Sparse and Redundant Representations and Motion-Estimation-Free Algorithm for Video Denoising

Matan Protter[a] and Michael Elad[a]

[a]The Computer Science Department
The Technion – Israel Institute of Technology
Haifa 32000, Israel.

ABSTRACT

The quality of video sequences (e.g. old movies, webcam, TV broadcast) is often reduced by noise, usually assumed white and Gaussian, being superimposed on the sequence. When denoising image sequences, rather than a single image, the temporal dimension can be used for gaining in better denoising performance, as well as in the algorithms' speed. This paper extends single image denoising method reported in[1,2] to sequences. This algorithm relies on sparse and redundant representations of small patches in the images. Three different extensions are offered, and all are tested and found to lead to substantial benefits both in denoising quality and algorithm complexity, compared to running the single image algorithm sequentially. After these modifications, the proposed algorithm displays state-of-the-art denoising performance, while not relying on motion estimation.

1. INTRODUCTION

Inverse problems are a family of problems, whose target is recovering a high quality signal from a degraded version of it. The simplest of these tasks is image denoising. Due to the vast wealth of image denoising methods, a review would not be provided here; a comprehensive review can be found in[3] and in[4] .

The emergence of noisy image sequences, such as those captured by cellular phones, recorded by webcams, and old archive movies, requires extending image denoising to video denoising. As in the image denoising setting, the noise is assumed to be white, zero mean, iid, and Gaussian. It is therefore natural to consider the application of image denoising methods to each frame of the noisy sequence independently. However, putting the temporal dimension to use results both in improved denoising results and in reduced complexity. This improvement is attributed to the high temporal redundancy in sequences. Due to the high capture rate, image sequences tend to be noisier than images, stressing the need to efficiently use the temporal dimension.

Motion estimation between consecutive frames has been the basis for many video denoising algorithms in the past decade, as in[5–9] . However, several recent contributions[10–14] suggest that explicit motion estimation is in fact not necessary, whilst state-of-the-art results are obtained. These algorithms are described in section 5.

This paper describes a novel video denoising method that does not relay on motion estimation either. It relies on the principles of sparse and redundant representations, described in.[1,2] For a single image, the corrupted image is used for training a sparsifying dictionary describing the image using the K-SVD algorithm.[15,16] A relatively simple state of the art algorithm emerges by incorporating this into a maximum a-posteriori probability (MAP) framework.

The extension of this algorithm to video denoising is described in this paper. The described extensions stem from rigourously modifying the penalty function so it best handles video sequences. One extension is considering 3D patches (spatio-temporal) instead of the original 2D patches. A second is the propagation of the trained dictionary between successive images, followed by a much shorter training process. This is likely to succeed due to the high similarity between consecutive images. The third extension is using patches from nearby images to train a dictionary for a specific image. All these extensions are experimentally tested and are found to

Further author information: (Send correspondence to M.P.). Matan Protter: E-mail: matan.protter@gmail.com , Elad Michael: E-mail: elad@cs.technion.ac.il.

lead to substantial improvement in denoising performance, outperforming all recently published video denoising methods.

The remainder of the paper is structured as follows. Section 2 is devoted to describing the single image algorithm. Section 3 discusses the generalization to video, discussing various options of using the temporal dimension with their expected benefits and drawbacks. Section 4 describes an experimental study designed to test and compare the extensions. Section 5 surveys the existing literature, and compares the leading papers' performance to the algorithm proposed here, demonstrating the superiority of this approach. Section 6 discusses the role of motion in the algorithm's performance. Section 7 summarizes the paper.

2. IMAGE DENOISING USING SPARSITY AND REDUNDANCY

This section provides a brief description of an image denoising method based on sparse and redundant representations, reported in[1,2] . It serves as the foundation for the video denoising we develop in Section 3.

A noisy image $\mathbf{Y}$ results from noise $\mathbf{V}$ superimposed on an original image, $\mathbf{X}$. The noise $\mathbf{V}$ is assumed to be white, zero-mean Gaussian noise, with a known standard deviation σ,

$$\mathbf{Y} = \mathbf{X} + \mathbf{V}, \quad \text{where} \quad \mathbf{V} \sim \mathcal{N}\left\{\mathbf{0}, \sigma^2 \mathbf{I}\right\}. \tag{1}$$

The basic assumption of the denoising method developed in[1,2] is that each image patch (of a fixed size) can be represented as a linear combination of a small subset of patches (*atoms*), taken from a fixed *dictionary*. Noise, due to it being random and unstructured, cannot be modeled this way. The way to denoise an image is therefore to find an image which is both similar to the input image and can be sparsely constructed. Put formally,

$$X = D\alpha \quad , \quad \hat{\alpha} = \arg\min_{\alpha} \|\mathbf{D}\alpha - \mathbf{Y}\|_2^2 + \mu\|\alpha\|_0 \tag{2}$$

The first term requires a proximity between the reconstructed image and the noisy image, and the second term requires that the number of coefficients used in the representation is small. Unfortunately, images are too big to be processed in this manner. Therefore, a patch-based approach has to be adopted instead. Rewriting the penalty function to work on patches yields

$$f_{Still}\left(\{\alpha_{ij}\}_{i,j}, \mathbf{X}\right) = \lambda\|\mathbf{X} - \mathbf{Y}\|_2^2 + \sum_{ij\in\Omega} \|\mathbf{D}\alpha_{ij} - \mathbf{R}_{ij}\mathbf{X}\|_2^2 + \sum_{ij\in\Omega} \mu_{ij}\|\alpha_{ij}\|_0. \tag{3}$$

In this formulation, each image patch (denoted by* $\mathbf{R}_{ij}\mathbf{X}$) is represented by a set of coefficients α_{ij}, which is required to be small for every patch. Patches are weighted using the the values μ_{ij}. Minimizing this functional with respect to its unknown yields the denoising algorithm.

The choice of $\mathbf{D}$ is of high importance to the performance of the algorithm. In[1,2] it is shown that training can be done by minimizing (3) with respect to $\mathbf{D}$ as well (in addition to $\mathbf{X}$ and α_{ij}). The proposed algorithm in[1,2] is an iterative block-coordinate relaxation method, that fixes all the unknowns apart from the one to be updated, and alternating between the following update stages:

1. *Update of the sparse representations* $\{\alpha_{ij}\}$: Assuming that $\mathbf{D}$ and $\mathbf{X}$ are fixed, we solve a set of problems of the form

$$\hat{\alpha}_{ij} = \arg\min_{\alpha} \|\mathbf{D}\alpha - \mathbf{R}_{ij}\mathbf{X}\|_2^2 + \mu\|\alpha\|_0 \tag{4}$$

per each location $[i, j]$. This seeks for each patch the sparsest vector to describe it using atoms from $\mathbf{D}$. In,[1,2] the orthogonal matching pursuit (OMP) algorithm is used for this task.[17–19]

*The matrix $\mathbf{R}_{ij}$ stands for an operator that extracts a patch of fixed size from the image in location $[i, j]$.

2. *Update the dictionary* $\mathbf{D}$: This stage assumes a fixed $\mathbf{X}$. Atoms are updated one at a time in $\mathbf{D}$, together with the coefficients in $\{\alpha_{ij}\}_{ij}$ that use it, using a rank-one approximation of a residual matrix[15,16,20] .

3. *Update the estimated image* $\mathbf{X}$: After several rounds of updates of $\{\alpha_{ij}\}_{ij}$ and $\mathbf{D}$, the final output image is computed by fixing these unknowns and minimizing (3) with respect to $\mathbf{X}$. This leads to the quadratic problem

$$\hat{\mathbf{X}} = \arg\min_{\mathbf{X}} \quad \lambda\|\mathbf{X}-\mathbf{Y}\|_2^2 + \sum_{ij} \|\mathbf{D}\alpha_{ij} - \mathbf{R}_{ij}\mathbf{X}\|_2^2, \tag{5}$$

which is solved by a simple weighting of the represented patches with overlaps, and the original image $\mathbf{Y}$.

The improved results obtained by training a dictionary based on the noisy image itself stem from the dictionary adapting to the content of the actual image to be denoised. An added benefit is that the K-SVD algorithm has noise averaging built into it, by taking a large set of noisy patches and creating a small, relatively clean representative set. More recently, the above described algorithm was generalized to handle color image denoising, demosaicing, and inpainting, leading in all these applications to state-of-the-art results.[21]

3. EXTENSIONS TO VIDEO DENOISING

3.1 Mathematical Formulation

Considering the objective function in Equation (3), extending it to handle image sequences might seem to be a simple task. By letting $\mathbf{Y}$ and $\mathbf{X}$ represent the noisy and clean videos respectively (instead of the noisy and clean images), and adding an index t in the range $[1,T]$ to account for the time dimension, we arrive at a desired penalty term that contains all the forces described in the previous section. This formulation forms one MAP energy function for the entire sequence:

$$\begin{aligned} f^{All}_{Video}(\{\alpha_{ijt}\}_{ijt},\mathbf{X},\mathbf{D}) &= \lambda\|\mathbf{X}-\mathbf{Y}\|_2^2 + \\ &+ \sum_{ij\in\Omega}\sum_{t=1}^{T}\mu_{ijt}\|\alpha_{ijt}\|_0 + \sum_{ij\in\Omega}\sum_{t=1}^{T}\|\mathbf{D}\alpha_{ijt} - \mathbf{R}_{ijt}\mathbf{X}\|_2^2. \end{aligned} \tag{6}$$

Minimizing this functional with respect to its unknowns generates a single dictionary for the entire sequence, and cleans all the images at once with it. The transition to three dimensions appears in the finer details of the algorithm. The patches are transformed into 3D ones, in the sense that they can contain pixels from more than one image. The sliding window operation (collecting the patches), used both for dictionary training and image cleaning, is also three dimensional, as all the patches in the sequence are used in both these tasks.

However, the training of a single dictionary for the entire sequence is problematic. As the scene is expected to change rapidly, objects move in and out of the scene. This means that for the dictionary to be able to suit all images, it is required to change along the sequence.

Following this line, an approach defining a temporally local penalty term is proposed. On one hand, it allows the dictionary to adapt to the scene. On the other hand, it exploits the benefits of the temporal redundancy. A natural such attempt is rewriting the penalty in Equation (6) for each image separately,

$$\begin{aligned} f^{(t)}_{Video}(\{\alpha_{ij}\}_{ij},\mathbf{X}_t,\mathbf{D}_t) &= \lambda\|\mathbf{X}_t-\mathbf{Y}_t\|_2^2 + \\ &+ \sum_{ij\in\Omega}\mu_{ij}\|\alpha_{ij}\|_0 + \sum_{ij\in\Omega}\|\mathbf{D}_t\alpha_{ij} - \mathbf{R}_{ijt}\mathbf{X}\|_2^2, \end{aligned} \tag{7}$$

defined for $t = 1, 2, \ldots, T$. Note that the term $\mathbf{R}_{ijt}\mathbf{X}$ extract a patch of fixed size from the volume $\mathbf{X}$ in time t and spatial location $[i,j]$. This patch may be 3D in general, this way exploiting the temporal axis to our benefit.

In this formulation, only patches centered in the current image are used for training the dictionary and cleaning the image. In the global temporal term as in Equation (6), all the patches in the sequence were used for these tasks. A compromise between temporal locality and exploiting the temporal redundancy is again called

for. This compromise is achieved by also using patches centered in a limited number of neighboring images of the image currently denoised. Introducing this into the penalty term in Equation (7) leads to the modified version,

$$f_{Video}^{(t\pm\Delta t)}\left(\{\alpha_{ijk}\}_{ijk},\mathbf{X}_t,\mathbf{D}_t\right) = \lambda\|\mathbf{X}_t-\mathbf{Y}_t\|_2^2 + \\ +\sum_{ij\in\Omega}\sum_{k=t-\Delta t}^{t+\Delta t}\mu_{ijk}\|\alpha_{ijk}\|_0+\sum_{ij\in\Omega}\sum_{k=t-\Delta t}^{t+\Delta t}\|\mathbf{D}_t\alpha_{ijk}-\mathbf{R}_{ijk}\mathbf{X}\|_2^2, \tag{8}$$

defined for $t=1,2,\ \ldots\ ,T$. This is the penalty term we target in the algorithm that follows. The principles of the algorithm to minimize this functional are similar to those described in Section 2, with obvious modifications made to accommodate the 3D treatment done here. Parameters for the algorithm (such as spatial and temporal patch size) were chosen experimentally. This automatic process is discussed in Section 4.

Minimizing $f_{Video}^{(t\pm\Delta t)}$ as defined in Equation (8) offers one more way to exploit the repetitiveness of the video sequence. As the images $\mathbf{X}_t$ and $\mathbf{X}_{t-1}$ are similar, their corresponding dictionaries are also expected to be similar. This temporal coherence can help speed-up the algorithm. Fewer training iterations are necessary if the initialization for the dictionary $\mathbf{D}_t$ is the one trained for the previous image.

To summarize, minimizing the functional in Equation (8) offers three extensions compared to the naive method of applying the single image algorithm to each image separately, as in:[1,2]

1. *Dictionary propagation:* The initial dictionary for each image is the one trained for the previous one. Fewer training iterations are thus required.
2. *3D Atoms:* Patches are constructed from more than one image, grasping both spatial and temporal common behaviors.
3. *Extended temporal set of patches:* Patches in neighboring frames are also used for dictionary training and image cleaning for each frame.

3.2 Complexity of the Algorithm

The described algorithm is very demanding computationally. This is because of the sparse coding stage (solving equation 4), in which each patch is multiplied by the dictionary. This has to be repeated l times on average, for every atom added to the representation of each patch. Furthermore, the sparse coding stage is repeated for J training iterations. Denoting the patch size (the number of pixels in the patch) by n, and the number of atoms in the dictionary by d, we arrive at the following amount of calculations per pixel:

$$\text{Complexity} = n\cdot\ell\cdot J\cdot d$$

In a nominal case, $n=180$ pixels (a $6\times6\times5$ patch) and the dictionary contains 300 atoms. The average number of atoms in the representation is noise-level dependant, decreasing from 5 for noise level $\sigma=5$ to $0-1$ for noise level of $\sigma=25$. The number of training iterations is usually $J=2$. Assigning these value results in over $200,000$ operations per pixel. This amount grows even larger if patches from neighboring images are also used.

This is a very substantial computational load. However, several ways to decrease this load substantially (gaining $2-3$ orders of magnitude) are possible:

1. The core operation of matrix multiplication can be replaced by an approximation of it (e.g., using singular value decomposition (SVD) on the dictionary[20]),
2. This core operation and the fact that OMP is done independently on each patch, lend this algorithm easily to a parallel implementation, again leading to a substantial speedup.

3. The training of the dictionary is currently done on all image patches. Considering that the initial dictionary is already a good approximation (if it is propagated), complexity can be greatly reduced by using only a subset of the patches for training, with the hope that denoising performance is not sustantially degraded.

4. We have a direct control over the complexity of the algorithm versus its denoising performance by controlling the amount of overlap between the patches to be processed.

4. AN EXPERIMENTAL STUDY

To test the various ideas described in the previous section, as well as coming up with a standard set of parameters, we select a set of four different image sequences - "Football", "Tennis", "Flower Garden", and "Mobile". Each of the four test sets is superimposed with synthetic white Gaussian noise, using noise levels $\sigma = 5, 10, 15, 20, 25, 30, 35, 40,$ and 50. The measure of quality of the denoising result $\hat{\mathbf{X}}$ versus the original video $\mathbf{X}$ is the Peak-Signal-to-Noise-Ratio (PSNR), which is an objective quality measure:

$$PSNR = 10 \log_{10} \left(\frac{255^2 \cdot p}{\|\hat{\mathbf{X}} - \mathbf{X}\|_2^2} \right) \quad [\text{dB}],$$

where both volumes use the scale $0 - 255$. The translation between noise level and PSNR of the noisy sequence appears in Table 1, as the clipping of out-of-range gray-values causes some variation, especially noticed in the strong noise cases.

Noise Sigma	5	10	15	20	25	30	35	40	50
Football	34.1446	28.1398	24.6191	22.1287	20.2348	18.6824	17.3685	16.2616	14.4705
Tennis	34.1596	28.1263	24.6201	22.1398	20.2096	18.6517	17.3242	16.2014	14.3856
Garden	34.1623	28.1567	24.7063	22.2673	20.3964	18.8939	17.6543	16.5809	14.8762
Mobile	34.1785	28.1715	24.6875	22.2590	20.4008	18.9238	17.6862	16.6440	14.9481

Table 1. PSNR of noisy sequences for each sequence and noise level combination. The difference is because of the out-of-range values.

We next describe the tests designed to evaluate the effect of each of the extensions on the denoising performance. Two visual examples of the denoising results can be seen in Figures 1 and 2.

4.1 Three-Dimensional Atoms

A 3D patch is created by taking a block around the pixel (i, j, t) that extends in all axes and is symmetrical around the centeral image. It is therefore not causal[†]. Creating an initial dictionary also has to be addressed. We chose to simply replicate each atom in the overcomplete DCT dictionary, for $2\Delta t + 1$ times to create a three dimensional atom of the wanted width.

In Figure 3 a comparison between the performance of 3D atoms and 2D atoms is shown. For the 3D case, one set of parameters (noise-level dependent) is used for all movies (all patches are 5 images wide). For the 2D case, it is difficult to find one set of parameters that does justice to all movies. We therefore use the optimal set found for each movie, as this does not change the conclusions drawn from this comparison.

A better understanding of the reasons the 3D atoms greatly outperform the 2D ones can be gained by looking at the content of the dictionary. Figure 4 shows several atoms from the trained 3D dictionary, for image #10 of the "garden" sequence. This sequence has a camera motion to the right, so the entire scene moves to the left. The described motion can be seen clearly in the atoms. The central part of the atom (coming from the current image) has moved to the left compared to the top part (coming from the previous image). In the same manner, the bottom part (coming from the next image) has moved to the left relative to the center part.

The question of what happens to the dictionary when the motion is not global naturally arises. In such cases, the dictionary has several atoms reflecting each of the objects and motions.

[†]Causality can be enforced, but was found to lead to slightly inferior performance.

Figure 1. Garden Sequence (1 every 30 frames) with $\sigma = 50$. Left: Original Frame; Middle: Noisy Frame; Right: Cleaned frame.

Complexity-wise, the improved performance seems to come at a high price - five times the computations. In practice, this is not true, as the optimal spatial size of the patches become smaller when turning to use three dimensional patches. This partially compensates for the temporal width.

4.2 Dictionary Propagation

As explained above, propagation of the dictionary can be accompanied by fewer training iterations to assist in reducing computation. We test how many iterations are required for obtaining similar results to the none-propagation option (using 15 training iterations per image). Figure 5 presents the results of this experiment.

The clear conclusion from these results is that propagation is crucial and leads to improved denoising performance. More insight to the reasons for such an improvement can be seen in Figure 6, that shows the dictionaries trained for frame #30 of the "garden" sequence. The left dictionary is the one trained after propagating the dictionary (from image #10), using 4 training iterations for each image. The right part of the figure shows the trained dictionary from the DCT using 15 training iterations. This comparison shows that propagation of the dictionary leads to a cleaner version with clearer and sharper texture atoms. These benefits are attributed to the memory induced by the propagation.

A second conclusion is that the number of training iterations should depend on the noise level. It appears that the less noisy the sequence, the more training iterations are needed for each image. In higher noise levels, adding more iterations hardly results in any denoising improvement.

Based on this finding, all of the following experiments were run with propagation of the dictionary. The number of training iterations is adapted to the noise level, in accordance with these results. In a real system,

Figure 2. Mobile Sequence (1 in 30 frames) with $\sigma = 40$. Left: Original Frame; Middle: Noisy Frame; Right: Cleaned frame.

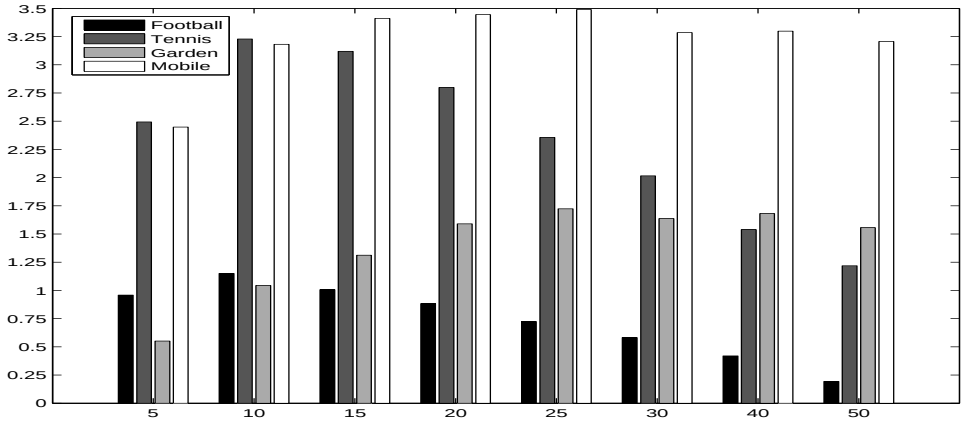

Figure 3. PSNR gain (difference in dB relative to the 2D atoms' method) achieved by using 3D atoms versus 2D atoms.

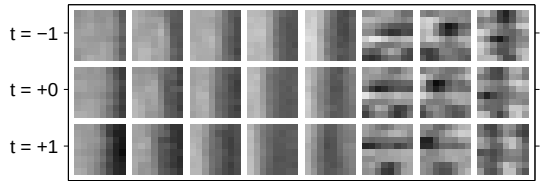

Figure 4. Several (8) 3D atoms (each of size $8 \times 8 \times 5$, but only the temporal center is shown) trained for image #10 of the garden sequence. Each vertical column is one atom.

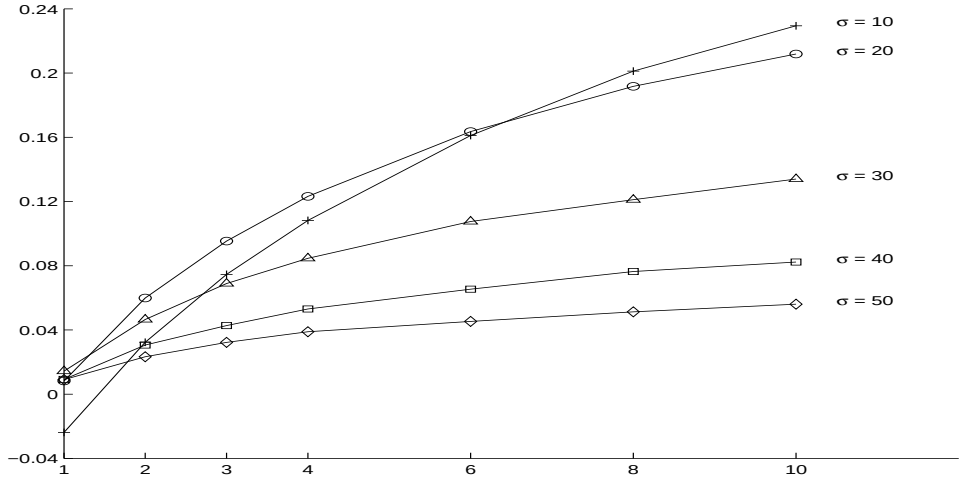

Figure 5. PSNR gain (average over all sequences) by propagating the dictionary and using the specified number of training iterations for each image. The reference option is with no propagation, and 15 K-SVD iterations.

some scene-cut detection algorithm to reset the dictionary when the scene changes at once is required.

4.3 Extended Temporal Training Set

As described above, it is possible to use patches centered in neighboring images both for dictionary training and image cleaning. We have experimentally that using patches centered one image away (i.e. taken from 3 images) leads to an average improvement of $0.2 - 0.4$ dB in denoising performance. Using patches centered further away did not lead to any further improvement.

This idea is parallel, but inverse to the one presented above, of using less patches for training, aimed at reducing the computational complexity.

4.4 Other Parameters

We have also experimented with the other parameters of the algorithm for their effect on denoising results. We found, for example, that the blocks should be slightly smaller spatially when the noise level increases.

We have also found that at high noise levels, the redundancy factor (the ratio between the number of atoms in the dictionary to the size of an atom) should be smaller. At high noise levels, obtaining a clean dictionary requires averaging of a large number of patches for each atom. This is why only a relatively small number of atoms is used. At low noise levels, many details in the image need to be represented by the dictionary. Noise averaging takes a more minor role in this case. This calls for a large number of atoms, so they can represent the wealth of details in the image.

4.5 Summary: Extensions to Video Results

The described tests were used for selecting a single set of parameters (as a function of the noise level). This set includes propagation of the dictionary, three-dimensional atoms that are five images wide, and an extended

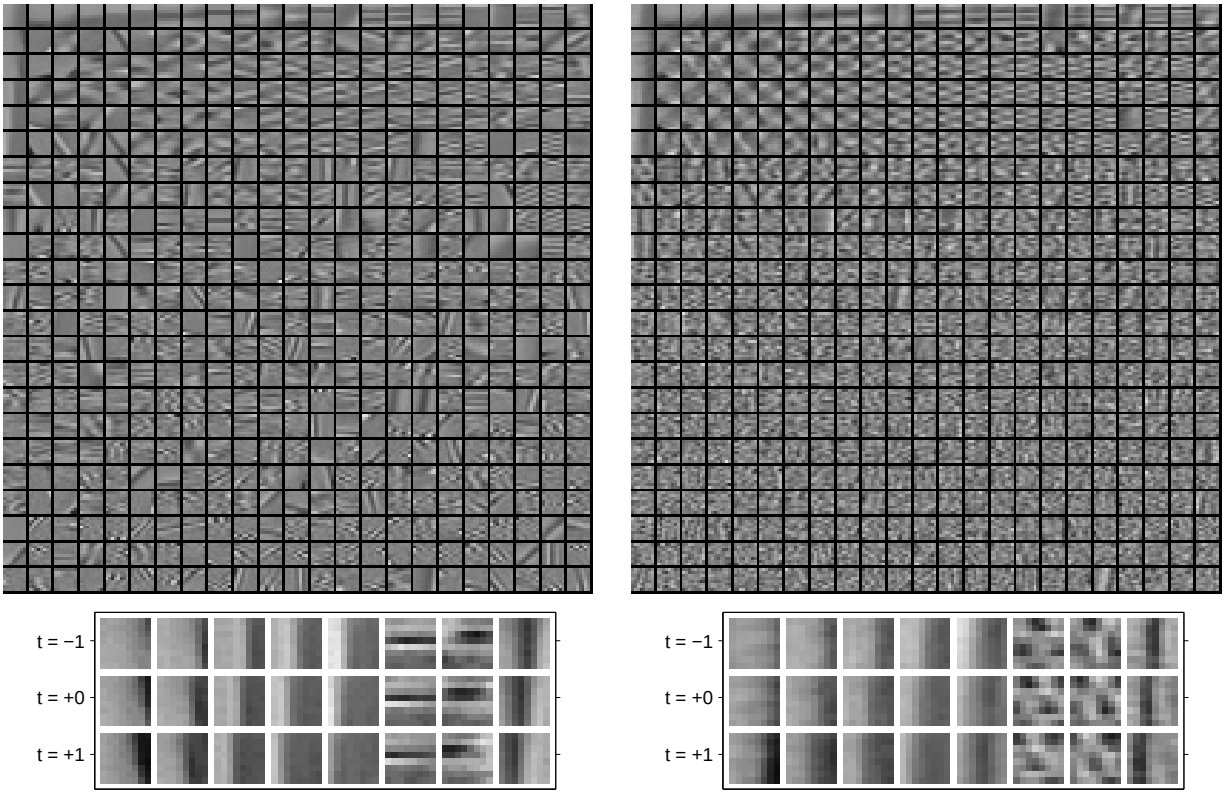

Figure 6. Dictionaries trained by propagating (left) and not propagating (right) the dictionary between images. Top: Central part (time-wise) of each dictionary. Bottom: Several enlarged atoms, showing the three center temporal layers.

patches set that extends one image in every direction. A comparison between the final 3D algorithm and the original single image algorithm can be seen in table 3, which appears in the following section.

5. COMPARISON TO STATE-OF-THE-ART

5.1 Overview of other methods

Traditional video denoising techniques rely on motion estimation for noise suppression. Indeed, it has been assumed for a long time that motion estimation is the foundation of any inter-frame video denoising algorithm. A variety of such works are described in[5–10] . However, several recently published papers are disproving this belief. The first of these is the Non-Local Means (NL-Means) algorithm reported in.[3,11] Instead of an explicit motion estimation for each patch, all the patches in its three dimensional neighborhood are considered and their center pixels are averaged. Each patch is weighted according to its similarity to the center patch, making the motion-estimation a fuzzy and implicit one.

While this approach is very simple, its denoising results were better than any method previously published, leading the authors to justifiably claim "*Image sequence denoising does not require motion estimation*". Later published methods, such as[13,14] continue to avoid explicit motion estimation. These three methods[11,13,14] have been shown to be the leading in video denoising performance, and therefore we concentrate on their performance when comparing our results to the state-of-the-art.

Both methods reported in[13,14] use statistical approaches to select optimal filtering window size and neighborhood search area. The method described in[13] proceeds with a weighted averaging of the patches in the selected neighborhood. This method uses motion estimation only when the expected motion is large. The method in[14] operates differently, by concatenating the found similar patches to a 3D volume, and transforming it using a

unitary transform. The obtained transform coefficients are hard thresholded for noise attenuation. The center pixel in the reconstructed patches in the volume is averaged and represents the denoised outcome.

We now turn to present a comprehensive comparison of these three methods and ours. Note that since the above papers chose different video sequences to test on, we provide several groups of tests, to address each.

5.2 Comparison Results

We first compare the proposed method to the work reported in,[13] which displayed superior denoising results relative to other methods it was compared to. We synthesized the same experiments as those in,[13] and report the results in Table 2. It is clear that our proposed method performs better in these tests.

Test Sequence	Input PSNR [dB]	[13] Results [dB]	Our Results [dB]	Difference [dB]
Salesman	28	35.13	37.68	+2.55
	24	32.60	35.41	+2.81
Garden	28	31.33	31.93	+0.60
Miss America 1	28	39.39	40.21	+0.82
Suzie	28	37.07	37.45	+0.38
	24	35.11	35.42	+0.31
Trevor	28	36.68	37.77	+1.09
	24	34.79	35.72	+0.93
Foreman	28	34.94	37.44	+2.50
	24	32.90	35.42	+2.52

Table 2. Results of the proposed algorithm compared to those reported in.[13] The chosen sequences and noise powers are those reported in.[13]

The proposed method was also compared to the results of the classic NL-Means[3,11] and to the current benchmark in video denoising, the SW3D.[14] The NL-means' parameters were varied so they best fit each test, since no one common set of parameters was found for all tests. This optimization gives the NL-means an advantage, compared to using a fixed set of parameters as in the proposed algorithm and the SW3D. For the SW3D, the authors of[14] were very kind to provide us with an improved implementation of their algorithm. We note that the results reported here for their algorithm are better than those reported in.[14] The results of these three methods, along with the original K-SVD denoising applied on single images,[1] are compared in Table 3.

It is evident from the comparison that the proposed algorithm and the SW3D obtain the best results. At low noise levels, the SW3D slightly outperforms the proposed algorithm on most tests, but under-performs on the rest. At higher noise levels, the proposed algorithm clearly displays the best denoising results. Averaging the above results, we get that an average performance (from the best downwards) of 29.01dB for our method, 28.12dB for the SW3D, 27.92dB for the NL-Means, and finally, 26.95dB for the single-frame K-SVD algorithm. We note that these results mean that the proposed extension of[1,2] to handle video yields about 2 dB better results on average than the single image method.

6. WHO NEEDS MOTION ESTIMATION?

There is no need for motion estimation in the proposed algorithm, as the noise averaging is not based on simple temporal proximity. Instead, the dictionary encodes the temporal structure for the sequence. Avoiding the need to estimate motion is very beneficial, as accurate motion estimation in a noisy setting is very hard to obtain. A second benefit, shared with the NL-Means and the SW3D, is that many patches are used for driving out the noise, instead of the just one "true" patch. This allows better denoising, as more patches are averaged.

Having said all the above, we should also add that the motion-estimation-free algorithm proposed here, and the competitive ones mentioned above, all lean on several core assumptions about the motion in the sequence:

1. *Image content is preserved:* The assumption that most of the content in one image also exists in the next is standard in the video denoising field. Otherwise, the task is essentially a series of image denoising tasks.

σ	Football		Tennis		Garden		Mobile		MeanPSNR	
5	**37.2357**	35.9529	**38.3428**	37.1241	**36.6256**	36.0330	37.0691	36.2199	37.3183	36.3325
	36.0284	37.1901	35.4216	38.0578	35.6874	36.3532	35.4926	**38.0878**	35.6575	**37.4222**
10	**33.2386**	31.8212	**34.7098**	32.0496	**32.1844**	31.4657	32.9308	31.7220	33.2659	31.7646
	31.7345	33.0584	30.8462	34.2097	30.6805	31.9738	30.4723	**34.0107**	30.9334	**33.3132**
15	**31.1099**	29.6079	**32.5621**	29.5608	**29.7780**	29.0423	30.6341	29.4250	**31.0210**	29.4090
	29.5922	30.7114	28.7770	31.9739	27.9585	29.4284	27.7225	**31.3976**	28.5125	30.8778
20	**29.4733**	28.1421	**30.6886**	28.1434	**28.0680**	27.5580	28.8675	27.7251	29.2743	27.8922
	28.1345	29.1450	27.5933	30.4767	26.1474	27.9057	25.8322	**29.5829**	26.9269	**29.2776**
25	**28.1599**	27.1243	28.7156	27.3494	26.4882	26.3790	27.1189	26.3689	27.6206	26.8054
	27.0807	27.9121	26.7759	**29.1991**	24.7780	**26.6054**	24.3940	**28.0947**	25.7572	**27.9528**
30	**26.9412**	26.5247	27.1787	26.6819	24.5829	25.3323	24.8755	25.3558	25.8946	25.9737
	26.2691	26.9307	26.1996	**28.2695**	23.7193	**25.6505**	23.2735	**26.9042**	24.8654	**26.9387**
35	25.6763	25.8011	25.8921	26.2920	22.6928	24.4250	22.4002	24.4399	24.1654	25.2395
	25.5697	**26.1496**	25.7180	**27.5211**	22.8490	**24.5586**	22.2319	**25.7132**	24.0922	**25.9856**
40	24.5183	24.9666	24.6714	25.9348	21.4826	23.5144	20.9477	23.6005	22.9050	24.5041
	24.9585	**25.4672**	25.3091	**26.9225**	22.1057	**23.8031**	21.4587	**24.8941**	23.4580	**25.2717**
50	23.2540	24.1070	23.5034	25.3388	20.1516	22.1568	19.5466	21.9327	21.6139	23.3838
	23.9200	**24.2172**	24.6424	**25.9818**	20.9143	**22.5327**	20.0299	**23.3265**	22.3766	**24.0145**

Table 3. Comparison of the denoising results of several methods on a number of test sequences and noise levels. Top Left: SW3D; Top Right: NL-Means; Bottom Left: K-SVD single image; and Bottom Right: The proposed algorithm. Best result for each set is written in bold. Results are for images 10-20 of each set, using other images (for temporal filtering) as necessary.

2. *Motion pattern is preserved:* When using 3D atoms, it is assumed that similar objects undergo similar motion. Furthermore, it is also assumed that this motion is roughly preserved (or changes slowly) between one image to the next. This assumption is usually true in video sequences with high capture rate.

3. *Motion is mainly translational:* This assumption stems from the way the patches are used. Handling more complex motion patterns is possible, but we have not explored this option in this work.

Note that there is no assumption that the motion is small, nor is there is a search area parameter in the proposed algorithm. A strong indication that all these assumptions are generally true and are not limiting is the very good performance of the algorithm on the above described tests.

7. CONCLUSIONS

In this paper we propose an image sequence denoising algorithm based on sparse and redundant representations. This algorithm is based on the single image denoising algorithm introduced in.[1,2] The extension of this basic algorithm to handle image sequences is discussed both on the mathematical level and in practical terms. Three extensions are proposed: the use of spatio-temporal (3D) atoms, dictionary propagation coupled with fewer training iterations, and an extended patch-set for dictionary training and image cleaning. All these extensions are thoroughly tested on an extensive set of image sequences and noise levels, and found to dramatically improve denoising performance. The proposed algorithm is also compared to other state of the art methods, and shown to produce comparable or favorable results. Finally, the need, or rather no-need, for motion estimation, is discussed.

Acknowledgements

The authors would like to thank the authors of,[14] D. Rusanovskyy, K. Dabov, and Prof. K. Egiazarian, for their willingness to provide us with a working implementation of the SW3D algorithm. A special thanks goes to K. Dabov, who provided us with the source code of the improved SW3D and the test sequences used to test it on. The authors would also like to thank the authors of[13] , J. Boulanger, C. Kervrann, and P. Bouthemy, for their willingness to help in comparing the denoising methods.

REFERENCES

1. M. Elad and M. Aharon, "Image denoising via learned dictionaries and sparse representation", Proc. of the *International Conference on Computer Vision and Pattern Recognition (CVPR)*, New-York, June 17-22, 2006.
2. M. Elad and M. Aharon, "Image denoising via sparse and redundant representation over learned dictionaries", *IEEE Trans. on Image Processing*, Vol. 15, No. 12, pp. 3736–3745, December 2006.
3. A. Buades, B. Coll, and J.M. Morel, "A review of image denoising algorithms with a new one", *Multiscale Modeling and Simulation*, Vol. 4(2), pp. 490–530, 2005.
4. J. Portilla, V. Strela, M.J. Wainwright, and E.P. Simoncelli, "Image denoising using scale mixtures of gaussians in the wavelet domain", *IEEE Trans. On Image Processing*, Vol. 12, No. 11, pp. 1338–1351, November 2003.
5. V. Zlokolica, A. Pizurica, and W. Philips, "Recursive temporal denoising and motion estimation of video", Proc. of the *International Conference on Image Processing (ICIP)*, Singapore, October 2004.
6. F. Jin, P. Fieguth, and L. Winger, "Wavelet video denoising with regularized multiresolution motion estimation", *EURASIP Journal on Applied Signal Processing*, Vol. 2006, pp. 01–11, 2006.
7. I. Selesnick and K.Y. Li, "Video denoising using 2d and 3D dual-tree complex wavelet transforms", *Wavelet Applications in Signal and Image Processing X (SPIE)*, San-Diego, August 2003.
8. R. Dugad and N. Ahuja, "Video denoising by combining Kalman and Wiener estimates", Proc. of the *International Conference on Image Processing(ICIP)*, Kobe, Japan, October 1999.
9. N.M. Rajpoot, Z. Yao, and R.G. Wilson, "Adaptive wavelet restoration of noisy video sequences", Proc. of the *International Conference on Image Processing (ICIP)*, Singapore, October 2004.
10. R.G. Wilson, and N.M. Rajpoot, "Image volume denoising using a fourier-wavelet basis", Proc. of the *6th Baiona Workshop on Signal Processing in Communications (Baiona SPC'03)*, Baiona, Spain, September 2003.
11. A. Buades, B. Coll, and J.M. Morel, "Denoising image sequences does not require motion estimation", Proc. of the *IEEE Conf. on Advanced Video and Signal Based Surveillance September (AVSS)*, pp. 70–74, 2005.
12. V. Zlokolica, M.D. Geyer, S. Schulte, A. Pizurica, W. Philips, and E. Kerre, "Fuzzy logic recursive change detection for tracking and denoising of video sequences", *Image and Video Communications and Processing 2005 (SPIE)*, Vol. 5685, pp. 771–782, March 2005.
13. J. Boulanger, C. Kervrann, and P. Bouthemy, "Adaptive space-time patch-based method for image sequence denoising", Proc. of the *International Workshop on Statistical Methods in Multi-Image and Video Processing (SMVP)*, Graz, Austria, May 2006.
14. D. Rusanovskyy, K. Dabov, and K. Egiazarian, "Moving-window varying size 3D transform-based video denoising", Proc. of the *International Workshop on Video Processing and Quality Metrics (VPQM)*, Scottsdale, USA, 2006.
15. M. Aharon, M. Elad, and A.M. Bruckstein, "On the uniqueness of overcomplete dictionaries, and a practical way to retrieve them", *Journal of Linear Algebra and Applications*, Vol. 416, pp. 48-67, July 2006.
16. M. Aharon, M. Elad, and A.M. Bruckstein, "The K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representation", *IEEE Trans. On Signal Processing*, Vol. 54, No. 11, pp. 4311–4322, November 2006.
17. S. Mallat and Z. Zhang,"Matching pursuit in a time-frequency dictionary", *IEEE Trans. Signal Processing*, Vol.41, No. 12, pp. 3397–3415, December 1993.
18. J.A. Tropp,"Greed is good: algorithmic results for sparse approximation", *IEEE Trans. Information Theory*, Vol. 50, No. 10, pp. 2231–2242, October 2004.
19. D.L. Donoho, M. Elad, and V. Temlyakov, "Stable recovery of sparse overcomplete representations in the presence of noise", *IEEE Trans. On Information Theory*, Vol. 52, pp. 6–18, January 2006.
20. G.H. Golub and C.F.Van-Loan, *Matrix Computations*, third ed., John Hopkins University Press, Baltimor, 1996.
21. J. Mairal, M. Elad, and G. Sapiro, "Sparse Representation for Color Image Restoration", submitted to *IEEE Trans. on Image Processing.*