Towards Natural Image Denoising by Sparse Code Shrinkage: Improvements and Applications

Ying Yu Jian Yang Dan Xu

School of Information Science and Engineering, Yunnan University, Kunming 650091, P. R. China

Abstract

Sparse code shrinkage (SCS) has been proved to be a very promising method for natural image denoising, but it still has some drawbacks such as considerable complexity of computation and inevitable loss of image details. In this paper, we propose a new compensation operation and an improved shrinkage function to effectively combat the loss of image details in SCS. We also propose a new strategy of sliding window and non-square ICA filters to relieve the computational complexity of SCS in real-time system. Compared with ordinary SCS, our method can remain better performance while achieving higher efficiency in natural image denoising.

Keywords: Sparse code shrinkage, Feature extraction, Independent component analysis, Image denoising.

1. Introduction

The classic experiments of Hubel and Wiesel [1] have suggested that neurons with line and edge selectivities found in primary visual cortex of cats and monkeys form a sparse-distributed representation of natural scenes, and it has been reasoned that such response should emerge from an unsupervised learning algorithm that attempts to find a factorial code of independent visual features. Barlow was thus let to propose that our visual cortical feature detectors might be the end result of a "redundancy reduction" process [2] [3], in which the activation of each feature detector is supposed to be as statistically independent from the others as possible. Such a "factorial code" potentially involves dependency of all orders, but most studies have used only the second-order statistics required for decorrelating the outputs of a set of feature detectors.

A variety of Hebbian feature-learning algorithms for decorrelation have been proposed [4] [5]. One popular decorrelating solution is principal components analysis (PCA), but the principal components of natural scenes amount to a global spatial frequency analysis [6]. Therefore, second-order statistics alone do not suffice to predict the formation of localized edge detectors.

Field [7] [8] has argued for the importance of sparse, or "minimum entropy", coding [9], in which each feature detector is activated as rarely as possible. This has led to feature-learning algorithms [10], the most successful of which has been the Olshausen and Field [11] demonstration of the self-organization of local, oriented receptive fields using a sparseness criterion.

Bell and Sejnowski demonstrated the ability of this nonlinear information maximization [12] to find statistically independent components to solve the problem of separating mixed sources. This "Independent Components Analysis" (ICA) problem [13] is equivalent to Barlow's redundancy reduction problem. From a viewpoint of "projection pursuit" [14] [15], ICA makes the most non-Gaussian direction as the aim of projection pursuit. In 1996, Olshausen and Field [16] described that some basis-functions, obtained by ICA from natural images, were similar to response of receptive field of simple cells in visual cortex. Bell and Sejnowski [17] presented that the "independent components" of natural scenes are edge filters and the sparse encoding of image could be implemented by ICA. Zhang and Mei [18] have mathematically proved the relationship between ICA-filter and simple cell's receptive field in vision system.

After 1999, Hyvärinen et al. proposed a method of sparse code shrinkage (SCS) [19] [20], which has been proved to be a promising method for natural image denoising and can achieve a better performance than any other traditional method. PCA and Wiener filtering only consider the second-order statistical properties in natural image data. Small wonder, sparse code shrinkage, using higher order moment in image data, will take the advantage. The wavelet transform depends largely on some abstract mathematical properties which nearly have no relation to statistical properties of natural data. Compared to the method of wavelet shrinkage [21]-[23], SCS has the important benefit that the features are estimated directly from data.

However, SCS still has some annoying drawbacks in some applications. For instance, the algorithm is computationally demanding which may pose great difficulties in real-time processing. Another drawback

is that it will inevitably lose some image edge details due to the adoption of soft-threshold although it achieves good performance in image denoising.

In view of these problems, we present some new methods. On one hand, we adopt steady-transition threshold, instead of soft-threshold, and propose a compensation operation to avoid losing details in filtering procedure. On the other hand, we propose a new sliding and non-square window method to greatly alleviate the pressure of computation in real-time processing. In addition, we have been successful in applying our improved sparse code shrinkage method in the infrared video processing.

The remainder of this paper is organized as follows. In Section 2, we simply introduce the ICA method used in this paper. In Section 3, we design a new compensation operation and a shrinkage function to improve the performance of SCS. In addition, we describe in Section 4 how to use a simple sliding window method to relieve the great complexity of computation. In Section 5, we conduct experiments to compare our method with traditional SCS method. Finally, we introduce in Section 6 the applications with our SCS method and conclude in Section 7.

2. Independent component analysis

ICA is one of the methods to solve linear equations based on stochastic vector, which is shown as Eq. (1):

$$\mathbf{X} = \mathbf{AS}, \qquad \mathbf{X} \in \mathbf{R}^m, \quad \mathbf{A} \in \mathbf{R}^{m \times n}, \quad \mathbf{S} \in \mathbf{R}^n. \tag{1}$$

Given a stochastic vector $\mathbf{X} = (x_1, x_2, ..., x_m)^{\mathrm{T}}$, unknown matrix $\mathbf{A}$ and source signal vector $\mathbf{S} = (s_1, s_2, ..., s_n)^{\mathrm{T}}$, in general, Eq. (1) has no solution. However, if the elements of $\mathbf{S}$ are mutually independent and non-Gaussian distribution, and $m \geq n$, we can apply an unsupervised neural network to solve it. If there is a weight matrix $\mathbf{W}$ that satisfies the following equation

$$\mathbf{WX} = \mathbf{WAS} \approx \mathbf{S}, \tag{2}$$

then $\mathbf{A}$ and $\mathbf{S}$ can be derived, which $\mathbf{W} \approx \mathbf{A}^{-1}$ for $m = n$ and $\mathbf{W} \approx \mathbf{A}^{+}$ for $m \neq n$, $\mathbf{A}^{+}$ is pseudo-inverse matrix of $\mathbf{A}$. Here, the row vector of $\mathbf{W}$ is called ICA filter. Accordingly, the corresponding column vectors of $\mathbf{A}$ are named ICA basis function.

There are several algorithms to implement ICA method. In this paper, FastICA proposed by Hyvärinen [24] [25] is used, in which the stochastic vector $\mathbf{X}$ is prewhitened, normalized and the weight vectors are orthogonalized in process. The learning rule of the neural network on the ith unit is

$$\mathbf{W}_i^{+} = E\{\mathbf{x}\, g(\mathbf{W}_i^{\mathrm{T}} \mathbf{x})\} - E\{g'(\mathbf{W}_i^{\mathrm{T}} \mathbf{x})\}\, \mathbf{W}_i, \tag{3}$$

$$\mathbf{W}_i = \mathbf{W}_i^{+} / \|\mathbf{W}_i^{+}\|, \tag{4}$$

where $E\{.\}$ is expectation and $g(u) = \tanh(u)$, $\mathbf{W}_i^{+}$ is un-normalized weight vector for the ith unit. In learning phase, weight vector $\mathbf{W}_i^{+}$ of the neural network is updated by Eqs. (3) and (4) for learning image set $\{\mathbf{x}(1), \mathbf{x}(2), ..., \mathbf{x}(k)\}$. All weight vectors of the neural network are updated one by one using the same procedure, and then the weight matrix $\mathbf{W}=(\mathbf{W}_1, \mathbf{W}_2, ..., \mathbf{W}_n)^{\mathrm{T}}$ is orthogonalized by the following formula:

$$\mathbf{W} \leftarrow (\mathbf{W}\mathbf{W}^{\mathrm{T}})^{-1/2}\, \mathbf{W}. \tag{5}$$

When the iterations of formula (3) – (5) converge, we can obtain the final weight matrix $\mathbf{W}$, rows of which is orthogonalized each other.

3. Improving SCS

In our experiments, we use the fixed-point FastICA algorithm [24] [25] to perform the estimation of ICA transform. After we estimate the ICA transform from the natural image data, we can realize the sparse coding of natural images. An illustration of sparse coding of natural image is shown in Fig. 1. We can see that the energy of image after ICA transform is concentrated on several components and other components are nearly close to zero, which means that the redundancy of natural image data is reduced. Here, 'normalized' means that the image data have been converted to zero mean and unit variance.

The shrinkage of sparse code is the key stage of image denoising. The denoising procedure of sparse code shrinkage (SCS) is simple: we transform the data into a representation with suitable properties, shrink the components using the shrinkage function, and invert the transformation.

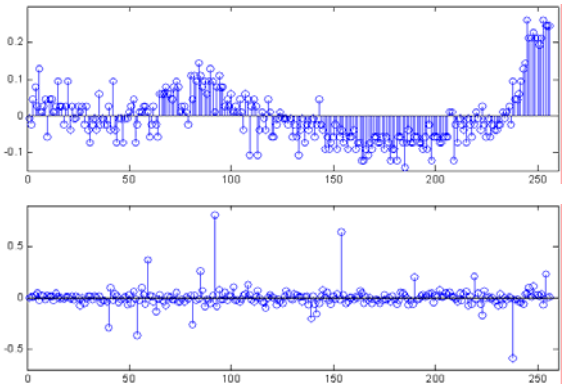

Fig. 1: Illustration of sparse coding of natural image data. Upper: A normalized image patch (16-by-16) reshaped into a 256-dimensional vector. Lower: Sparse code vector of the image patch after ICA transform.

3.1. Choice of shrinkage function

Usually, there are two types of shrinkage function: hard-threshold and soft-threshold [26].

- Hard-threshold

$$\delta_{\lambda}^{H}(x)=\begin{cases}x & |x|\geq\lambda\\ 0 & |x|<\lambda\end{cases} \tag{6}$$

- Soft-threshold

$$\delta_{\lambda}^{S}(x)=\begin{cases}sign(x)(|x|-\lambda) & |x|\geq\lambda\\ 0 & |x|<\lambda\end{cases} \tag{7}$$

Here, λ is threshold in sparse code shrinkage.

Hard-threshold method shrinks to zero the components whose absolute value is smaller than the threshold, and makes other components intact. It will bring some unexpected noise. We can notice that there are two broken points in the hard-threshold shrinkage function. Just the two broken points would associate with some artificial noise.

In view of the drawback of hard-threshold, soft-threshold method is more preferable in sparse code shrinkage. It shrinks to zero the components whose absolute value is smaller than the threshold, and subtract the threshold value from other components, which secures the global continuity in shrinkage function. Unfortunately, it will inevitably lose some details and features in the image due to the shrinkage of all components, particularly in the case of applying a great threshold value.

In this paper, we propose the steady-transition threshold shrinkage function, with two transitional belts instead of broken points, which not only avoids bringing in unexpected noise but also makes large components untouched.

- Steady-transition threshold

$$\delta_{\lambda,\eta}(x)=\begin{cases}0 & |x|\leq\eta\lambda\\ sign(x)\dfrac{|x|-\eta\lambda}{(1-\eta)} & \eta\lambda<|x|\leq\lambda\\ x & |x|>\lambda\end{cases} \tag{8}$$

Here, the constant η is smaller than 1 ($\eta\approx 0.85$).

We also propose a smooth-transition threshold shrinkage function whose first-order derivative is continuous. But we can hardly tell any difference from the denoising results of steady-transition and smooth-transition threshold. Moreover, the steady-transition threshold is easier to implement than the smooth one. So the former is preferable. Plots of different types of shrinkage function are shown in Fig. 2. The effect of using steady-transition threshold will be showed in Section 5.

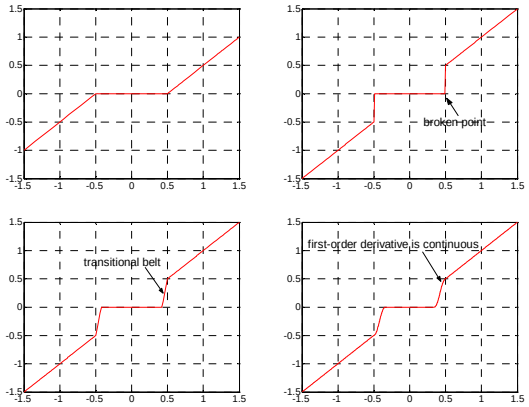

Fig. 2: Plots of the shrinkage function (Threshold = 0.5). Upper left: Soft-threshold. Upper right: Hard-threshold. Lower left: Steady-transition threshold. Lower right: Smooth-transition threshold.

3.2. Compensation operation after shrinkage

However, whatever shrinkage function we choose, the shrinking operation will inevitably result in some loss of edge features in the image. So we appropriately compensate the components by multiplying them with a compensation factor β (usually $1.05<\beta<1.20$, positive proportionate with the threshold value) before inverting the transformation. But β can not be too large, otherwise it will lead to the overflow of gray-scale value of the reconstructed image. Due to the nonlinear nature of shrinkage function, compensation operation differs from simply enhancing the contrast level of the reconstructed image. The effectiveness of the compensation operation will be proved by our experimental results in Section 5.

Therefore, the modified algorithm of sparse code shrinkage is summarized as follows:

1. Using a set of representative noise-free data with the same statistical properties as the data that we wish to denoise, estimate the sparse coding transform **W** by first estimating the ICA transform matrix and then orthogonalizing it.

2. Observing noisy vectors $\mathbf{x}^{(j)}$

(a) Transform each vector into the sparse basis by $\mathbf{W}\mathbf{x}^{(j)}$.

(b) Apply the estimated nonlinearity **g** to each component i of each vector j: $\mathbf{s}^{(j)}=\mathbf{g}[\mathbf{W}\mathbf{x}^{(j)}]$.

(c) Multiply each component of each vector $\mathbf{s}^{(j)}$ with the compensation factor β : $\mathbf{s}^{(j)}=\beta\times\mathbf{s}^{(j)}$.

(d) Perform the inverse transform to each vector $\mathbf{s}^{(j)}$ to get the denoised vectors $\mathbf{v}^{(j)}=\mathbf{W}^{-1}\mathbf{s}^{(j)}$.

Here, the nonlinearity **g** is a shrinkage function.

4. Relieving the Complexity of Computation

4.1. Why SCS is computationally demanding?

The denoising procedure of SCS is computationally demanding for the following two reasons:

One is that the dimension of ICA transform is rather considerable. Usually, if the size of block processing for an image is $N\times N$, the corresponding dimension of the transform is N. However, linear ICA yields the same number of basis vectors and filters as the number of degrees of freedom of the input. Thus, the dimension of SCS will soar to considerable N^2, because each patch in size of $N\times N$ will be reshaped into a column vector in size of $N^2\times 1$ as the input of ICA transform.

Another reason is that a sliding window approach [20] is used to combat a blocking artifact. A simple way to apply SCS to images could be to divide the image into distinct sub-windows in size of $N\times N$, and denoise each sub-window separately. This approach, however, ignores statistical dependencies across the synthetic edges, resulting in a blocking artifact. This problem has been solved by taking a sliding window approach. We do not divide the image into distinct windows, but denoise every possible $N\times N$ window of the image. We then have N^2 different suggested values for nearly each pixel [27], and determine the final result as the mean of these values. Therefore, the complexity of computation will be greatly increased.

4.2. Countermeasure

Increasing the step-length of the sliding window is an effective way to improve the efficiency of SCS. For example, Let the step-length=2, then we have $N^2/4$ different values for each pixel. This means that the efficiency of SCS is improved by 4 times. In practice, we can hardly tell any difference between the result of the original approach and that of the improved method. If let the step-length=4, we only have $N^2/16$ different values for each pixel, which means that the efficiency is considerably improved by 16 times, only leading to a slight degradation of the image quality. We show in Fig. 3 the denoising results of SCS using different step-length for sliding window.

Non-square window approach is another way to further improve the efficiency of SCS. From preceding analysis, decreasing the window size can also simplify the computation. But empirically, block processing of too small size can not eliminate low-frequency noise completely. So, we propose a new non-square window ($N/2\times N$) method to further relieve the computational complexity of SCS. Non-square ICA filter-bank has similar performance in denoising as square ICA filter-bank (to be attested in Section 5) while reducing half dimensionality of ICA transform. An example of non-square ICA basis is displayed in Fig. 4.

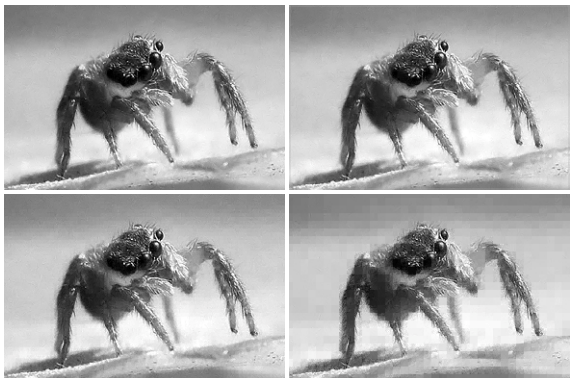

Fig. 3: Denoising with different sliding window step. Upper left: Results of step-length=1. Upper right: Step-length=2. Lower left: Step-length=4. Lower right: Step-length=8.

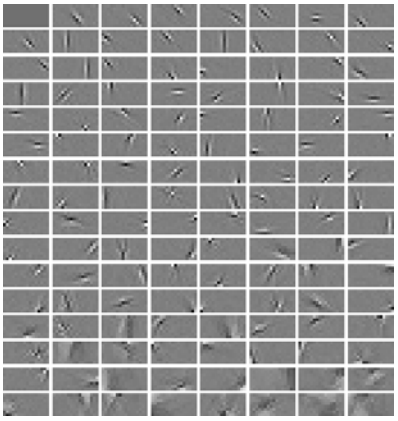

Fig. 4: Orthogonalized non-square ICA filters (8×16 patch).

5. Experiments

5.1. Image database

For the purpose of simplicity for making comparison between various image denoising methods, we tested the performance on images which were artificially corrupted with noise. We chose two separate data sets, in order to compare the performance of the results on different data sets. The first set of images comprises natural scenes which are hoped to reflect truly natural images free of human-imposed structure. The second set is intended to represent images of the human-built world, which are called man-made scenes and have quite different statistics from the natural scenes. Our

natural scene image dataset includes 489 images such as forests, mountains, insects, open landscapes and other natural scene images. The manmade scene image dataset includes 306 images such as tall-buildings, streets, highways, indoor-scenes or other urban scene images. We used the two image dataset to estimate the ICA transform, and picked some separate images for the actual denoising experiments.

5.2. Transform estimation

In order to estimate ICA transform, we first linearly normalize each image so that pixels have zero mean and unit variance. For the first phase, training patches in size of 4×8, 8×8, 8×16 and 16×16 are respectively selected from the images at random locations. Each patch is reshaped to a column vector as the input data of ICA networks. The matrix **X** is composed by 20000 training patches. PCA method is used to whiten the matrix **X** in order to remove the correlation between the pixels. Then these pre-processed data are used as the input to the ICA algorithm showed in Section 2. The ICA networks are updated their weights as Eqs. (3)–(5) to obtain the ICA filter-banks. In addition, corresponding PCA transforms for each training set are also estimated.

5.3. Component statistics

Since the denoising procedure is based on the property that individual components in the transform domain have sparse distribution, it is obvious that it must be tested how well this requirement holds. Measuring the sparseness of the distributions can be done by the normalized kurtosis, the most widely used non-Gaussianity measure, which is defined as

$$k(s) = \frac{E\{s^4\}}{(E\{s^2\})^2} - 3 \ . \qquad (9)$$

The average sparseness for each of these transforms (PCA, ICA and orthogonalized ICA) is calculated the following way: First, 3,000,000 image patches are selected from the image dataset which is used to estimate the transform. Then, these data are transformed using these PCA, ICA and orthogonalized ICA transforms respectively. The normalized kurtoses for each component of each transform are separately calculated. The mean of these component kurtoses is displayed in Fig. 5 for each transform of each dataset. Because of the sparse structure in the images, all these transforms show super-Gaussian distributions, indeed even the individual pixel values show a mildly super-Gaussian distribution when the local mean has been subtracted. From the graph, it can be seen that the ICA transform clearly finds a sparser representation of natural image data than PCA transform. Also, note that the data obtained by the orthogonalized ICA transform is not quite as sparse as that obtained by the standard ICA transform, but it is still far more super-Gaussian than PCA on average. It is worth noticing that the normalized kurtoses for non-square ICA filters (4×8 or 8×16) are approximately equal to the ones for square ICA filters (8×8 or 16×16). These figures prove that non-square ICA filters can also transform image data into a sparse representation as the square ones.

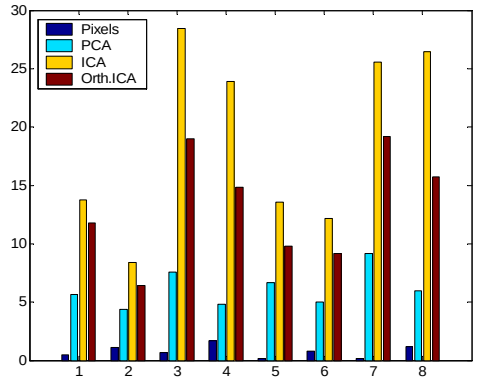

Fig. 5: Mean of normalized kurtosis of components for the different datasets and different transforms. (1) Natural scenes, 8×8 patch. (2) Natural scenes, 16×16 patch. (3) Man-made scenes, 8×8 patch. (4) Man-made scenes, 16×16 patch. (5) Natural scenes, 4×8 patch. (6) Natural scenes, 8×16 patch. (7) Man-made scenes, 4×8 patch. (8) Man-made scenes, 8×16 patch.

5.4. Applying SCS to images

A simple way to apply the method to images could be to divide the image into distinct 8×8 windows, and denoise each window separately. But this approach ignores statistical dependencies across the synthetic edges, resulting in a blocking artifact. This problem can be solved by taking a sliding window approach. We do not divide the image into distinct windows, but denoise every possible 8×8 window of the image. We then have 64 different values for nearly each pixel, and select the final result as the mean of these values. But we do not invariably have 64 different values for the pixels close to the image border, resulting in a "dark border" in the reconstructed image. Therefore, we solve such dark-border problem with the '88_table' proposed in [27] as a dividing matrix.

Because ICA transform applied to image data usually produces one component representing the local mean intensity of the pixels, which generally has a distribution that is not sparse and should be treated differently from other sparse components, we first

subtract in all experiments the local mean, and then estimate the suitable sparse encoding filters for other components. After denoising, to restructure the image properly, we add the local means again. We normalize the local variance in an image by dividing each image window by its norm just as some image processing method. This can be done in both the ICA transform estimation and the denoising procedure.

5.5. Denoising results

Several images are randomly selected from our image dataset for denoising, and Gaussian noise of different level is added. The SCS method is subsequently applied using hard-threshold, soft-threshold, and our steady-transition threshold with a compensatory factor. Fig. 6 shows the denoising results of our experiments. Visually, the method using hard-threshold gives a good noise reduction while retaining the features in the image. But when we inspect closely, we can find in images some artificial noise caused by the discontinuity of shrinkage function. Although SCS with soft-threshold seems to achieve a desirable performance, some edge features will still diminish after shrinkage. The compensation operation after shrinkage, which is different from simply enhancing the contrast of reconstructed images, and the steady-transition threshold make our results more superior and retain more features. It is evident that our method presents a sensible improvement in the quality of denoised images, especially in the edge preserving capability.

Then, our experiments are also conducted using such traditional method as Wiener filter and wavelet shrinkage to give a comparison. There are a large number of different variants of the wavelet shrinkage method [21]-[23], differing in choice of wavelet basis as well as the choice of shrinkage function. No one choice would have made a fair comparison, and thus we use Matlab function *wdencmp* to accomplish the wavelet shrinkage in this paper. Our method retains those features which are clearly visible in the noisy data but cuts out anything which is probably a result of the noise. Some structures of denoised image are almost as sharp as those in the original. Thus, it reduces noise effectively, at the expense of sometimes cutting out small features (usually textures) which were in the original image. It is easily seen that our SCS method outperforms the traditional method, highlighting its particular effectiveness on 'difficult' images that present sharp edges and many little details. We give in Table 1 an objective estimation of the quality of the results, evaluated by means of the peak signal-to-noise ratio (PSNR = 20log10 (255/RMSE)), where RMSE is the root mean square error. For a comparison purpose, we show both the results using traditional methods and those obtained by ordinary SCS with soft-threshold or hard-threshold. It can be seen that the steady-transition threshold is better (increasing about 0.6db of PSNR) than the other two thresholds. When using the compensation operation, we can also improve the performance by some 0.5-0.6(db). If we use the steady-transition threshold and compensation operation simultaneously, PSNR can be increased by some 1.2(db) on average from that of ordinary SCS. These experiments have attested the effect of our improvements for SCS.

In addition, we have conducted experiments with different step-length for sliding window and non-square ICA filters. PSNRs of them are shown in Table 2 and Table 3 respectively. In Table 2, we can see that the PSNR will decrease steadily with the increase of step-length for sliding window. But the objective numerical error functions seldom tell the whole truth. Actually, we can hardly tell any difference between the denoising result with step-length=1 and that with step-length=2 (which have been shown in Fig. 3). Amazedly, we notice in Table 3 that our method using non-square 4×8 filters can achieve similarly good performance with the square 8×8 ones. But when we use the 4×4 filter-bank, the PSNR will be decreased obviously. That's why we employ non-square ICA filter-bank to alleviate the computational pressure. For example, complexity of computation with 4×8 filters and step-length=2 can be decreased by eight times at the reduction of some 0.4db PSNR.

Fig. 6: Sparse code shrinkage experiments in denoising an image. Upper left: The noisy image (noise level 0.5). Upper right: SCS with hard-threshold. Lower left: With soft-threshold. Lower right: With steady transition threshold and compensating operation.

noise level	0.1	0.2	0.3	0.4	0.5
noisy image	32.27	26.25	22.73	20.21	18.26
wiener filter	35.75	32.28	29.75	28.21	26.96
wavelet shrink	35.79	31.53	29.20	27.68	26.55
hard-thr. ica	35.66	31.59	29.12	28.30	27.25
soft-thr. ica	35.98	31.91	29.28	28.42	27.43
steady-thr. ica	36.63	32.57	30.48	29.02	27.97
soft-thr. comp.ica	36.58	32.46	30.55	29.13	27.92
steady-thr. comp.ica	**37.10**	**33.16**	**31.13**	**29.63**	**28.62**

Table 1: PSNR(db) of different denoising method.

noise level	noisy image	8×8 step=1	8×8 step=2	8×8 step=4
0.1	32.27	**37.10**	36.69	35.29
0.2	26.25	**33.16**	32.78	31.26
0.3	22.73	**31.13**	30.75	29.29
0.4	20.21	**29.63**	29.18	27.61
0.5	18.26	**28.62**	28.20	26.72

Table 2: PSNR(db) of SCS with different step-length.

noise level	noisy image	8×8	4×8	4×4	6×6
0.1	32.27	37.10	**37.17**	37.00	37.03
0.2	26.25	33.16	**33.22**	33.06	33.17
0.3	22.73	**31.13**	**31.13**	30.84	31.08
0.4	20.21	29.63	**29.67**	29.28	29.63
0.5	18.26	**28.62**	28.57	28.10	28.51

Table 3: PSNR(db) of SCS with ICA basis of different size.

6. Application

It is clear that our method is suitable for denoising those images which consist of sharp object boundaries and little textured areas. Textures are in fact extremely difficult to denoise (even the human visual system must use a bunch of prior information to distinguish texture from the noise).

Fortunately, infrared images comprise few texture details because the texture can't give off much energy taken in by infrared sensors. More often than not, they possess many sharp object boundaries. So, our method is very suitable for the infrared image or video denoising. Fig. 7 shows the denoising results of some genuine infrared images (frames grabbed from infrared videos) with unknown distributed noise. In order to reserve the real performance of denoising, we insert these demo images with original size (640×480 pixels). When magnified, the noise and more details will be seen clearly. It can be seen that the noise is removed effectively from original images while retaining most of features. Especially, those interesting objects such as vehicles on the road are retained perfectly after the denoising procedure. The excellent performance may be attributed to both the steady-transition threshold shrinkage function and the compensation operation in sparse code shrinkage.

Another application of our method is consumer's digital photographs denoising. A photograph captured by a digital camera (Canon Powershot A75) with high sensitivity (ISO 200) is illustrated in Fig. 8. Usually, high sensitivity of digital camera will give rise to some unexpected noise. Here, we can find some annoying noise spread over the original picture. On the right hand is the corresponding picture denoised with our method. It is worth noting that the farina clung to the body of the flying bee, which might be blurred by other denoising method, seems to be intact after our denoising procedure.

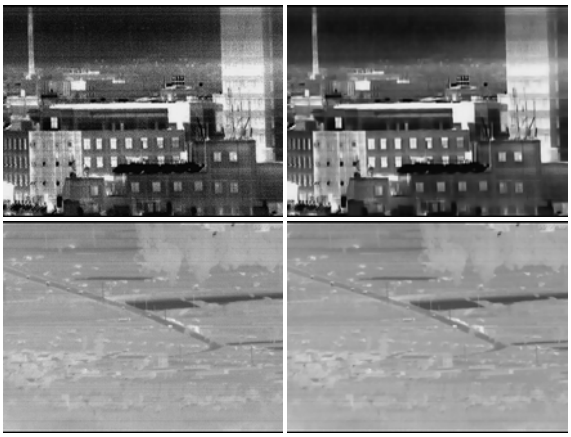

Fig. 7: Infrared video denoising. Left: Infrared images with unknown distributed noise. Right: Denoised with our ICA method.

Fig. 8: Digital photograph denoising. Left: A photograph captured by a digital camera with high sensitivity. Right: Denoised with our ICA method.

7. Conclusions

Sparse coding based on ICA can be applied to image feature extraction, producing a filter-bank for image windows. As a practical application of such filters, we introduced the method of sparse code shrinkage (SCS). We also introduced our improved method using the shrinkage compensation and a new shrinkage function which restrains the loss of edge features caused by

soft-threshold. The adoption of non-square ICA filters and new sliding window approach can greatly reduce computational complexity of SCS. The experimental results confirmed that our modified SCS method can reduce noise more effectively and efficiently without blurring edges or other sharp features in images. It performs much better than other traditional denoising method and outperforms the ordinary SCS method.

Acknowledgement

This work is partially supported by National Natural Science Foundation of China (Grant No: 60162001). The authors would like to thank Zhijie Zheng and Liming Zhang for helpful comments. Correspondence should be addressed to Jian Yang.

References

[1] D.H. Hubel and T.N. Wiesel, Receptive fields and functional architecture of monkey striate cortex. *Journal of Physiology*, 195:215-243, 1968.

[2] H.B. Barlow, Unsupervised learning. *Neural Computation*, 1:295-311, 1989.

[3] J.J. Atick, Could information theory provide an ecological theory of sensory processing? *Network*, 3:213-251, 1992.

[4] E. Oja, Principal components, minor components, and linear neural networks. *Neural Networks*, 5:927-935, 1992.

[5] R. Linsker, Local synaptic learning rules suffice to maximize mutual information in a linear network. *Neural Computation*, 4:691-702, 1992.

[6] P.J.B. Hancock, R.J. Baddeley, and L.S. Smith, The principal components of natural images. *Network*, 3:61-72, 1992.

[7] D.J. Field, Relations between the statistics of natural images and the response properties of cortical cells. *Journal of the Optical Society of America A*, 412:2370-2393, 1987.

[8] D.J. Field, What is the goal of sensory coding? *Neural Computation*, 6:559-601, 1994.

[9] H.B. Barlow, What is the computational goal of the neocortex? *Large-scale neuronal theories of the brain*, MIT Press, Cambridge, MA, 1994.

[10] N. Intrator, Feature extraction using an unsupervised neural network. *Neural Computation*, 4:98-107, 1992.

[11] B.A. Olshausen and D.J. Field, Natural image statistics and efficient coding. *Network: Computation in Neural Systems*, 7:333-339, 1996.

[12] A.J. Bell and T.J. Sejnowski, An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7:1129-1159, 1995.

[13] P. Comon, Independent component analysis, a new concept? *Signal Processing*, 36:287-314, 1994.

[14] P.J. Huber, Projection pursuit. *The Annals of Statistics*, 13:435-475, 1985.

[15] S. Haykin, *Neural Networks A Comprehensive Foundation*, 2nd Edition, Prentice Hall, 2001.

[16] B.A. Olshausen and D.J. Field, Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607-609, 1996.

[17] A.J. Bell and T.J. Sejnowski, The 'independent components' of natural scenes are edge filters. *Vision Research*, 37:3327-3338, 1997.

[18] L. Zhang and J. Mei, Shaping up simple cell's receptive field of animal vision by ICA and its application in navigation system. *Neural Networks*, 16:609-615, 2003.

[19] A. Hyvärinen, Sparse code shrinkage: Denoising of nongaussian data by maximum likelihood estimation. *Neural Computation*, 11:1739-1768, 1999.

[20] A. Hyvärinen, P.O. Hoyer, and E. Oja, Image denoising by sparse code shrinkage. *Intelligent Signal Processing*, IEEE press, 2000.

[21] D.L. Donoho, I.M. Johnstone, G. Kerkyacharian, and D. Picard, Wavelet shrinkage: asymptopia? *Journal of the Royal Statistical Society ser. B*, 57:301-337, 1995.

[22] D. Cho and T.D. Bui, Multivariate statistical modeling for image denoising using wavelet transforms. *Image Communication*, 20:77-89, 2005.

[23] G. Deng, D.B.H. Tay, and S. Marusic, A signal denoising algorithm based on overcomplete wavelet representations and Gaussian models. *Signal Processing*, 87:866-876, 2007.

[24] A. Hyvärinen, Fast and robust fixed-point algorithms for independent component analysis. *IEEE Trans. on Neural Networks*, 10:626-634, 1999.

[25] A. Hyvärinen and E. Oja, Independent component analysis: algorithms and applications. *Neural Networks*, 13:411-430, 2000.

[26] D.L. Donoho and I.M. Johnstone, Threshold selection for wavelet shrinkage of noisy data. *Proceedings of the 16th Annual International Conference of the IEEE*, 1:24-25, 1994.

[27] Ying Yu and Jian Yang, A new method of image feature extraction and denoising based on independent component analysis. *Proceedings of the 2006 IEEE International Conference on Robotics and Biomimetics*, pp. 380-385, 2006.