability decreases too quickly to yield meaningful weights. Second, and more importantly, a pixel may be visited by several random walks of different lengths but the weight for the pixel should only depend on whether there is a smooth path from x_0 to x_j, and not on the length of this path. We therefore use the geometric mean of $p(T_{1,j}^i|x_0)$ to define the weight

$$W_j^i = p(T_{1,j}^i|x_0)^{(1/j)}, \tag{3}$$

where j is the number of transitions in the sequence. Thus, in effect, W_j^i is proportional to the average log-transition probability along the path from x_0 to x_j.

The final denoised estimate $\vec{I}(x_0^*)$ is given by

$$\vec{I}(x_0^*) = \frac{1}{C} \sum_{i=1}^{m} \sum_{j=1}^{k} W_j^i \vec{I}(x_j), \tag{4}$$

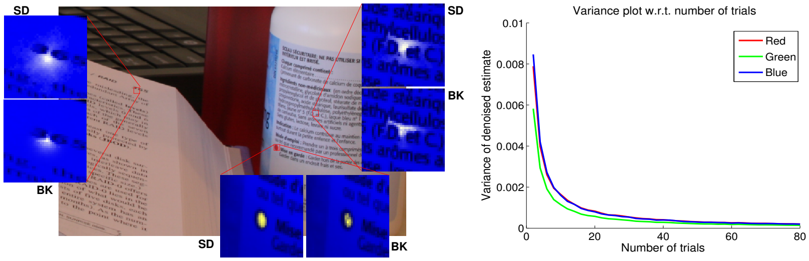

Figure 1: **Left:** Test image and three selected patches. For each patch, we show the weights (indicated by the brightness of the yellow hue) determined by our algorithm (SD) using 25 random walk sequences each with 50 steps. Also shown are the weights obtained by computed blur kernels (BK) on the corresponding patches.**Right:** Variance of the denoised estimate given random walks started with different random seeds as a function of the number of trials. The variance is in gray-levels. Clearly even for 2 trials the estimate is stable.

where $C = \sum_{i=1}^{m} \sum_{j=1}^{k} W_j^i$, is a normalization constant, and $\vec{I}(x_j)$ is the colour of pixel x_j. We note that the above formulation does not consider a distance term like the one proposed for bilateral filtering [24]. Spatial proximity is implicitly encoded in the neighbourhood structure, and the scale of diffusion is controlled by a parameter discussed below.

Typical weight patterns generated by the above procedure are shown on the left in Fig. 1. We compare these weights with the blur kernels computed for the same patches using the formulation in [10], and note that the weights produced by the sampling process are qualitatively very similar to those obtained using the blur kernel computation. This is expected since as the number of steps in the random walk tends towards infinity, the observed distribution over a given neighbourhood would approach the stationary distribution of a Markov matrix M with transitions given by $p(x_{j+1}|x_j)$. These results indicate that the random sampling process is assigning large weights to pixels that would be assigned to the same region by segmentation methods like [10], and [23], or by image matting algorithms such as [15].

There are two remaining details with the above formulation that must be specified to completely define the algorithm. One is the number of trials required to obtain a good estimate for $\vec{I}(x_0^*)$. We expect that computing $\vec{I}(x_0^*)$ with different random seeds will lead to slightly different estimates. We need to determine how many trials are needed in order to obtain an estimate of $\vec{I}(x_0^*)$ that is stable with regard to the randomness of the sampling process. To explore this problem, we compute estimates for a randomly selected subset of pixels on the test image shown in Fig. 1. At each pixel, we estimate the value of $\vec{I}(x_0^*)$ over 250 separate runs of the algorithm starting with different random seeds, and different numbers of trials. We then plot the variance in the estimate of $\vec{I}(x_0^*)$ as a function of the number of trials. Results in Fig. 1 (right) show that only a small number of trials are required to obtain a stable estimate for $\vec{I}(x_0^*)$. The lower variance in the green channel is an artifact of the color cast of the picture used to compute this plot. The explanation for the small variance even on runs with few trials per pixel is found in the length of each random walk. As long as the walks are long enough, few trials are needed to obtain a reliable sampling of the neighbourhood.

The remaining issue is the length of each walk. A simple approach is to simulate a pre-determined number of steps. However we find it more practical to set a threshold on $p(T_{1,k}|x_0)$. Once the probability of a sequence has become less than a small threshold, *e.g.*

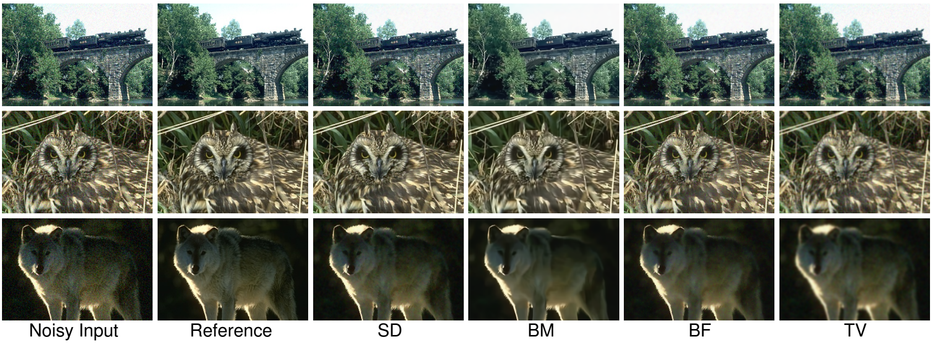

Figure 2: Detail crops of sample BSD images showing typical denoising results for noise with $\sigma = 15$, close inspection reveals that SD clearly achieves the best results, removing most visible noise while preserving detail and avoiding artifacts such as those introduced by BM (see the locomotive picture, for example).

1×10^{-6} the walk is terminated. This has the advantage of stopping early when sequences have crossed strong image boundaries. Notice that were we to continue the walk, the effect of pixels visited along that path would still be small since the probability of the sequence goes down.

4 Evaluation and Comparison

We now provide a thorough evaluation and comparison of stochastic denoising (SD) and three alternate denoising methods representative of the current state-of-the-art. We compare our framework against the bilateral filter (BF) [24] using the implementation provided at [14], against the block matching (BM) algorithm [9] found at [8], and against the total variation (TV) method [5] using the implementation from [11]. Total variation and bilateral filtering serve to compare our method against the closely related, anisotropic diffusion class of algorithms, while block matching has been shown to produce excellent results and represents stats-of-the-art algorithms based on coefficient-shrinkage. [1]

The evaluation was carried out on the 300 images of the Berkeley Segmentation Database (BSD). We evaluated the performance of the algorithms for removal of synthetic Gaussian noise. To each image we added zero-mean Gaussian noise with standard deviations of 5, 10, and 15 (noise is added independently for each colour channel). The resulting 900 noisy images comprise our testing set. The original 300 BSD images form our reference set.

Typically, the success of the denoising process is evaluated by comparing the peak signal-to-noise ratio (PSNR) of the denoised and noisy images with regard to the reference. However, it is well known that PSNR does not correlate well with the visual quality of the results [26]. Instead we use the structural similarity framework of Wang *et al*. [27] with the implementation found at [25]. Their structural similarity index (SSIM) considers three reasonably independent image components: luminance, contrast, and structure. SSIM has been shown to be a better predictor of subjective image quality than PSNR and other existing

[1] To facilitate additional comparisons our algorithm implementation and test data, as well as large versions of all figures in this paper are available at http://www.cs.utoronto.ca/~strider/Denoise.

quality measures. The SSIM index takes values in $[0,1]$ where 1 indicates that the reference and target images are identical.

4.1 Quantitative Comparison and Benchmark

To achieve an evaluation that is independent of input parameters we tested for each algorithm, and for each image, a reasonable range of input parameters, and selected those that yield the optimal SSIM value. This gives an optimistic measure of the denoising ability of each method which assumes that the optimal parameters are known for each test case. Algorithms were tested using uniformly sampled parameters from the following ranges: For stochastic denoising, $\sigma \in [.1,.35]$, sequence stopping threshold $\tau \in [1 \times 10^{-5}, 1 \times 10^{-3}]$, 25 trials. For the bilateral filter, spatial $\sigma \in [2,6]$ pixels, range $\sigma \in [.1,.35]$, and a window size of 21 pixels. For the block matching method we used a $\sigma \in [.1,.35]$. Finally, for the total variation method we used $\lambda \in [.01,1]$. These values were selected to cover the ranges of input parameters that yield useful denoised estimates for each method. In total, the results presented below required $28,000$ denoising algorithm runs.

Results are shown in Fig. 3. Each plot shows the difference in SSIM index between SD and one of the competing methods, if the difference is positive, the result obtained by SD has higher quality (i.e. it better matches the luminance, contrast, and structure of the reference image). Results are sorted by the magnitude of the SSIM difference between SD and each alternate algorithm. This facilitates inspection and shows the fraction of images for which SD produces better results than each competing method. For the worst noise level ($\sigma = 15$), SD outperforms BF in 95% of the images, BM in 70%, and TV in 99.6%. Median SSIM results for all algorithms and noise levels are shown in the table below. The left column gives results when algorithm parameters are optimized for each image. The right columnt gives results for the best, fixed parameter set over the entire test set. Stochastic denoising consistently outperforms the competing methods. Ongoing experiments show that SD also outperforms the non-local means method [4], [16], which achieves a median SSIM of .8210 with optimized parameters for the worst noise level.

Noise	**SD**	**BM**	**BF**	**TV**	**SD**	**BM**	**BF**	**TV**	**Noisy**
Std. dev.	**optimal**	**optimal**	**optimal**	**optimal**	**fixed**	**fixed**	**fixed**	**fixed**	**input**
$\sigma = 5$	**.9728**	.8844	.8952	.8557	**.9727**	.8844	.8952	.8557	.9568
$\sigma = 10$	**.9378**	.8819	.8961	.8270	**.9360**	.8819	.8961	.8270	.8306
$\sigma = 15$	**.9004**	.8797	.8768	.7806	**.8946**	.8797	.8764	.7768	.7542

Fig. 2 shows sample denoising results on typical BSD images. Close inspection shows that SD provides the best results.The SD algorithm preserves fine detail while removing a significant amount of visible noise. In contrast, we find that TV produces soft, blurry results, BF does not remove as much of the noise, and BM, while providing good denoising, sometimes over-smooths textures and introduces artifacts (see the locomotive in Fig. 2).

In terms of run time, stochastic denoising and block matching (the two top methods in terms of denoising performance, and for which we have binary executables running on the same hardware) are similar, consuming close to 15 seconds on BSD-sized images. BF and TV can not be compared directly since they are implemented in Matlab. However we note that total variation is, even in its Matlab implementation, faster than both SD and BM.

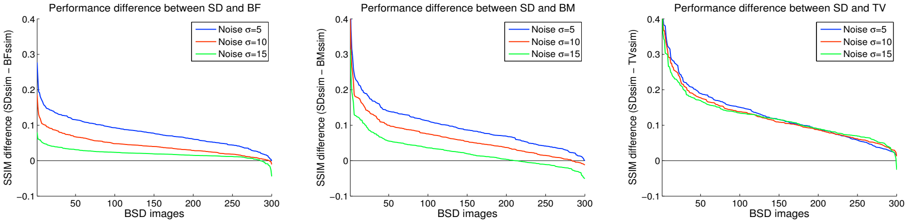

Figure 3: Difference in SSIM index between SD and the three competing algorithms on the BSD. Results are sorted by performance difference to facilitate interpretation. It is clear that SD outperforms the competing methods on most BSD images, and across all noise levels.

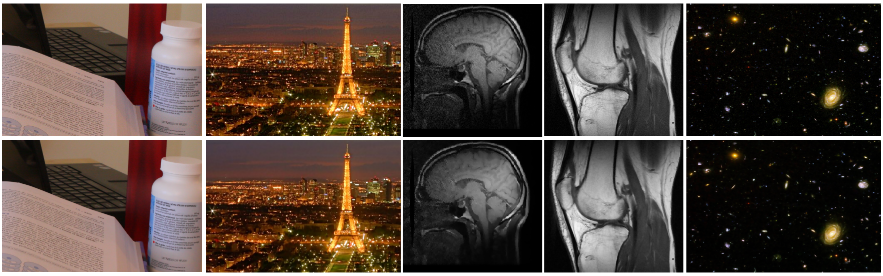

Figure 4: Denoising results for real world applications. **Top Row:** Input images. **Bottom row:** Results from stochastic denoising. We observe that SD does an excellent job of preserving fine image detail while removing visible noise on a variety of input problems.

5 Applications and Extensions

To give a complete picture of the ability of our method to remove noise on typical applications, this section shows denoising results for typical examples of real-world denoising problems. The goal is to illustrate the behaviour of stochastic denoising for images taken under varying conditions, and with different types and amounts of noise. Fig. 4 shows denoising results for several applications. The first two images show cropped regions from high-ISO, low light digital photographs. The next two show typical medical imaging results, and the last picture shows a small region from Hubble's Ultra-Deep Field map containing noise. All of these images suffer from low signal-to-noise ratio.

We observe that stochastic denoising does an excellent job of removing large amounts of visible noise, while accurately preserving very fine detail (see for example, the very small text on the bottle shown in the leftmost image of Fig. 4, and details of stars and galaxies in the Hubble Deep Field). This demonstrates that stochastic denoising provides very strong noise removal capability over a wide range of imaging conditions.

5.1 Multi-pass Denoising

Typically, denoising methods assume that noise is uncorrelated across neighbouring pixels. In reality, camera sensors often produce noise patterns in which larger-scale noise is visible. Regular stochastic denoising will treat such noise as image structure. The question arises of

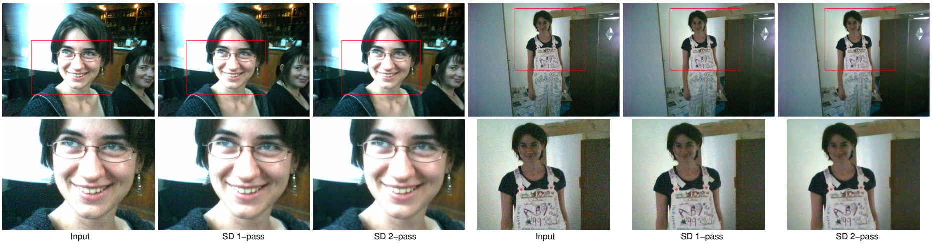

Figure 5: Noisy images from a cell-phone camera and detail crops. These photographs show that for heavily corrupted images, the two pass algorithm achieves excellent denoising results.

whether it is possible to modify SD to remove noise with different spatial scales.

The random walk formulation from Sec. 3 can be extended to incorporate information across larger scales by allowing pixel transitions beyond 3×3 neighbourhoods. The similarity function can also be made to incorporate information across larger regions. Both of these approaches introduce complications, both in the probabilistic formulation of the random walk process, and in terms of deciding what the appropriate scale of analysis should be. Here we describe a simple, computationally cheap way of extending stochastic denoising to perform larger-scale smoothing.

The modified method performs denoising in two passes over the input image. The first pass proceeds exactly as described above, and yields a denoised estimate I^* at the same resolution as the original image. To produce a second denoising estimate computed over larger pixel neighbourhoods we blur and down-sample (by a factor of 2) the input I using a standard Gaussian kernel with $\sigma = 1$. The half-resolution denoised estimate I_h^* is then up-sampled to I_{hu}^* and combined with I^* using at each pixel the following blending equation: $I_{final}(x,y) = \alpha(x,y)I^*(x,y) + (1-\alpha(x,y))I_{hu}^*(x,y)$, where $\alpha = |\nabla I^*|$. Intuitively, this equation states that we care about preserving sharp detail wherever the high resolution denoised estimate contains edge structure. For more uniform image regions, we will use a denoised estimate computed at a coarser scale.

This extension, though very simple, yields excellent results on heavily corrupted images. Fig. 5 shows images taken with a standard cellphone camera. The single-pass denoising estimates, though better than the input, are still fairly noisy. On the other hand, the two-pass estimate computed using the algorithm described above is significantly better. It shows greatly reduced noise and preserves sharp edges and structure. This simple, two-pass denoising method can easily be extended into a multi-scale algorithm, should it be required to remove from the image larger correlated groups of noisy pixels.

It is worth noting that extending the algorithm to perform denoising on 3D data should be relatively straightforward. In its simplest form this would involve random walks defined over $3 \times 3 \times 3$ neighbourhoods centred at each pixel. Motion sequences would require more complex neighbourhood structures, adapted to the optical flow field. We expect the algorithm to remain efficient for 3D data since complexity depends on the length of the random walks, and the size of the neighbourhood contributes only a constant factor. This latter extension remains a topic of ongoing research.

6 Conclusion

We have presented a novel algorithm for image denoising based on simulated random walks on image space. We showed that these random walks produce stable estimates even for few trials, and that the overall behaviour of the random walks approximates that of more computationally expensive blur kernels. Quantitative evaluation of our algorithm shows that it consistently outperforms alternate methods representative of the state-of-the-art in image denoising, and we provide sample results on real-world applications. We expect stochastic denoising will become a useful tool in image processing applications for which noise removal is important. In addition to this, the benchmark provided here will aid in the evaluation of other existing and future algorithms for image denoising.

References

[1] N. Azzabou, N. Paragios, F. Guichard, and F. Cao. Random walks, constrained multiple hypothesis testing and image enhancement. *ECCV*, pages 379–390, 2006.

[2] D. Barash and D. Comaniciu. A common framework for nonlinear diffusion, adaptive smoothing, bilateral filtering and mean shift. *Image and Video Computing*, 22:73–91, 2004.

[3] M. Black, G. Sapiro, D. Marimont, and D. Heeger. Robust anisotropic diffusion. *IEEE TIP*, 7:421–432, 1998.

[4] A. Buades, B. Coll, and J. Morel. A non-local algorithm for image denoising. In *CVPR*, pages 60–65, 2005.

[5] A. Chambolle. An algorithm for total variation minimization and applications. *Journal of Mathematical Imaging and Vision*, 20:89–97, 2004.

[6] T. Chan, S. Osher, and J. Shen. The digital tv filter and nonlinear denoising. *IEEE TIP*, 10:231–241, 2001.

[7] S. G. Chang, B. Yu, and M. Vetterli. Adaptive wavelet thresholding for image denoising and compression. *IEEE TIP*, 9(9):1532–1546, 2000.

[8] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. BM3D image and video denoising software. http://www.cs.tut.fi/~foi/GCF-BM3D/.

[9] K. Davob, R. Foi, V. Katkovnik, and K. Egiazarian. Image denoising with block matching and 3d filtering. In *SPIE Electronic Imaging*, number 6064A-30, 2006.

[10] F. Estrada, A. Jepson, and C. Chennubhotla. Spectral embedding and min-cut for image segmentation. In *BMVC*, pages 317–326, 2004.

[11] P. Guetreuer. Total variation image denoising Matlab implementation. http://www.mathworks.com/matlabcentral/fileexchange/16236.

[12] C. Kervrann and J. Boulanger. Local adaptivity to variable smoothness for exemplar-based image regularization and representation. *IJCV*, 79(1):45–69, 2008.

[13] H. Krim and Y. Bao. A stochastic diffusion approach to signal denoising. In *IEEE Sensor Array and Multichannel Signal Processing Workshop*, pages 236–239, 2000.

[14] D. Lanman. Bilateral filter Matlab implementation. http://www.mathworks.com/matlabcentral/fileexchange/12191.

[15] A. Levin, D. Lischinski, and Y. Weiss. A closed form solution to natural image matting. In *CVPR*, pages 61–68, 2006.

[16] M. Mahmoudi and G. Sapiro. Fast image and video denoising via nonlocal means of similar neighborhoods. *IEEE Signal Processing Letters*, 12(12):839–842, 2005.

[17] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *ICCV*, pages 416–423, 2001.

[18] M. Meila and J. Shi. Learning segmentation by random walks. In *NIPS*, pages 873–879, 2000.

[19] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE PAMI*, 12(7):629–639, 1990.

[20] A. Pizurica, W. Philips, I. Lemahieu, and M. Acheroy. A joint inter- and intrascale statistical model for bayesian wavelet based image denoising. *IEEE TIP*, 11:545–557, 2002.

[21] J. Portilla, V. Strela, M. Wainwright, and E. Simoncelli. Image denoising using scale mixtures of gaussians in the wavelet domain. *IEEE TIP*, 12(11):1338–1351, 2003.

[22] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal. *Physica*, 60:259–268, 1992.

[23] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE PAMI*, 22(8): 888–905, 2000.

[24] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *ICCV*, 1998.

[25] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. SSIM Matlab code. http://www.ece.uwaterloo.ca/~z70wang/research/ssim/.

[26] Z. Wang, A. Bovik, and L. Lu. Why is image quality assessment so difficult. In *IEEE Intl. Conf. Acoustics, Speech, and Signal Processing*, pages 3313–3316, 2002.

[27] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE TIP*, 13(4):600–612, 2004.

[28] J. Weickert. *Anisotropic Diffusion in Image Processing*. Teubner-Verlag, 1998.