Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2014, Article ID 108393, 9 pages
http://dx.doi.org/10.1155/2014/108393

Research Article

Image Denoising Using Fourth Order Wiener Filter with Wavelet Quadtree Decomposition

Issam Dagher and Catherine Taleb

Department of Electrical and Computer Engineering, University of Balamand, 100 Koura, Lebanon

Correspondence should be addressed to Issam Dagher; dagheri@balamand.edu.lb

Received 12 November 2013; Revised 20 January 2014; Accepted 20 January 2014; Published 2 March 2014

Academic Editor: Sos Agaian

Visual information transmitted in the form of digital images is becoming a major method of communication in the modern age, but the image obtained after transmission is often corrupted with noise. The received image needs processing before it can be used in applications. Image denoising involves the manipulation of the image data to produce a visually high quality image. This paper uses the fourth order nonlinear wiener filter with wavelet quadtree decomposition and median absolute deviation. It will be shown that this new algorithm is comparable to other algorithms like BM3D, LPG-PCA, and KSVD.

1. Introduction

In digital image processing, image denoising is a very important issue. Certain degradation will happen to the transmitted images [1]. This degradation can be noise or blurring. Blurring is a kind of bandwidth reduction due to some errors related to methods of capturing the photos. However, noise is related to the transmission medium errors or errors of measurements [2].

Image denoising is the estimation of the original image from the noisy image. Many methods have been used. Signal to noise ratio and mean square error are used as performance measures. The denoising concept can be represented by a system where the input is the noisy image and the output is the reconstructed image. We assume that the noise is an additive Gaussian noise given by

$$f(g) = \frac{1}{\sigma^2\sqrt{2\pi}} e^{-(g-m)^2/2\sigma^2}, \quad (1)$$

where g is the pixel of the image to which is added the noise, m is the mean of the Gaussian noise, and σ^2 is the variance of the noise. Usually the variance of the noise is unknown. It is estimated via a method called median absolute deviation (MAD) given by

$$\mathrm{MAD}_n = b \,\mathrm{med}_i \left| x_i - \mathrm{med}_j x_j \right|. \quad (2)$$

The MAD has the best possible breakdown point (50%, twice as much as the interquartile range), and its influence function is bounded, with the sharpest possible bound among all scale estimators [3]. The constant b is needed to make the estimator consistent for the parameter of interest. In the case of the usual parameter σ at Gaussian distributions, we need to set $b = 1.4826$ [3].

1.1. Denoising Using Wavelet Transform. Wavelet coefficients calculated by a wavelet transform represent change in the time series at a particular resolution. By considering the time series at various resolutions, it is then possible to filter out noise. Wavelet thresholding is explained as decomposition of the data or the image into wavelet coefficients, comparing the detail coefficients with a given threshold value, and shrinking these coefficients close to zero to take away the effect of noise in the data [4]. The image is reconstructed from the modified coefficients. This process is also known as the inverse discrete wavelet transform. During thresholding, a wavelet coefficient is compared with a given threshold and is set to zero if its magnitude is less than the threshold; otherwise, it is retained or modified depending on the threshold rule. Thresholding distinguishes between the coefficients due to noise and the ones consisting of important signal information. The choice of a threshold is an important point of interest. It plays a major

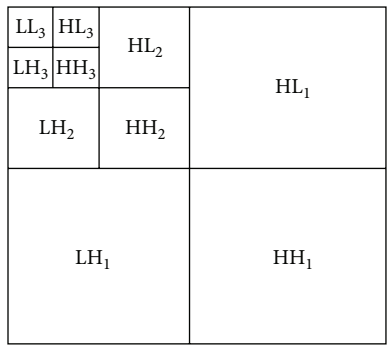

Figure 1: DWT on 2-dimensional data.

role in the removal of noise in images because denoising most frequently produces smoothed images, reducing the sharpness of the image. Care should be taken so as to preserve the edges of the denoised image. There exist various methods for wavelet thresholding, which rely on the choice of a threshold value. Some typically used methods for image noise removal include Visushrink and Sureshrink [4].

In Visushrink or Sureshrink, the image is first subjected to a discrete wavelet transform, which decomposes the image into various subbands [4] as shown in Figure 1.

The subbands HHk, HLk, LHk, $k = 1, 2, \ldots, j$ are called the details, where k is the scale and j denotes the largest or coarsest scale in decomposition. Note that LLk is the low resolution component. Thresholding is now applied to the detail components of these subbands to remove the unwanted coefficients, which contribute to noise. And as a final step in the denoising algorithm, the inverse discrete wavelet transform is applied to build back the modified image from its coefficients.

It should be noted that Visushrink uses a hard thresholding rule with a universal threshold $Th = \sigma\sqrt{2\log n}$, where σ^2 is the variance of the noise and n is the signal size [4].

The Sureshrink is a combination of the universal threshold and the SURE threshold. This method specifies a threshold value tj for each resolution level j in the wavelet transform which is referred to as level dependent thresholding [5].

In [6] a new threshold operator has been used. This method consists of thresholding the wavelet coefficients using fourth order polynomial and gave better results than the previous thresholding techniques. It should be noted that our fourth order Wiener filter is performed directly on the wavelet coefficients without thresholding.

1.2. Denoising Using Neighboring Wavelet Coefficients. Wavelet denoising by thresholding tends to kill too many wavelet coefficients that might contain useful image information. As a solution for this problem, wavelet thresholding is done by incorporating neighboring co-efficients [7]. The idea of considering the influence of other wavelet coefficients on the current wavelet coefficient to be thresholded is of great importance. A large wavelet coefficient will probably have large wavelet coefficients at its neighbor locations. The reason is that wavelet transforms produce correlated wavelet coefficients [7].

In the following wavelet denoising scheme, the neighboring coefficients are incorporated into the thresholding process. Suppose that $d_{j,k}$ is the set of wavelet coefficients of the noise 1D signal:

$$S_{j,k}^2 = d_{j,k-1}^2 + d_{j,k}^2 + S_{j,k+1}^2. \quad (3)$$

If (3) is less than or equal to λ^2, then the wavelet coefficient $d_{j,k}$ is set to zero. Otherwise, $d_{j,k} = d_{j,k}(1 - \lambda^2/S_{j,k}^2)$, where $\lambda = \sqrt{2\sigma^2 \log n}$, and n is the length of the signal [7].

1.3. KSVD or K-Means Clustering Process. Nowadays, it is important to search for sparse representations of signals using a dictionary matrix $D \in R^{n\times k}$ that contains k prototype signal atom for columns, $\{d_j\}_{j-1}^k$, and a signal $y \in R^n$ can be represented as a sparse linear combination of these atoms. The representation of y may either be exact $y = Dx$, or approximated. The vector $x \in R^k$ contains the representation coefficients of the signal y. Extraction of the sparse representation is a hard problem that has been extensively investigated in the past few years. It was assumed that the dictionary is known and fixed. Here the issue is to design the proper dictionary in order to better fit the sparsity model imposed. A full description of this algorithm is described in Figure 3 [8].

Each iteration (containing the sparse coding and the dictionary update) improves the denoising results because in this algorithm the work is to optimize $||y - Dx||_2^2$, the higher the number of iterations, the best performance is achieved [9].

1.4. Image Denoising by Sparse 3D Transform-Domain Collaborative Filtering. This is done by grouping similar 2D fragments of the image into 3D data arrays called "groups." In order to deal with 3D groups, collaborative filtering is used, and it includes 3 successive steps:

(1) 3D transformation of a group,

(2) shrinkage of transform spectrum,

(3) inverse 3D transformation.

After doing these steps, an array of jointly filtered 2D fragments is obtained. Due to the similarity between the grouped blocks, the transform can achieve a highly sparse representation of true signal, so that the noise can be well separated by shrinkage [10]. In this way, the collaborative filtering reveals even the finest details shared by grouped fragments and at the same time it preserves the essential unique features of each of individual fragments.

1.5. BM3D-SAPCA. Square image blocks containing fine image details, or sharp, or edges are examples where a nonadaptive transform is not able to deliver a sparse representation [11].

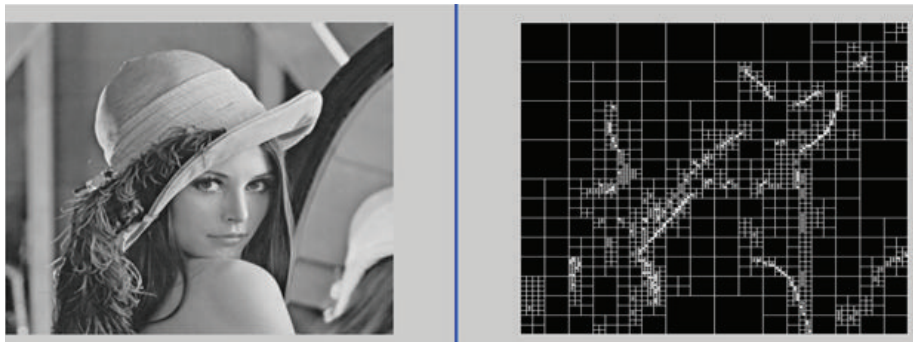

Figure 2: Quadtree decomposition of Lena.

Figure 3: The test images used.

In this case, denoising using BM3D filter is not effective. That is why in order to increase the sparsity of the true signal and improve the BM3D filter, similar adaptive-shape neighborhoods are grouping together and PCA is a part of the 3D transform used for collaborative filtering [11]. This method is called BM3D-SAPCA.

1.6. Two-Stage Image Denoising by Principal Component Analysis with Local Pixel Grouping. BM3D algorithm achieves important results in denoising, but the problem is that its implementation is complex [12]. Another algorithm has appeared with high efficiency and less complexity, the LPG-PCA algorithm or PCA based denoising method with local pixel grouping. In the PCA domain, noise can be removed from an image due to the reservation of only the most significant principal components [12].

The first stage is used to remove the most of noise in the image and the second stage to improve the output. The 2

stages have the same principle, only the noise parameter will be changing [13].

2. Wiener Filter in the Wavelet Domain

In this method, wavelet coefficients are considered conditionally independent Gaussian random variables. The noise is also considered as an independent stationary zero mean Gaussian variable [14]. Let us assume the relation between the noisy image and the original one as follows:

$$y_{i,j} = s_{i,j} + n_{i,j}, \tag{4}$$

where $y_{i,j}$ represent the coefficients of the noisy image in the wavelet domain, $s_{i,j}$ represents the coefficient of the original image, and $n_{i,j}$ represents the coefficient of the noise [14].

$n_{i,j}$'s are normal with zero mean and variance σ_n^2. Due to the decorrelation property of the orthogonal wavelet transform, the signal components $s_{i,j}$ are uncorrelated.

An optimal linear estimator for a signal in additive noise is formed as $s_{i,j} = ay_{i,j} + b$; the noise and the signal are assumed to be independent and a and b are chosen in a way to minimize the mean squared estimation error [15]:

$$J(a,b) = E\left[\left(\widehat{s}_{i,j} - s_{i,j}\right)^2\right] = E\left[\left(ay_{i,j} + b - s_{i,j}\right)^2\right]. \tag{5}$$

In order to get the minimum of, we have to get the derivatives with respect to a and b and set them equal to zero:

$$\frac{\partial J(a,b)}{\partial a} = 2E\left[\left(ay_{i,j} + b - s_{i,j}\right) y_{i,j}\right] = 0,$$
$$\frac{\partial J(a,b)}{\partial b} = 2E\left[\left(ay_{i,j} + b - s_{i,j}\right)\right] = 0. \tag{6}$$

This will give the following.

Consider $aE(y_{i,j}^2) + bE(y_{i,j}) - E(s_{i,j}\ y_{i,j}) = 0$.

Assume that $E(s_{i,j}) = 0$ and this implies that $E(y_{i,j}) = E(s_{i,j}) + E(n_{i,j}) = 0$.

Consider $aE(y_{i,j}^2) - E(s_{i,j}(s_{i,j} + n_{i,j})) = 0 \longrightarrow a = E(si, j^2)/E(yi, j^2)$.

And:

Consider $b = E(s_{i,j}) - aE(y_{i,j}) = 0$.

As a result we have the following.

Consider $s_{i,j} = a\ y_{i,j}$, where s is the best linear estimator of the signal component s; because the noise and the signal are independent, we can say that

$$E\left(y_{i,j}^2\right) = E\left[\left(s_{i,j} + n_{i,j}\right)^2\right]$$
$$= E\left(s_{i,j}^2\right) + \sigma_n^2 + 2E\left(n_{i,j}\ s_{i,j}\right)$$
$$= E\left(s_{i,j}^2\right) = E\left(y_{i,j}^2\right) - \sigma_n^2 \tag{7}$$
$$a = \frac{E\left(yi, j^2\right) - \sigma n^2}{E\left(yi, j^2\right)}.$$

Without loss of generality, we can assume that the $E\{y_{i,j}^2\}$'s can be determined by averaging the squared values of $y_{i,j}$ in a window centered at (i, j). This information can be expressed as

$$Q_{i,j} = \sum_{k=-R}^{R} \sum_{l=-R}^{R} y^2. \tag{8}$$

$M = (2R + 1)^2$ is the number of coefficients in the kernel:

$$E\left(yi, j^2\right) \sim= \frac{Qi, j}{M}. \tag{9}$$

Restricting the values of $E(y_{i,j}^2)$ to only positive values, the numerator of the above equation takes the form:

$$\widehat{s}_{i,j} = \max(0, a)\ y_{i,j}. \tag{10}$$

3. Fourth Order Nonlinear Wiener Filtering with QTD

3.1. Quadtree Decomposition. Applying QTD to an image will splits a parent block into four children blocks if the intensity gradient within the block is greater than a predefined threshold. The decomposition will stop when the final blocks are composed of pixels or coefficients that are close to each other, or the difference between the maximum value and the minimum value is less than a certain small threshold [3]. An example is shown in Figure 2.

In Figure 2, the right image represents the QTD of Lena image. It is clear how the values in each blocks are close and how the block sizes differ from smooth region to region where there are edges.

3.2. Summary of the Wavelet QTD + Wiener Algorithm. QTD is used with the wiener filtering algorithm in order to get a better performance. Instead of filtering the same block size, different block size related to the quadtree decomposition is used. The algorithm can be summarized as follows:

(1) Apply the discrete wavelet transform to the noisy image. It should be noted that we have used:

The dwt2(I,' d b4') matlab functions which uses the Daubechies Db4 wavelet.

It decomposes the image I into four subbands LL, LH, HL, and HH.

(2) Apply the QTD to each of the high frequency subbands (LH, HL, and HH). It should be noted that we have used:

The qtdecomp(b,Threhold) matlab function which returns a sparse matrix S containing the top left corner of each block coordinate and the size of each block. This function splits a block b if the maximum value of the block elements minus the minimum value of the block elements is greater

than threshold. Threshold is specified as values between 0 and 1.We have used a threshold of 0.2.The default value of the minimum block size is used. It should be noted that the effect of the variance of added noises will be decreased using the fourth order Wiener filter.

(3) Apply the Wiener filter on each variable size block:

We start with the center of the first block as the pixel to be estimated, after estimating the value, we shift the block by one.

It should be noted that we have used the fourth order filter because of its good PSNR performance with different noise variances. The first order filter for example, will give good PSNR for small values of noise variances. Its performance degrades when the noise variances increase.

(4) Apply the inverse wavelet transform on the filtered image.

3.3. Fourth Order Nonlinear Wiener Filter. In order to get the coefficients of this filter, the steps are summarized as follows:

The filter output is given by $\widehat{S}_{i,j} = ay_{i,j}^4 + by_{i,j}^3 + cy_{i,j}^2 + dy_{i,j} + e$; the error function is given by

$$J(a,b,c,d,e) = E\left\{\left[ay_{i,j}^4 + by_{i,j}^3 + cy_{i,j}^2 + dy_{i,j} + e - s_{i,j}\right]^2\right\}. \tag{11}$$

In order to get MMSE, we have to get the partial derivatives of (11) as follows:

$$\frac{\partial J(a,b,c,d,e)}{\partial a} = 0; \quad \frac{\partial J(a,b,c,d,e)}{\partial b} = 0;$$
$$\frac{\partial J(a,b,c,d,e)}{\partial c} = 0; \quad \frac{\partial J(a,b,c,d,e)}{\partial d} = 0; \tag{12}$$
$$\frac{\partial J(a,b,c,d,e)}{\partial e} = 0.$$

Now the system becomes

$$aE\left(y^8\right) + bE\left(y^7\right) + cE\left(y^6\right) + dE\left(y^5\right) + eE\left(y^4\right) - E\left(y^5\right) + E\left(ny^4\right) = 0,$$
$$aE\left(y^7\right) + bE\left(y^6\right) + cE\left(y^5\right) + dE\left(y^4\right) + eE\left(y^3\right) - E\left(y^4\right) + E\left(ny^3\right) = 0,$$
$$aE\left(y^6\right) + bE\left(y^5\right) + cE\left(y^4\right) + dE\left(y^3\right) + eE\left(y^2\right) - E\left(y^3\right) + E\left(ny^2\right) = 0,$$
$$aE\left(y^5\right) + bE\left(y^4\right) + cE\left(y^3\right) + dE\left(y^2\right) + eE(y) - E\left(y^2\right) + E(ny) = 0,$$
$$aE\left(y^4\right) + bE\left(y^3\right) + cE\left(y^2\right) + dE(y) + e - E(y) = 0. \tag{13}$$

The solution of the system above is as follows:

$$a = \frac{-k23}{k22} \qquad b = \frac{k15}{k16} - \frac{k17}{k16}a,$$
$$c = \frac{-k10 - k8b - k7a}{k9} \qquad d = -k_1c - k_2b - k_3a, \tag{14}$$
$$e = k_4a + k_5b + k_6c + E(y),$$

where

$$k1 = \frac{E\left(y^3\right) - E(y)E\left(y^2\right)}{E\left(y^2\right) - E^2(y)},$$
$$k2 = \frac{E\left(y^4\right) - E(y)E\left(y^3\right)}{E\left(y^2\right) - E^2(y)},$$
$$k3 = \frac{E\left(y^5\right) - E(y)E\left(y^4\right)}{E\left(y^2\right) - E^2(y)},$$
$$k4 = k3E(y) - E\left(y^4\right),$$
$$k5 = k2E(y) - E\left(y^3\right) \qquad k6 = k1E(y) - E\left(y^2\right),$$
$$k7 = E\left(y^6\right) - k3E\left(y^3\right) + k4E\left(y^2\right),$$
$$k8 = E\left(y^5\right) - k2E\left(y^3\right) + k5E\left(y^2\right),$$
$$k9 = E\left(y^4\right) + k6E\left(y^2\right) - k1E\left(y^3\right),$$
$$k10 = E(y)E\left(y^2\right) - E\left(y^3\right) + E\left(ny^2\right),$$
$$k11 = E(y)E\left(y^3\right) - E\left(y^4\right) + E\left(ny^3\right),$$
$$k12 = E\left(y^7\right) - k3E\left(y^4\right) + k4E\left(y^3\right),$$
$$k13 = E\left(y^6\right) - k2E\left(y^4\right) + k5E\left(y^3\right),$$

Table 1: Comparison between the 4 h order Wiener and other methods.

Methods	4th order Wiener with QTD	KSVD	BM3D	LPG-PCA
House				
$\sigma = 10$	36.2214	35.3585	36.1068	36.1742
$\sigma = 20$	33.1816	32.4790	33.1156	32.95
$\sigma = 30$	31.7216	30.5347	31.5037	31.06
$\sigma = 40$	30.8782	28.9990	30.0611	29.60
Cameraman				
$\sigma = 10$	34.2733	33.6567	34.0315	33.6807
$\sigma = 20$	31.5959	29.8680	30.4017	29.8169
$\sigma = 30$	28.7305	27.9519	28.5305	27.8619
$\sigma = 40$	27.7321	26.6280	27.1690	26.4450
Pepper				
$\sigma = 10$	33.2117	33.2631	33.6664	33.4680
$\sigma = 20$	29.2328	29.4737	29.9465	29.6773
$\sigma = 30$	27.9729	27.5576	28.0310	27.6634
$\sigma = 40$	26.3014	26.2714	26.6208	26.2851
Lena				
$\sigma = 10$	34.9483	34.3841	34.8737	34.9105
$\sigma = 20$	31.9964	30.6819	31.1590	30.9613
$\sigma = 30$	28.8193	28.4998	29.0058	28.6486
$\sigma = 40$	27.5619	26.9767	27.4874	27.1664

$$
\begin{aligned}
k14 &= E\left(y^5\right) - k1E\left(y^4\right) + k6E\left(y^3\right),\\
k15 &= \frac{k14k10}{k9} - k11 \qquad k16 = k13 - \frac{k8k14}{k9},\\
k17 &= k12 - \frac{k14k7}{k9},\\
k18 &= E\left(y^8\right) - k3E\left(y^5\right) + k4E\left(y^4\right),\\
k19 &= E\left(y^7\right) - k2E\left(y^5\right) + k5E\left(y^4\right),\\
k20 &= E\left(y^6\right) - k1E\left(y^5\right) + k6E\left(y^4\right),\\
k21 &= E\left(ny^4\right) - E\left(y^5\right),\\
k22 &= k18 - \frac{k17k19}{k16} - \frac{k7k20}{k9} + \frac{k8k17k20}{k9k16},\\
k23 &= \frac{k19k15}{k16} - \frac{k20k10}{k9} + k21 - \frac{k8k15k20}{k9k16}.
\end{aligned}
\tag{15}
$$

However, we only have the noisy image, not the noise. Knowing that the noise is Gaussian noise with zero mean leads to

$$
E\left(n^p\right) = \begin{cases} 0, & \text{if } p \text{ is odd},\\ \sigma_n^p\,(p-1)!!, & \text{If } p \text{ is even}, \end{cases}
\tag{16}
$$

where $(p\text{-}1)!!$ is the multiplication of all the odd numbers between $(p\text{-}1)$ and 0.

This will give

$$
\begin{aligned}
E\left(ny^2\right) &= 2\sigma_n^2 E\left(y\right),\\
E\left(ny^3\right) &= 3\sigma_n^2 E\left(y^2\right),\\
E\left(ny^4\right) &= 3\sigma_n^2 + 4\sigma_n^2\left(y^3\right) + 12\sigma_n^4 E\left(y\right),\\
k10 &= E\left(y\right)E\left(y^2\right) - E\left(y^3\right) + 2\sigma_n^2 E\left(y\right),\\
k11 &= E\left(y\right)E\left(y^3\right) - E\left(y^4\right) + 3\sigma_n^2 E\left(y^2\right),\\
k21 &= 3\sigma_n^2 + 4\sigma_n^2 E\left(y^3\right) + 12\sigma_n^4 E\left(y\right) - E\left(y^5\right).
\end{aligned}
\tag{17}
$$

4. Implementation and Results

We have done a comparison between the fourth order nonlinear wiener filtering with QTD and the BM3D, LPG-PCA, and KSVD algorithms. Four different images (house, Cameraman, Pepper, and Lena) were used with 4 different variances. The PSNR's (18) between the original $M \times N$ image $I(i, j)$ and the reconstructed image $Ir(i, j)$ of these methods are shown in Table 1:

$$
\mathrm{PSNR} = 10\log_{10}\left(\frac{255^2}{(1/MN)\sum_{i=1}^{M}\sum_{j=1}^{N}\left(I\left(i,j\right) - I_r\left(i,j\right)\right)^2}\right).
\tag{18}
$$

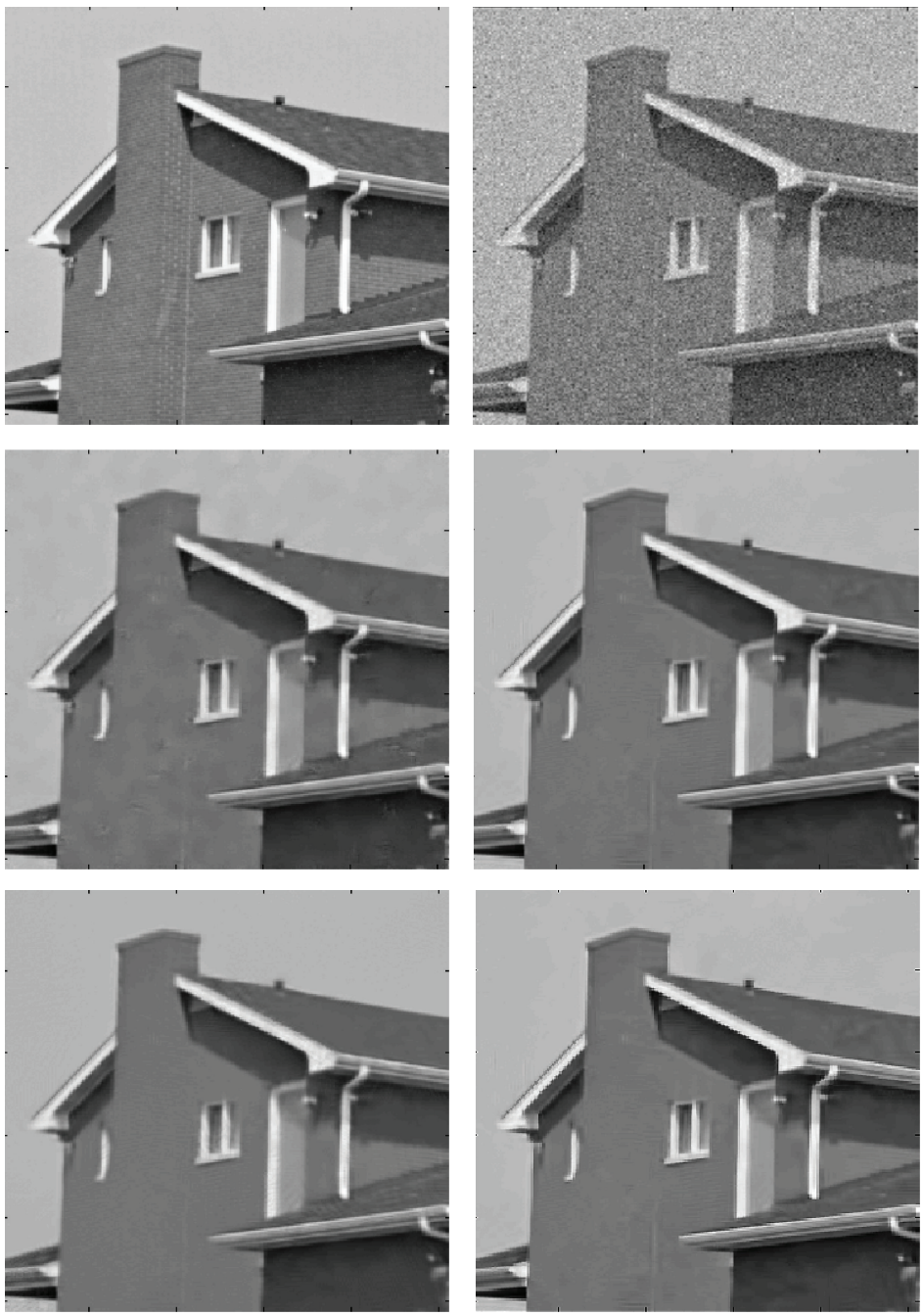

Figure 4: The denoising results of house by different schemes with $\sigma = 20$.

Figures 4 and 5 show the denoising results of the different methods for the "house" and "Lena" images using a variance of 400. (*From top to bottom, left to right*: Original image, Noisy image, KSVD, BM3D, LPG-PCA, 4th order Weiner results).

5. Conclusion

In this paper, we presented a new denoising method: the fourth order nonlinear wiener filter with wavelet quadtree decomposition and median absolute deviation. It is based on

(a) applying the discrete wavelet transform to the noisy image,

(b) applying the QTD to each of the high frequency subbands,

(c) applying the 4th order Wiener filter on each variable size block.,

(d) applying the inverse wavelet transform.

It was shown that this algorithm is comparable to other algorithms like BM3D, LPG-PCA, and KSVD algorithms.

Figure 5: The denoising results of Lena by different schemes with $\sigma = 20$.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] R. K. Castleman, *Digital Image Processing*, Prentice Hall, Upper Saddle River, NJ, USA, 1979.

[2] L. R. Lagendijk and J. Biemond, *Iterative Identification and Restoration of Images*, Kluwer Academic Publishers, Boston, Mass, USA, 1991.

[3] J. P. Rousseeeuw and C. Croux, "Alternatives to the median absolute deviation," *Journal of the American Statistical Association*, vol. 88, no. 424, pp. 1273–1283, 1993.

[4] D. L. Donoho, "De-noising by soft-thresholding," *IEEE Transactions on Information Theory*, vol. 41, no. 3, pp. 613–627, 1995.

[5] A. Antoniadis, J. Bigot, and T. Sapatinas, "Wavelet estimators in nonparametric regression: a comparative simulation study," *Journal of Statistical Software*, vol. 6, pp. 1–83, 2001.

[6] C. B. Smith, S. Agaian, and D. Akopian, "A wavelet-denoising approach using polynomial threshold operators," *IEEE Signal Processing Letters*, vol. 15, pp. 906–909, 2008.

[7] T. D. Bui, G. Y. Chen, and A. Krzyzak, "Image denoising using neighbouring wavelet coefficients," *IEEE Transactions on Image Processing*, vol. 2, no. 20, pp. 99–107, 2005.

[8] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

[9] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Transactions on Image Processing*, vol. 15, no. 12, pp. 3736–3745, 2006.

[10] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-D transform-domain collaborative filtering," *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2080–2095, 2007.

[11] K. Egiazarian, K. Dabov, A. Foi, and V. Katkovnik, "BM3D image denoising with shape-adaptive principal component analysis," in *Proceedings of the Workshop on Signal Processing with Adaptive Sparse Structured Representations*, 2009.

[12] L. Zhang, W. Dong, D. Zhang, and G. Shi, "Two-stage image denoising by principal component analysis with local pixel grouping," *Pattern Recognition*, vol. 43, no. 4, pp. 1531–1549, 2010.

[13] J. N. Ellinas, T. Mandadelis, and A. Tzortzis, "The statistical modeling of wavelet coefficients as a tool for image de-noising," in *Proceedings of the 2nd International Conference on Information Technology and Quality*, 2005.

[14] M. Kazubek, "Wavelet domain image denoising by thresholding and Wiener filtering," *IEEE Signal Processing Letters*, vol. 10, no. 11, pp. 324–326, 2003.

[15] P. Diniz, *Adaptive Filtering: Algorithms and Practical Implementation*, Springer, 2nd edition, 2010.