AREA/POWER EFFICIENT IMPLEMENTATION OF A WAVELET DOMAIN ROBUST IMAGE DENOISING SYSTEM.

Vijay Sundararajan[1] Michael E. Zervakis[2]
Keshab K. Parhi[1]

[1]Dept. of EE Univ. of Minnesota [2]Dept. of EE Univ. of Crete

ABSTRACT

Area/power efficient implementation is provided for a new wavelet domain robust image denoising algorithm. A novel approach to system implementation is presented which dramatically simplifies hardware implementation by using a Euclidean-norm approximation technique. Simulation results are presented, which show that the exact and norm approximation based implementations have comparable performance.

1. INTRODUCTION

Noise filtering in the wavelet domain is receiving increasing attention [1, 2, 3]. This trend is partially due to the following features: i) the ability to characterize signal and noise statistics more efficiently in the wavelet domain by dropping restrictive assumptions used in the signal domain, ii) the ability to design and apply test statistics that more closely fit the data process, and iii) as a result, the ability to estimate the signal more accurately in the wavelet domain rather than in the signal space.

However, most such sophisticated image processing systems are limited in practicality due to high complexity of hardware implementation. Presently, most of the existing, wavelet based image denoising, algorithms are based on signal classification by thresholding. The fixed threshold value $l_{vs} = \sigma\sqrt{2\log(N)}$ has been established by Donoho, et.al., [4]. In the presence of noise outliers, e.g., mixed-noise contamination, signal thresholding loses its effectiveness [5, 6]. To counter this, a robust wavelet-domain image denoising algorithm has been proposed in [7], which uses several, *complex*, wavelet domain vector operators. In this paper we present area/power efficient implementation of the various vector operators employed in [7].

This work suported in parts by grants from the National Science Foundation under grant number MIP-9258670 and the Army Research Office under grant number DA/DAAH-94-G-0405.

The reader is referred to [7] for detailed descriptions of the algorithm.

The rest of this paper proceeds as follows. Section 2 presents some preliminary aspects of the problem formation in the wavelet domain. The proposed algorithm is presented in Section 3, which also presents implementation issues of the algorithm in [7], whereas Section 4 presents several application examples . Finally, section 5 presents conclusions and summary.

2. PRELIMINARY DEFINITIONS

Consider the problem of signal denoising, where the samples $f(t_i)$ of the function $f(t)$, with normalized sampling points at $t_i = (i-1)/N$, must be recovered from N noisy samples

$$y_i = f(t_i) + z_i, \qquad i = 1, ..., N \tag{1}$$

where the noise sequence z_i is iid $N(0, \sigma^2)$. The denoising process derives an estimate vector $\hat{f} = \left(\hat{f}_i\right)_{i=1}^{N}$ of the original vector $f = (f(t_i))_{i=1}^{N}$, such as to minimize some risk function $R(f - \hat{f})$.

A compact matrix form of the wavelet transform is studied in [3]. Based on [3] the wavelet decomposition vector, g, of signal vector y is given by,

$$g = \mathbf{W}y \ , \tag{2}$$

where the matrix $\mathbf{W}$ encompasses filtering and decimation operations. Each sample g_i in the wavelet domain involves D values, depending on the depth of decomposition, i.e., $g_i = \left(g_i^k\right)_{k=1}^{D}$. In the separable 2-D DWT case, the matrix $\mathbf{W}$ involves Kronecker products of filtering and decimation operators.

3. VECTOR PROCESSING IN THE WAVELET DOMAIN

Consider the i^{th} pixel, g_i, of the wavelet-decomposed image at the first level of decomposition. Without loss

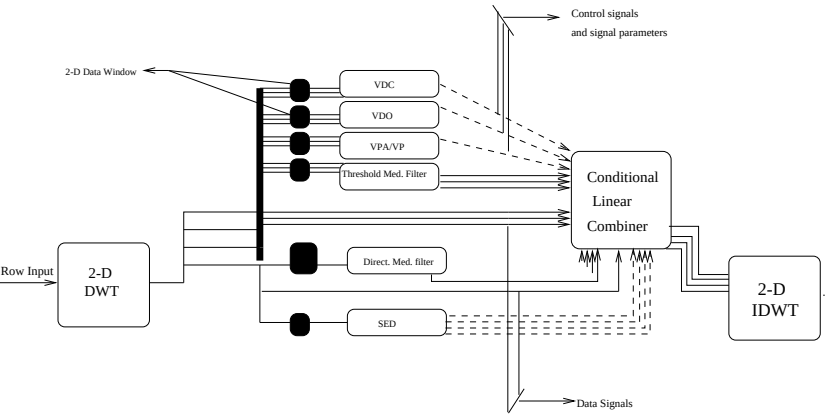


Figure 1: Simplified high level description of the new algorithm.

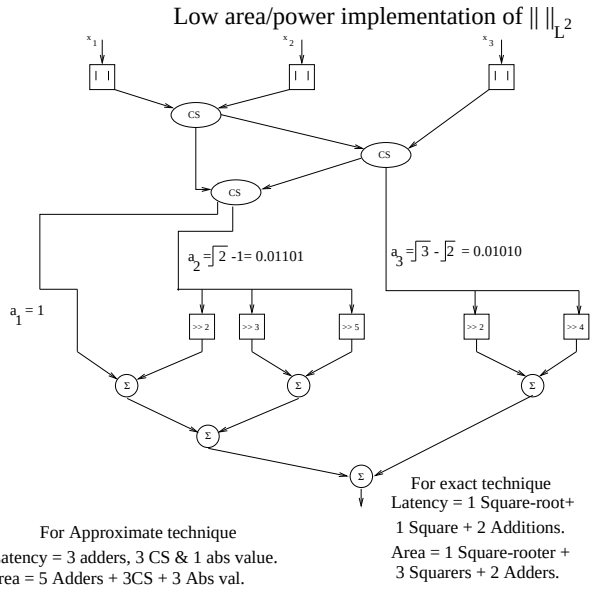


Figure 2: Example of approximate norm computation, CS is compare and swap.

of generality, we assume that the three first bands represent the high-detail bands, whereas the fourth band represents the low-detail band. The high-detail vector is given by, $x_i = \left(g_i^k\right)_{k=1}^3$. A simplified high level pictorial description of the algorithm presented in [7] is shown in Fig. 1. Implementation of operators used in [7] will be given next.

3.1. Vector Operators and their Implementation

An Euclidean norm-approximation algorithm was presented by Barni, et.al., in [8]. We employ this algorithm to simplify implementation of the vector operators in [7].

3.1.1. Euclidean Norm Approximation

In [8] the Euclidean norm of a vector is approximated by means of a linear combination of its components,

$$||x||_{L^2} \approx ||x||_{L^2_{approx}} = \sum_{i=1}^{N} a_i |x_{(i)}|. \qquad (3)$$

Where $x_{(i)}$ is the i^{th} ranked component of x obtained by sorting the components in decreasing order of their absolute values. In [8], use of, $a_i = \sqrt{i} - \sqrt{(i-1)}$, is advocated for the norm-approximation in (3). Since multiplication by a constant can be implemented using only additions, the approximate implementation has advantages of, a lesser latency, greatly reduced area and substantial reduction in power dissipation. Fig. 2 shows an example of hardware implementation of the above approximate algorithm, where the coefficients, a_i, have a word size = 5 and the vectors are 3-dimensional.

3.1.2. Implementation of VDO

Due to the orthogonality of the wavelet transform, edges with strong orientation in the image plane result in signal concentration in one of the wavelet bands. The VDO operator defines a measure of the proximity of the vector x_i to one of its coordinate axes. Its definition is based on the angular distance from x_i to each of its axes v_l . The set of squared angular distances at each pixel:

$$\phi_l(x_i) = \left| \frac{x_i^T \; v_l}{|x_i|} \right|^2, \qquad l = 1, 2, 3 \qquad (4)$$

performs like a probability distribution function ($\sum_{l=1}^{3} \phi_l(x_i) = 1$). Thus, its entropy, $e(i)$, provides a measure of dispersion on x_i and can be used for the definition of the VDO operator. The VDO operator is defined as:

$$\begin{aligned} VDO(i;k) &= e(i) \;\; if \;\; \phi_k(x_i) = max(\phi_l(x_i)) \\ VDO(i;k) &= 0 \quad otherwise. \end{aligned} \qquad (5)$$

The exact computation of Vector Directional Orientation (VDO) will require computation of an entropy. Instead, a much simpler implementation can be obtained by establishing a bijection between the entropy operator and a, computationally simpler, functional and requiring the bijection to preserve ordering, $<, =, >$, relationships, *almost everywhere*, see Fig. 3. This is, especially, true as we put a threshold on the entropy operator [7]. We define an alternative (to en-

$\alpha_1 + \alpha_2 + \alpha_3 = 1, \quad \alpha_1, \alpha_2, \alpha_3 \geq 0$

α_1	α_2	α_3	*Entropy*	*Approx norm*
0.0100	*0.0100*	*0.9800*	*0.8981*	*1.2892*
0.0100	*0.0990*	*0.9000*	*0.6990*	*1.1860*
0.0100	*0.1900*	*0.8000*	*0.5084*	*1.0337*
0.1000	*0.1000*	*0.8000*	*0.4183*	*0.9304*
0.1000	*0.2000*	*0.7000*	*0.2702*	*0.7884*
0.0100	*0.2900*	*0.7000*	*0.4041*	*0.8917*
0.2000	*0.2000*	*0.6000*	*0.1350*	*0.5316*
0.1000	*0.3000*	*0.6000*	*0.1827*	*0.6464*
0.0100	*0.3900*	*0.6000*	*0.3448*	*0.7651*
0.2000	*0.3000*	*0.5000*	*0.0628*	*0.3897*
0.1000	*0.4000*	*0.5000*	*0.1413*	*0.5226*
0.0100	*0.4900*	*0.5000*	*0.3245*	*0.6503*
0.3000	*0.3000*	*0.4000*	*0.0088*	*0.1329*
0.2000	*0.4000*	*0.4000*	*0.0398*	*0.2658*

We can clearly see that the monotonicity of entropy and the norm approximati are similar in this example.

Figure 3: A bijection between entropy and an alternate dispersion measure.

tropy) dispersion measure:

$$e(i) = \sqrt{\sum_l \sum_m (\sqrt{\phi_l(x_i)} - \sqrt{\phi_m(x_i)})^2},$$

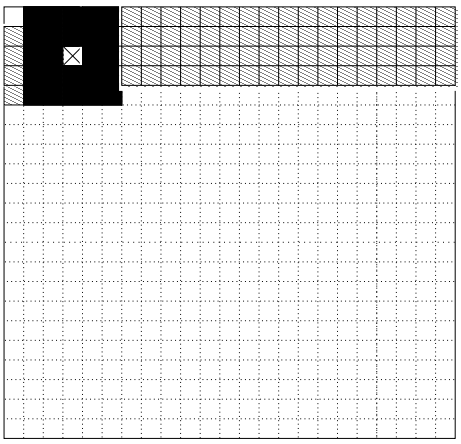
where $\sqrt{\phi_l(x_i)}$ represent the direction cosines of the vector x_i. The expression for $e(i)$ can be easily computed by the approximation in (3).

3.1.3. Implementation of VDC

Due to the signal dependent nature of the wavelet coefficients, we expect to find strong correlation among the vector orientations at the regions of edges. We define the VDC operator as:

$$\begin{aligned} VDC(i) &= \frac{1}{d-1}\left(\frac{\sum_{m,n,\ m\neq n,\ m,n\in sig()} |x_m^T||x_n|}{\sum_m |x_m|^2}\right) \\ & \quad m,n \in N\{i\} \ \ if\ ||N\{i\}||_{sig} = d \geq 3\ , \\ &= 0,\ otherwise. \end{aligned} \tag{6}$$

Where $sig()$ denotes the set of *significant edge pixels*. $||N\{i\}||_{sig}$ denotes the number of *significant edge pixels* in the neighborhood of pixel i. The requirement of at least 3 *significant edge pixels* in the neighborhood of pixel i serves to prevent pixels contaminated with impulses from having a high VDC. A large $VDC(i)$ value indicates strong concentration of similar vectors in the neighborhood of i. As with all other operators, the threshold for activating the VDC operator is very important. Small threshold values will cause noise to be falsely identified as edge, whereas large values may miss thin edges. In most experiments we used the rather conservative value 0.4-0.5, to avoid even small influence from noise.



Memory requirements for computation of VDC
The dark region represents the current window
The shaded region represents pixels for which storage is required.
The crossed pixel is the current pixel being processed.

Figure 4: Memory requirements for VDC computation by looking at lifetime of each DWT output

The exact computation of Vector Directional Correlation (VDC) requires computing eqn(6). With a 5×5 window this entails a worst case situation which requires an enormous number of multiplications (> 600) and additions (> 500) for computing the numerator and the denominator. Additionally, we also require a division to complete the computation. Storage requirements for VDC implementation can be gauged from Fig. 4.

Alternatively we can express VDC as,

$$\begin{aligned} VDC(i) &= \frac{1}{d-1}\left(\frac{||\sum_{m,\ m\in sig()} x_m||^2}{2 * \sum_{m,\ m\in sig()} ||x_m||^2} - \frac{1}{2}\right) \\ & \quad m \in N\{i\}\ if\ ||N\{i\}||_{sig} = d \geq 3\ , \\ &= 0,\ otherwise. \end{aligned}$$

By computing the numerator and denominator recursively using *running-sum* formulation; we can compute VDC very easily by using the norm-approximation in (3). In this way, for a system processed using a 5×5 window, hardware requirements come down to a worst case possibility of, 6 norm , 6 squaring, 13 additions and one division operations along with a scaling by $\frac{1}{d-1}$.

3.1.4. Implementation of SED

The conventional Scalar Edge Detector (SED) employed in [7], is given by,

$$num(i) = \sum_{m \in N\{i\}} \left(x_m - \frac{\sum_{n \in N\{i\}} x_n}{||N\{i\}||} \right)^2,$$

$$SED(i) = \frac{num(i)}{num(i) + N_0},$$

where N_0 is the variance of the, *inherent*, additive white Gaussian noise. If we consider, again, data-processing using a 5×5 window, we would require 25 squaring operations, about 50 additions, a mean computing filter and a division. Alternatively, first, express the SED in the following manner:

$$num(i) = \sum_{m \in N\{i\}} x_m^2 - (2||N\{i\}|| - 1) \left(\frac{\sum_{n \in N\{i\}} x_n}{||N\{i\}||} \right)^2$$

$$SED(i) = \frac{num(i)}{num(i) + N_0}.$$

Second, the above expression can be computed using simple recursive formulations for computing the *running-sum of squares*. As can be easily verified, for a 5×5 window, the resulting recursive formulation would require just 6 squaring operations, 5 additions, a mean computing filter and a division. We can further reduce the number of squaring to just two by grouping the DWT output to be squared ,*column-wise*, in groups of 5, and regarding the computation of the sum of squares as computation of the square of the Euclidean-norm of a 5-dimensional vector see Fig. 5.

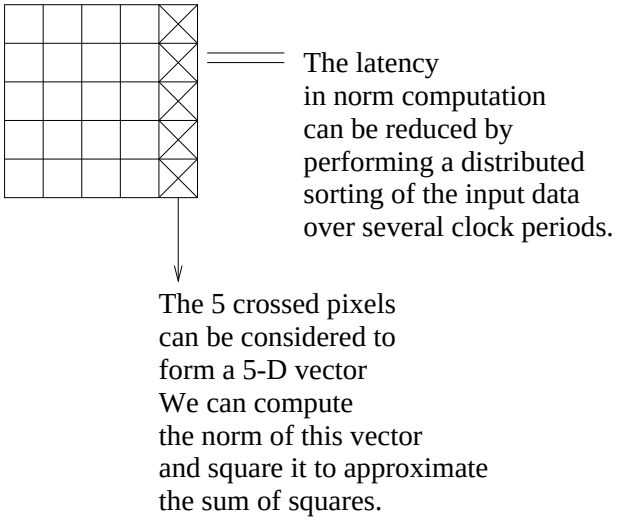


Figure 5: SED simplification using approximate norm computation

4. SIMULATION RESULTS

Results are shown for Cameraman, Lena and Airplane images which show that there is no loss in performance due to approximate implementation of the vector operators. The wavelet transform chosen was the *Daubechies* 8-tap wavelet [9]. We defined SNR in the conventional way as the ratio of signal power to noise power. The SNR in Gaussian noise case, for Cameraman = 13.86dB, Airplane = 9.9dB, Lena = 11.33dB, mixed noise consists of 4% impulse noise in addition to the Gaussian noise.

Image	Gauss. & Exact.	Mixed & Exact.	Gauss. & Approx.	Mixed & Approx.
Cameraman	3.94dB	3.54dB	3.89dB	3.73dB
Airplane	6.46dB	5.63dB	6.44dB	5.59dB
Lena	5.89dB	5.30dB	5.91dB	5.26dB

Table 1: Comparison of, SNR gain, in exact and approximate implementation of vector operators.

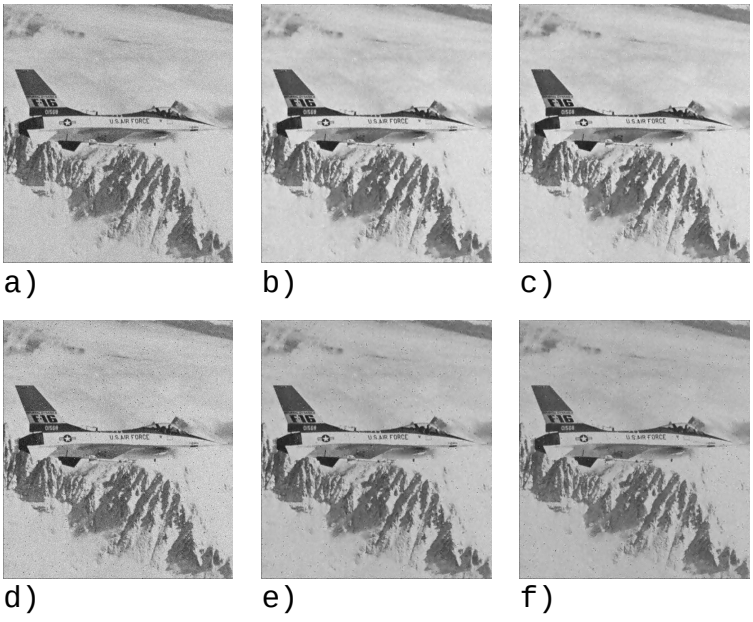


Figure 6: Gaussian noise a)Noisy image. Image restoration using: b)Exact scheme. c)Approx scheme. Mixed-noise d),e),f) same order as before.

5. CONCLUSION

This paper has presented a low area/power implementation of a novel robust algorithm for denoising of images using wavelet domain processing. The algorithm

is effective both in presence of Gaussian and mixed noise environments. Approximate versions of the operators used in the algorithm were developed. The use of approximate processing reduced the computational complexity considerably but did not degrade the performance.

6. REFERENCES

[1] M. Banham and A. Katsaggelos, "Spatially Adaptive Wavelet-Based Multiscale Image Restoration," *IEEE Transactions on Image Processing*, vol. 5, pp. 619–634, April 1996.

[2] M. Lazar and L. Bruton, "Combining the Discrete Wavelet Transform and Mixed Domain Filtering," *IEEE Transactions on Image Processing*, vol. 5, pp. 1124–1136, July 1996.

[3] M. Zervakis, T. Kwon, and J.-S. Yang, " Multiresolution Image Restoration in the Wavelet Domain," *IEEE Transactions on Circuits and Systems–II: Analog and Digital Signal Processing*, vol. 42, pp. 578–591, Sept. 1995.

[4] D. Donoho and I. Johnstone, "Ideal Spatial Adaptation by Wavelet Shrinkage," *Biometrika*, vol. 81, no. 3, pp. 425–455, 1994.

[5] M. Zervakis and A. Venetsanopoulos, "Linear and Nonlinear Image Restoration Under the Presence of Mixed Noise," *IEEE Transactions on Circuits and Systems*, vol. CAS-38, pp. 258–272, March 1991.

[6] S. Kassam and H. Poor, "Robust Techniques for Signal Processing: A Survey," *IEEE Proceedings*, vol. 73, pp. 433–481, March 1985.

[7] M. E. Zervakis, V. Sundararajan, and K. K. Parhi, "A Wavelet Domain Algorithm for Denoising in the Presence of Noise Outliers.," *To Appear in ICIP 97.*

[8] M. Barni, V. Cappellini, and A. Mecocci, "Fast Vector Median Filter Based on Euclidean Norm Approximation," *IEEE Signal Processing Letters*, vol. 1, pp. 92–94, June 1994.

[9] I. Daubechies, *Ten Lectures on Wavelets.* Philadelphia, Pennsylvania: SIAM, 1992.